The most important thing to know about data sets is the fact that they are organized into tables. Every single table contains rows and columns, also called information fields. Every row/line within a table has the name of record. The data that are organized in such manner (into independent tables) are relational data. The software application used for arranging data in such way is called relational database. SQL (Structured Query Language - programming language for querying) is the most frequently utilized programming language, being used for the retrieval and management of data stored in relational databases. The Visual Basic controls for databases play a significant role in establishing the connections between applications and data. Every table contains a primary key, which is used for the unique identifying of database’s recordings. In other words, the value of a primary key within a table can not be duplicated throughout the recording processes. For instance, a primary key could stand for a personal identification number, an e-mail address or another unique identifier (the persons’ names are not usually chosen as primary keys because there might be the possibility of a repetition).

Keywords: Basic, Connect to Database, Tables, SQL Server

In order to initiate a database working session, this has to be activated. For connecting to an existing database or to create a new one, the Server Explorer Window might be utilized (if it doesn’t appear, one may need to launch the window from the menu: View, Server Explorer). One needs to use the Add Connection option from the contextual menu by using the mouse right-click on Data Connections (the system Tools menu could also be accessed, followed by Connect to Database…). Using this dialogue one may select different types of databases, such as: Microsoft Access, SQL Server Database File, Oracle Database. Because the Visual Basic’s native database is SQL Server Database, one may specify the name of such database which exists within the personal computer or create a new one (Personnel for example, fig. 1):
One may add a new table within the database (*Personnel.mdf* here). The *Data* option appears in the *Visual Basic* menu, but the contextual menu might be also utilized when *right-clicking* on *Tables* (fig. 2).

### Data table's structure definition (Table Designer)

Within the *Table Designer* window one may define (*Table Definition*, fig. 3):

- the field names (Column name);
- *Data Type* within the field (character lines, decimals, date time, logical, etc.);
- the number of characters which is required for typing the field values;
- the number of decimals needed for the numerical data- for instance: *decimal (8,2)*;
- the fields which don’t require a completion (Allow Nulls);
- the primary keys (Set Primary key) – right-click on the field name.
Example:

✓ The Employee table will be defined in the Personnel.mdf database (fig. 4):

- The ID field is defined as a primary key.
- In order to define the data table’s structure (Open Table Definition) or its content (Show Table Data), one may utilize the contextual menu or the Data option from the main menu (fig. 5).

✓ Table’s content (fig. 6):
In order to visualize the form’s data (introduced within the project) one should activate the Add New Data Source… link for establishing a New Connection… with the Personnel.mdf database (fig. 7):

The Solution Explorer window contains the project’s elements. Please notice the PersonnelDataSet.xsd element – data set, database’s copy, on the local computer (fig. 8):

Visual Basic creates automatically the BindingNavigator and the DataGridView objects by dragging (using the mouse) the tables from Data Sources into the form. It also creates: DataSet, TableAdapter, BindingSource and BindingNavigator (fig. 9):
The program memorizes the data within the **DataSet** object which represent the entire database (fig. 10). The **TableAdapter** object copies data between database and **DataSet**. It also includes methods to facilitate the operations in relation to the database (the selection, the insertion, the update, the deletion of recordings).

The **BindingSource** object encapsulates all data from inside the **DataSet** and controls the data movement actions, as well as the procedure of elements’ addition and deletion.

![Diagram of database components](image)

**BindingNavigator** offers the interface through which the user may interact with the data source.

*Visual Basic* generates automatically code sequences. When loading the form, these instructions intermediate the data copying (through the **TableAdapter**) from the database into the **DataSet**. The **StudentiBindingNavigatorSaveItem_Click** event (control from **Binding Navigator**) allows the tables’ modifications to be saved in the database.

```vbnet
Public Class Form1

Private Sub EmployeeBindingNavigatorSaveItem_Click(ByVal sender As EventSender)
    Me.Validate()
    Me.EmployeeBindingSource.EndEdit()
    Me.EmployeeTableAdapter.Update(Me.PersonnelDataSet.Employee)
End Sub

Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As EventArgs)
    'TODO: This line of code loads data into the
    'PersonnelDataSet.Employee' table. You can move, or
    remove
    'it, as needed.
    Me.EmployeeTableAdapter.Fill(Me.PersonnelDataSet.Employee)
End Sub
End Class
```

1327
By using `DataGridView` and `BindingNavigator` controls, one may visualize on the form, data within the `Personnel` table (fig. 11):

One may add other controls on the form, for instance, a button that displays the income tax. The recordings within the table can be scanned sequentially, from the first to the last, each field’s values being available (fig. 12).

```vbnet
Private Sub btnIncomeTax_Click(ByVal sender As _, ByVal e As System.EventArgs)
    Dim i As Integer, name As String
    Dim Salary, IncomeTax As Integer

    For i = 0 To PersonnelDataSet.Tables("Employee").Rows.Count - 1
        name = PersonnelDataSet.Tables("Employee").Rows(i).Item("Name")
        Salary = PersonnelDataSet.Tables("Employee").Rows(i).Item("Salary")
        IncomeTax = Salary * 0.16
        MessageBox.Show(name & " " & IncomeTax)
    Next
End Sub
```

Even if it may be perceived as a difficult manner at the first sight, in order to grant access to the elements within the data set, one should demand a hierarchical class structure (collections and properties) that is to be represented in the following way (fig. 13):

- The form loading (`FormLoad`);
- The execution of the `Form1_Load` event procedure where the connection with the database is established and where the `Fill` method is used for filling the local data set.
- The database interrogation should be executed and the result is sent in the `PersonnelDataSet`.
- When pressing the `Income Tax` button the recordings within the local set are examined and the fields’ values are displayed.

**BIBLIOGRAPHY**