طراحی و پیاده‌سازی یک دفترچه تلفن

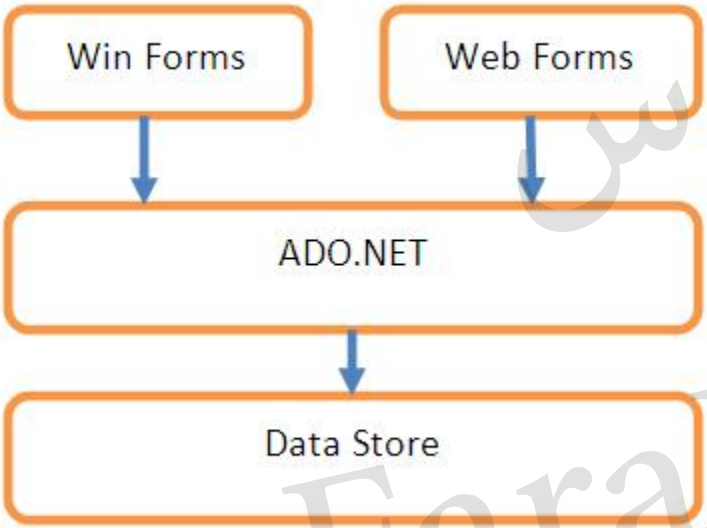
اهداف آموزشی:

- آشنایی با روش ایجاد بانک اطلاعاتی به کمک SQL Server
- آشنایی با تکنولوژی ADO.NET
- آشنایی با کلاس‌های موجود در ADO.NET
- معرفی نحوه ارتباط برقرار کردن C#.Net با بانک اطلاعاتی به روش ADO.NET
- آشنایی با نحوه ذخیره کردن، ویرایش کردن و حذف کردن داده‌ها
- ایجاد کردن یک جستجوگر پویا

بانک اطلاعاتی چیست؟

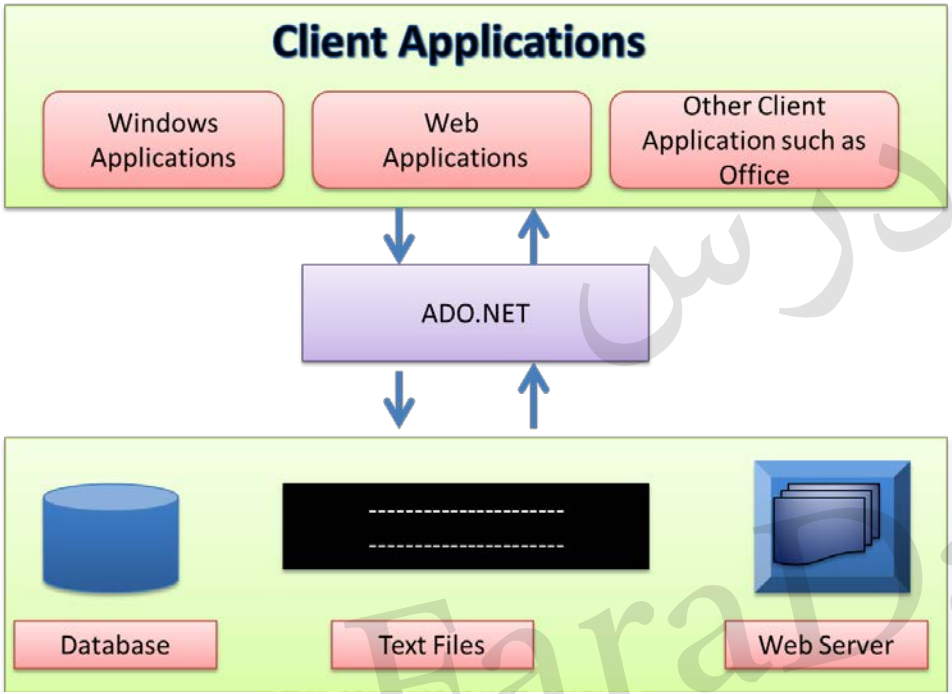
ذخیره و بازیابی اطلاعات یکی از مهمترین بخش‌ها در تولید یک محصول نرم افزاری می‌باشد. امروزه برای ذخیره و بازیابی اطلاعات در حجم زیاد از نرم افزارهای بانک اطلاعاتی همانند Access ,Sql Server Oracleو غیره استفاده می‌شود. در .Net روش‌های مختلفی برای ارتباط با بانک‌های اطلاعاتی ایجاد شده است که از جمله این تکنولوژی‌ها می‌توان ADO.NET، LINQو... را نام برد. در این بخش ابتدا اجزاء ADO.NET را بررسی کرده و در ادامه با استفاده از این تکنولوژی یک برنامه کاربردی را بصورت کامل پیاده‌سازی خواهیم کرد.

