Week – 11 & 12: Database Programming

- Over View of ADO.Net
- Types of Connection Objects.
- Types of Data Adapter
- Data Set
- Connectivity with Data Bases Using OLEDB and SQL Connection Objects
- Connectivity with SQL Server
- Connectivity with Access and Oracle

Over View of ADO.Net

ADO.Net

ADO.NET is a data access technology from the Microsoft .NET Framework. It provides communication between relational and non-relational systems through a common set of components. ADO.NET is a set of object-oriented classes that provides a rich set of data components. Programmers use these components to access data and data services from the databases.

ADO.NET is a part of the base class library of Microsoft .NET Framework. It is used to create high-performance, reliable, and scalable database applications for client-server applications as well as distributed environments over the Internet and intranets.

In the ADO.NET model applications connect to the data sources when they are reading or updating the data. After that, the connection closes. The ADO.NET model utilizes XML to store the data in cache and transfer the data among applications. It supports disconnected data access through its "DataSet" component. ADO.NET uses SQL queries (statements) and stored procedures to read, write, update, and delete data from a data source.

Architecture of ADO.NET

ADO.NET is conceptually divided into *consumers* and *data providers*.

- The ***consumers*** are the applications that need access to the data.
- A ***data provider*** is a software component that interacts with a ***data source***. The *data providers* implement the interface for data access and provide the data to the consumer.

ADO.NET includes .NET Framework *data providers* for connecting to a database, executing commands, and retrieving results. Those results are either processed directly, or are placed in an ADO.NET **DataSet** object. The ADO.NET classes are found in **System.Data.dll**, and are integrated with the XML classes found in **System.Xml.dll**.

ADO.NET data providers are analogous to ODBC drivers, JDBC drivers, and OLE DB providers. Different *data store technologies* can have different capabilities. A single ADO.NET provider cannot implement every possible interface available in the ADO.NET standard.

ADO.NET providers can be created to access

- simple data stores such as a text file and spreadsheet
- complex databases such as Oracle Database, Microsoft SQL Server, MySQL, PostgreSQL, SQLite, IBM Db2, Sybase ASE, and others
- hierarchical data stores such as email systems

ADO.NET Components

The two main components of ADO.NET for accessing and manipulating data are:

1. the **.**NET Framework **data providers** and
2. the **DataSet**

.NET Framework Data Providers

The .NET Framework Data Providers are the components that are designed to

- manipulate data,
- access data in forward-only, and read-only mode

The following is the lists of data providers that are included in .NET Framework.

.NET Framework data provider	*Description*
.NET Framework Data Provider for SQL Server (SqlClient)	Provides data access for Microsoft SQL Server. Uses the **System.Data.SqlClient** namespace.
.NET Framework Data Provider for OLE DB (OleDb)	For data sources exposed by using OLE DB. Uses the **System.Data.OleDb** namespace.
.NET Framework Data Provider for ODBC (Odbc)	For data sources exposed by using ODBC. Uses the **System.Data.Odbc** namespace.
.NET Framework Data Provider for Oracle (OracleClient)	For Oracle data sources. The .NET Framework Data Provider for Oracle supports Oracle client software version 8.1.7 and later, and uses the **System.Data.OracleClient** namespace.
EntityClient Provider	Provides data access for Entity Data Model (EDM) applications. Uses the **System.Data.EntityClient** namespace.
.NET Framework Data Provider for SQL Server Compact 4.0.	Provides data access for Microsoft SQL Server Compact 4.0. Uses the **System.Data.SqlServerCe** namespace.

The four core objects that make up a .NET Framework data provider are:

1. The **Connection** object provides connectivity to a data source.
2. The **Command** object enables access to database commands to return data, modify data, run stored procedures, and send or retrieve parameter information.
3. The **DataReader** reads a forward-only, read-only stream of data from a data source.
4. The **DataAdapter** provides the bridge between the *DataSet* object and the *data source*. The *DataAdapter* uses *Command* objects to execute SQL commands at the data source to both