ADO.NET چیست؟



به مجموعه کلاس‌هایی که اشیائی از آن کلاس‌ها برای دسترسی به داده‌های یک بانک اطلاعاتی در .Net استفاده می‌شود، ADO.NET گفته می‌شود.

کاربرد ADO.NET



تکنولوژی ADO.NET برای ذخیره و بازیابی داده ها در هر نوع منبع داده ای کاربرد دارد. همچنین این تکنولوژی برای ذخیره بازیابی داده ها در هر نوع برنامه نویسی از تحت وب تا تحت ویندوز کاربرد دارد.

فضای نام های ADO.NET هستند؟

کلاس‌های اصلی و مشترک ADO در فضای نام System.Data قرار دارند. این فضای نام خود نیز شامل چند فضای نام دیگر است که مهمترین آنها عبارتند از System.Data.SqlClient و System.Data.OleDb. فضای نام System.Data.SqlClientحاوی کلاس‌هایی است که برای دسترسی به بانک‌های اطلاعاتی از نوع Sql Server مورد استفاده قرار می‌گیرد. فضای نام System.Data.OleDb نیز حاوی کلاس‌هایی است که برای دسترسی به بانک‌های اطلاعاتی از نوع OLE(مانند بانک‌های اطلاعاتی Access)استفاده می‌شود. در طی این فصل با استفاده از فضای نام SqlClient به بانک‌های اطلاعاتی از نوع SQL Server دسترسی خواهیم داشت.

کلاس های بکار برده شده در این بخش

ابتدا با تعدادی از مهمترین کلاس های ADO.NET که در طی مثال‌های این بخش بـه کـار رفتـه‌انـد آشـنا می‌شویم.

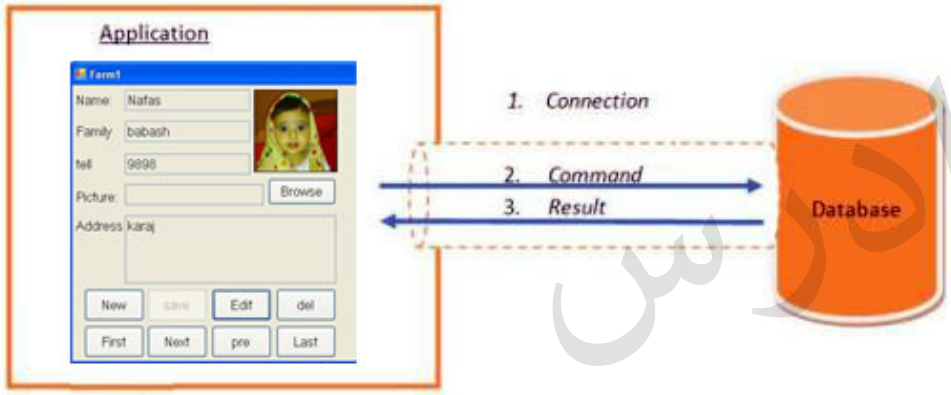
این کلاس‌ها عبارتند از:

SqlParameter - SqlDataAdapter - SqlCommand - SqlConnection

این کلاس‌ها فقط برای ارتباط با بانک‌های اطلاعاتی SQL Serverمورد استفاده قـرار مـی‌گیرنـد و در فضای نام System.Data.SqlClient قرار دارند. همچنین از کلاس DataSet استفاده می کنیم که در فضای نام System.Data قرار دارند و برای تمام سرویس دهنده ها قابل استفاده هستند. بنابراین به کمک دستور using فضای نام های مذکور را به شکل زیر به برنامه اضافه می کنیم.

```
using System.Data;
using System.Data.SqlClient;
```

کلاس SqlConnection



تقریباً می‌توانیم بگوییم که کلاس SqlConnection در قلب کلاس‌هایی قرار دارد که در این قسمت مورد استفاده قرار می‌گیرد، زیرا این کلاس وظیفه برقراری ارتباط بین برنامه و بانک اطلاعاتی برنامه را بر عهده دارد.

یک نمونه از این کلاس دارای جزء داده ای به نام ConnectionString می باشد. این خصوصیت به نام رشته اتصال معروف است که از نوع رشته‌ای می باشد و تمام داده‌های مورد نیاز برای اتصال به یک بانک اطلاعاتی را شامل می‌شود.

بخش‌های مختلف ConnectionString

ساختار متنی که باید برای ConnectionString مورد استفاده قرار گیرد، بستگی به سرویس‌دهنده‌ی اطلاعاتی دارد که استفاده می‌کنیم. بخش‌های مختلف ConnectionString در جدول مقابل توضیح داده شده است. برای ایجاد امنیت در بانک‌های اطلاعاتی ایجاد شده بوسیله‌ی SQL Server، باید هنگام دسترسی به آنها ابتدا هویت استفاده کننده توسط SQL Server را مشخص کرد. بنابراین اگر بخواهیم توسط یک برنامه به داده‌های موجود در یک بانک اطلاعاتی دسترسی داشته باشیم، باید اطلاعات لازم برای این تعیین هویت را همراه با دیگر اطلاعات در متن ConnectionString مشخص کنیم.

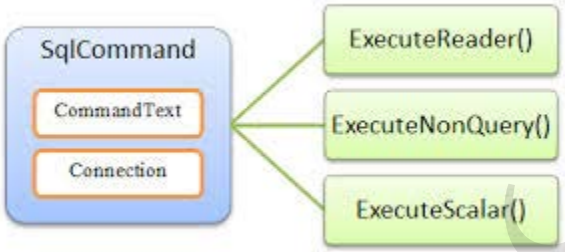
پارامتر	توضیح
Server	نام سرور بانک اطلاعاتی که می‌خواهید از آن استفاده کنید. این پارامتر معمولاً حاوی نام کامپیوتری است که موتور بانک اطلاعاتی SQL Serverدر آن نصب شده است. اگر SQL Server بر روی همان کامپیوتری که برنامه را اجرا می‌کند نصب شده است، می‌توانید از مقادیری مانند localو یا localhostبرای این پارامتر استفاده کنید. اما اگر از SQL Server که در کامپیوتر دیگری در شبکه نصب شده است استفاده می‌کنید، لازم است که مقدار این پارامتر را برابر با نام آن کامپیوتر در شبکه قرار دهید. همچنین اگر در آن کامپیوتر بیش از یک SQL Serverقرار داشته باشد، باید بعد از نام کامپیوتر یک علامت \ قرار داده و سپس نام SQL Server که می‌خواهید مورد استفاده قرار دهید را ذکر کنید.
Database	نام بانک اطلاعاتی که می‌خواهید مورد استفاده قرار دهید در این پارامتر قرار می‌گیرد(برای مثال بانک اطلاعاتی Phonebook).
User ID	این پارامتر باید حاوی نام کاربری باشد که می‌خواهیم برای اتصال به بانک اطلاعاتی از آن استفاده کنیم. برای اینکه بتوانیم با استفاده از این روش از بانک اطلاعاتی استفاده کنیم، باید یک حساب کاربری به این نام در SQL Server ایجاد شده و اجازه‌ی دسترسی به داده‌های مورد نیاز نیز به آن داده شود.
Password	کلمه‌ی عبوری که برای این نام کاربری مورد استفاده قرار می‌گیرد.