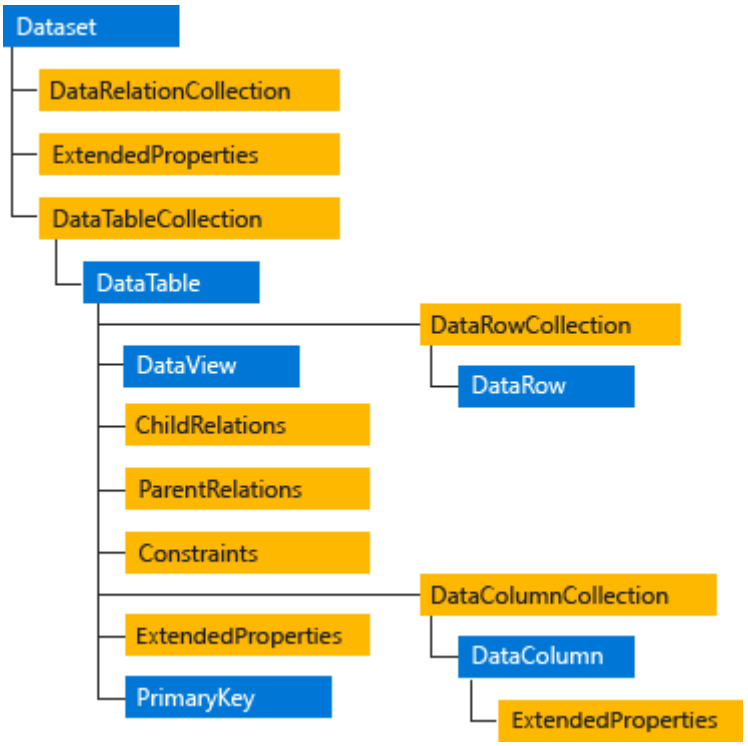
- load the DataSet with data and
- reconcile changes made in the DataSet back to the data source

The DataSet

The ADO.NET **DataSet** is designed for data access independent of any data source. It can be used

- with multiple and differing data sources,
- with XML data, or
- to manage data local to the application

The *DataSet* contains a collection of one or more **DataTable** objects. *DataTable* contains rows and columns of data. It may also contain primary key, foreign key, constraints, and relation information about the data in the DataTable *objects*.



Types of Connection Objects

A Connection object is used to connect to a specific *data source*. The necessary authentication information is provided to a Connection object as formatted string, called “connection string”.

The Connection object to be used depends on the type of data source. The base class for all Connection objects is the **DbConnection** class.

- The .NET Framework Data Provider for OLE DB includes an **OleDbConnection** object
- The .NET Framework Data Provider for SQL Server includes a **sqlconnection** object
- The .NET Framework Data Provider for ODBC includes an **odbcconnection** object

- The .NET Framework Data Provider for Oracle includes an **OracleConnection** object

Types of Data Adapter

A **DataAdapter** is used to retrieve data from a *data source* and populate tables within a DataSet. The DataAdapter uses the *Connection* object of the .NET Framework data provider to connect to a data source. It uses *Command* objects to retrieve data from and resolve changes to the data source.

The DataAdapter object to be used depends on the type of connection object. The base class for all DataAdapter objects is the **DbDataAdapter** class.

- The .NET Framework Data Provider for OLE DB includes an **oledbdataadapter** object
- The .NET Framework Data Provider for SQL Server includes **sqldataadapter** object
- The .NET Framework Data Provider for ODBC includes **odbcdataadapter** object
- The .NET Framework Data Provider for Oracle includes **OracleDataAdapter** object

Data Set

The ADO.NET DataSet is a memory-resident representation of data. It provides a consistent relational programming model regardless of the source of the data it contains. A DataSet represents a complete set of data including the tables that contain, order, and constrain the data, as well as the relationships between the tables.

There are several ways of working with a DataSet. It can be used independently or in combination.

- Programmatically create a DataTable, DataRelation, and Constraint within a DataSet and populate the tables with data.
- Populate the DataSet with tables of data from an existing relational *data source* using a *DataAdapter* object.
- Load and persist the DataSet contents using XML.

A strongly typed DataSet can also be transported using an XML Web service. A ***typed*** DataSet is a class that derives from a DataSet. It inherits all the methods, events, and properties of a DataSet. A *typed* DataSet provides strongly typed methods, events, and properties. This means you can access tables and columns by name, instead of using collection-based methods. A *typed* DataSet also allows the Visual Studio .NET code editor to automatically complete lines as you type. The strongly typed DataSet provides access to values as the correct type at compile time. With a strongly typed DataSet, type mismatch errors are caught when the code is compiled rather than at run time.

References:

- Professional Visual Basic 2010 and .Net, Bill Sheldon et.al, Wiley Publishing Inc., Chapter 10, 11, 12
- C# Database Basics, Michael Schmalz, O'Reilly, Chapter
- ADO.NET 4 Database Programming with C# 2010, Anne Boehm &Ged Mead, Mike Murach& Associates Inc., Chapter
- Mastering C# Database Programming, Jason Price, SYBEX Inc., Chapter
- Microsoft Visual Basic Guide
  - Working with data in Visual Studio
  - X
- https://en.wikipedia.org/wiki/ADO.NET
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/ado-net-overview
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/data-providers
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/dataset-datatable-dataview/
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/dataset-datatable-dataview/typed-datasets
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/connecting-to-a-data-source
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/retrieving-and-modifying-data
- https://learn.microsoft.com/en-us/dotnet/framework/data/adonet/dataadapters-and-datareaders
- x

Recommended Software:

- Microsoft SQL Server 2012
- Visual Studio 2012 with Visual Basic.net
- Dot Net Framework 2.0 or above