مثالی از نحوه ایجاد یک شی اتصال و نحوه استفاده از آن

بعد از اینکه با ایجاد ConnectionString نحوه‌ی برقراری ارتباط با بانک اطلاعاتی را مشخص کردیم، می‌توانیم با استفاده از متدهای Open و Close در کلاس SqlConnection به بانک اطلاعاتی متصل شده و یا اتصال خود را قطع کنیم. یک نمونه از این کار در قطعه کد زیر نشان داده شده است:
البته متدها و خواص فراوان دیگری در کلاس SqlConnection وجود دارند که می‌توانیم در برنامه از آنها استفاده کنیم، اما مواردی که در اینجا با آنها آشنا شدیم پرکاربردترین آنها به شمار می‌روند.

```
SqlConnection conn = new SqlConnection();
    conn.ConnectionString = "رشته اتصال";
    conn.Open();
        .
        .
        .
    conn.Close();
```

کلاس SqlCommand

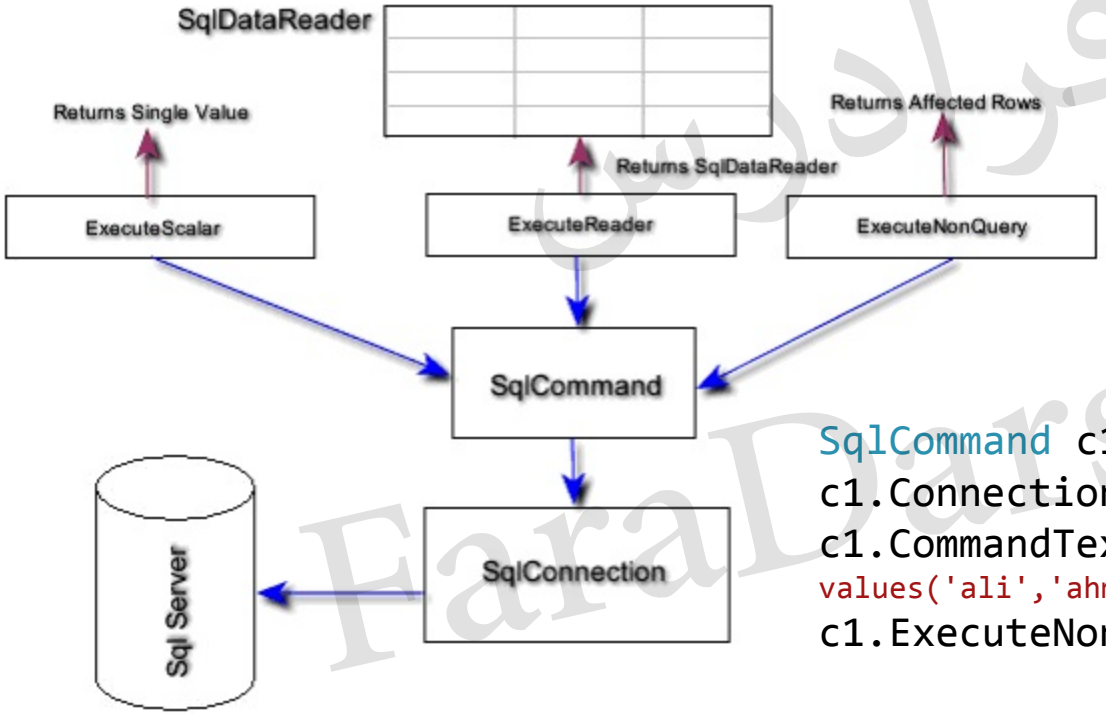


کلاس **SqlCommand** حاوی یک دستور **SQL** برای اعمال تغییرات بر روی داده‌های دریافت شده از بانک اطلاعاتی است. این دستور می‌تواند یک دستور **SELECT** برای انتخاب داده‌هایی خاص، یک دستور **INSERT** برای درج داده‌های جدید در بانک اطلاعاتی، یک دستور **DELETE** برای حذف داده‌ها از بانک اطلاعات و یا حتی فراخوانی یک پروسیجر ذخیره شده در بانک اطلاعاتی باشد. دستور **SQL**ی که در این کلاس نگهداری می‌شود می‌تواند شامل پارامترهایی نیز باشد. به شکل مقابل میتوانیم یک شی از نوع دستور تعریف کنیم.

خاصیت **CommandText** حاوی متنی است که می‌تواند یک دستور **SQL** و یا نام یک پروسیجر ذخیره شده در بانک اطلاعاتی باشد که باید برروی داده‌ها اجرا شود. خاصیت **Connection** مرجع به یک شی اتصال باز می باشد که می خواهیم دستوری در بانک اطلاعاتی مرتبط با آن شی اتصال اجرا گردد.

```
SqlCommand c1 = new SqlCommand();
c1.Connection = conn;
c1.CommandText = "insert into Phones values('ali','ahmadi',0912..','tehran..')";
```

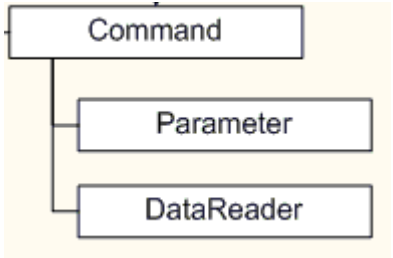
متدهای کلاس SqlCommand



ExecuteNonQuery();
ExecuteReader();
ExecuteScalar();

نمونه ای از برنامه دکمه ذخیره

```
SqlCommand c1 = new SqlCommand();
c1.Connection = conn;
c1.CommandText = "insert into Phones
values('ali','ahmadi',0912..','tehran..')";
c1.ExecuteNonQuery();
```



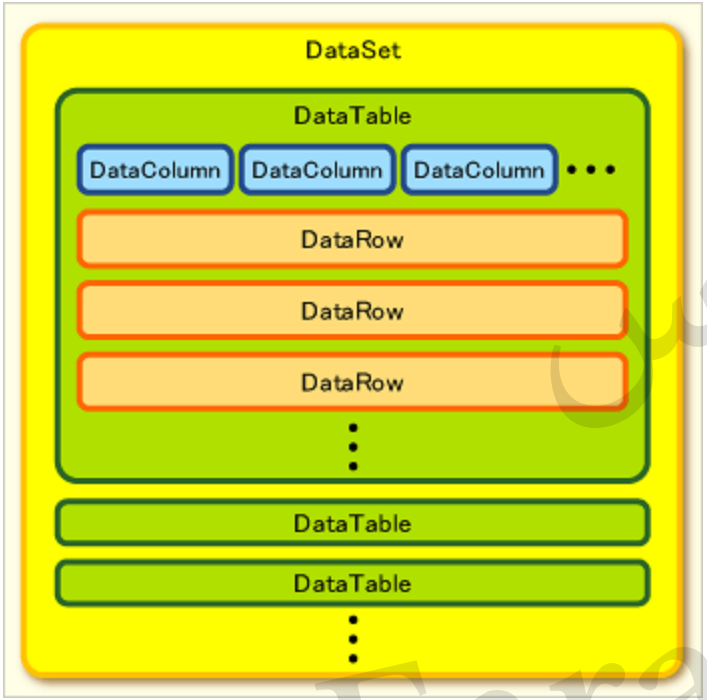
خاصیت Parameters از شی SqlCommand

پارامترها متغیرهایی هستند که در یک دستور SQL قرار می‌گیرند و می‌توانند در زمان اجرای برنامه جای خود را با عباراتی خاص عوض کنند. این متغیرها با علامت @ در یک دستور مشخص می‌شوند.

```
SqlCommand c1 = new SqlCommand();
c1.CommandText = "insert into Phones values(@p1,@p2,@p3,@p4)";
c1.Connection = conn;
c1.CommandType = CommandType.Text;
c1.Parameters.AddWithValue("p1", txtname.Text);
c1.Parameters.AddWithValue("p2",txtfamily.Text);
c1.Parameters.AddWithValue("p3", txttell.Text);
c1.Parameters.AddWithValue("p4",txtaddress.Text);
c1.ExecuteNonQuery();
```

متد **AddWithValue** نام یک پارامتر و متغیری که مقدار مربوط به آن را در زمان اجرای برنامه نگهداری می‌کند به عنوان پارامتر دریافت کرده و آن را به لیست **Parameters** اضافه می‌کند.

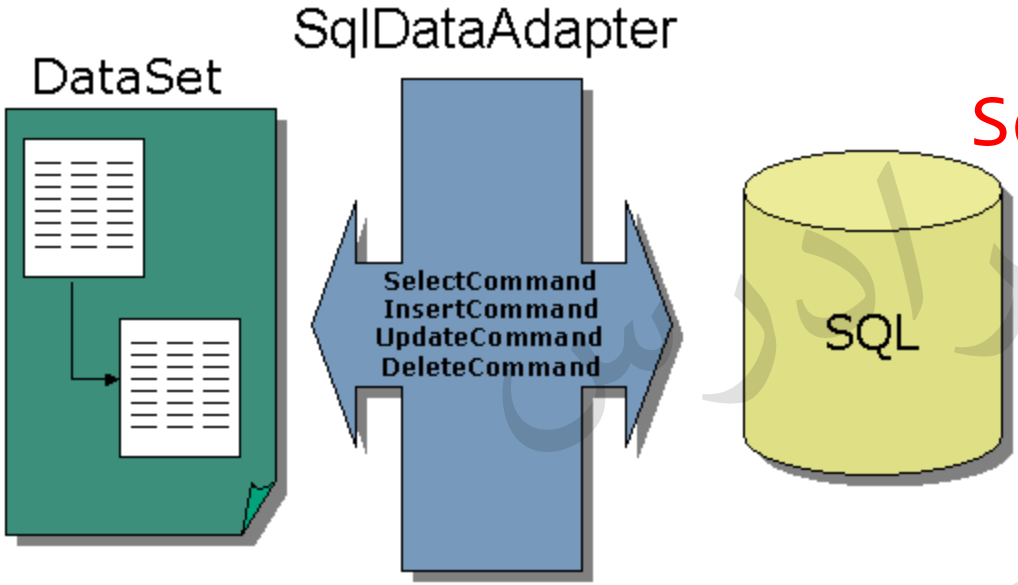
کلاس DataSet



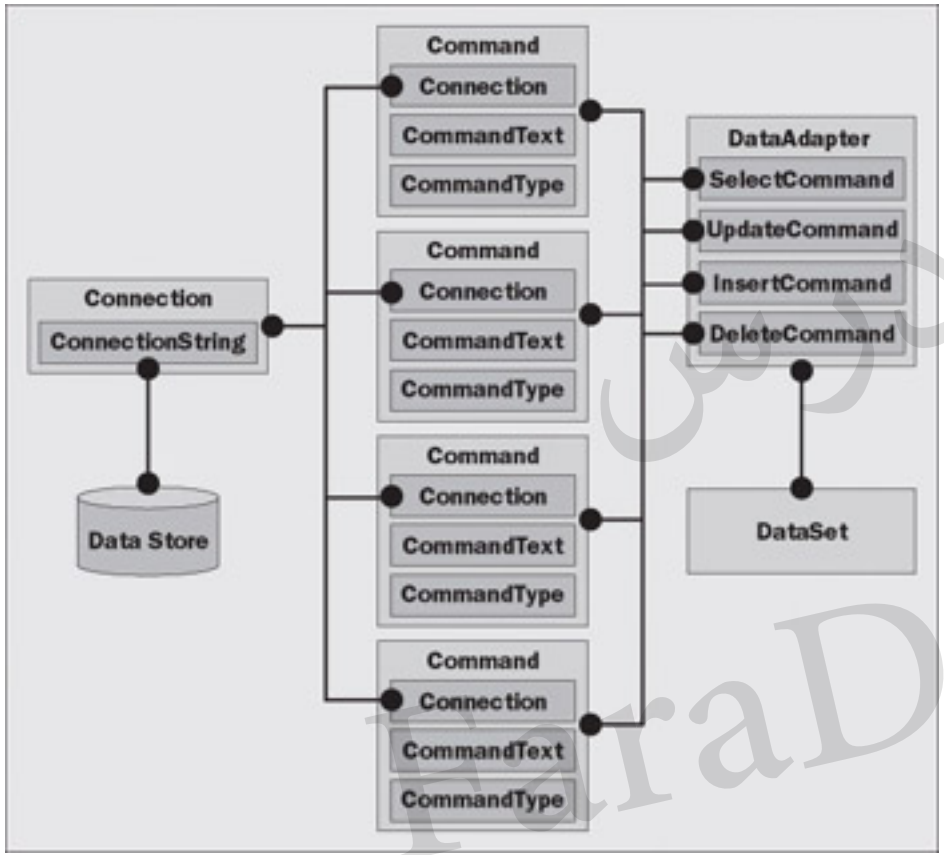
همانطور که مشخص است یک شی DataSet می‌تواند شامل چندین جدول باشد که هر یک از آنها به وسیله‌ی یک کنترل DataTable مشخص می‌شوند. دقت کنید که ابتدای کلاس DataSet کلمه‌ی Sql وجود ندارد. دلیل این مورد هم این است که این کلاس متعلق به فضای نام System.Data.SqlClient نیست بلکه در فضای نام System.Data قرار دارد. به عبارت دیگر، کلاس Dataset به سرویس‌دهنده‌ی اطلاعاتی خاصی از قبیل SqlClient یا OleDb تعلق ندارد و وظیفه‌ی آن نگهداری اطلاعات به دست آمده به هر نحوی در حافظه است. هنگامی که اطلاعات را در حافظه قرار دادیم دیگر نیازی نیست بدانیم که این اطلاعات از کجا به دست آمده‌اند.

```
DataSet ds = new DataSet();
```

کلاس SqlDataAdapter

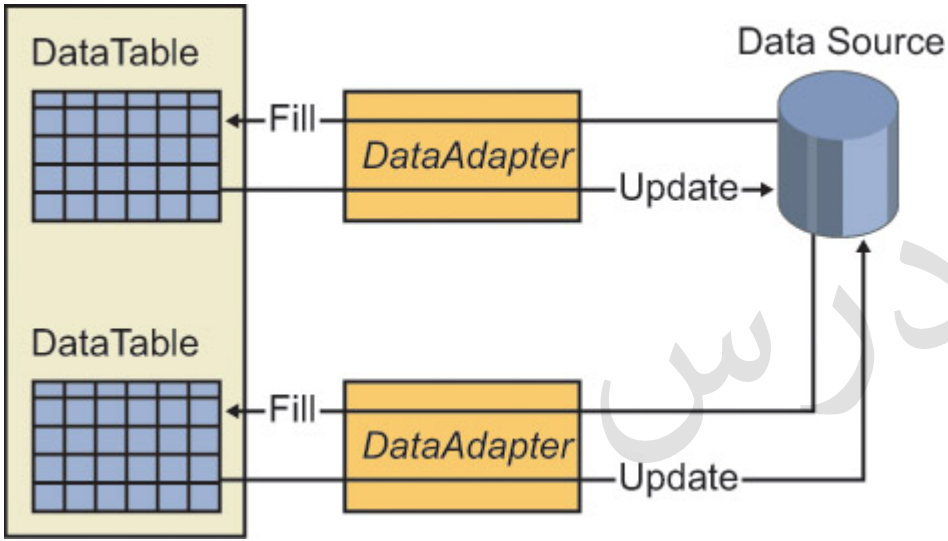


کلاس DataAdapter در برنامه‌های بانک اطلاعاتی، همانند پلی بین جداول بانک اطلاعاتی و نیز داده‌های موجود در حافظه که به وسیله‌ی DataSet نگهداری می‌شوند، عمل می‌کند. می‌توانیم بگوئیم که کلاس DataAdapter برای دسترسی به بانک اطلاعـاتی از کـلاس SqlCommand و SqlConnection استفاده می‌کند.



کلاس **DataAdapter** دارای خاصیتی به نام **SelectCommand** است که مرجع به شیی از نوع دستور می باشد که **DataAdapter** آنرا برروی بانک اطلاعاتی اجرا کرده و نتایج آن را در کلاس‌هایی مانند **DataSet** و یا **DataTable** قرار می‌دهد تا در برنامه‌ها مورد استفاده قرار گیرند. کلاس **DataAdapter** دارای خاصیت‌هایی به نام **InsertCommand**، **DeleteCommand**و **UpdateCommand** است که هر یک شیئی از نوع **SqlCommand** را قبول می‌کنند و **DataAdapter** از دستور ذخیره شده در هر یک از آنها به ترتیب برای حذف، درج و یا ویرایش داده‌ها در بانک اطلاعاتی استفاده می‌کند. در حقیقت هنگامی که ما در طی برنامه تغییراتی را در درون داده‌های موجود در حافظه نگهداری می‌کنیم، **DataAdapter** با استفاده از دستورات موجود در این خاصیت‌ها تغییرات ما را از داده‌های حافظه به داده‌های موجود در بانک اطلاعاتی منتقل می‌کند.

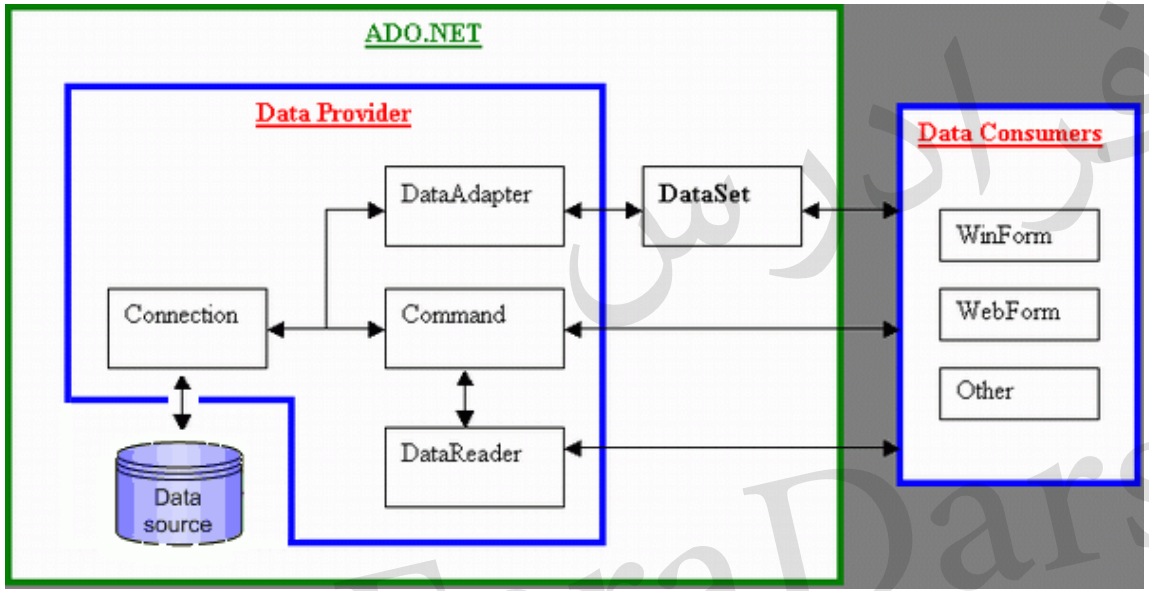
متد Fill از کلاس DataAdapter



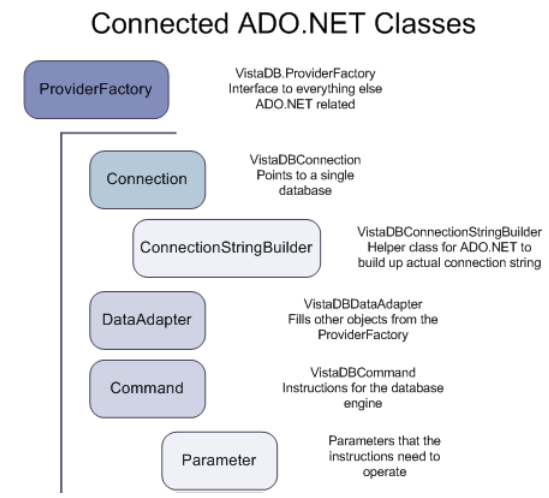

```
SqlConnection conn = new SqlConnection();
conn.ConnectionString = "رشته اتصال" ;
SqlDataAdapter da = new SqlDataAdapter();
DataSet ds = new DataSet();
SqlCommand c1 = new SqlCommand();
c1.Connection = conn;
c1.CommandText = "Select * from Telbook";
da.SelectCommand = c1;
da.Fill(ds, "T1");
```

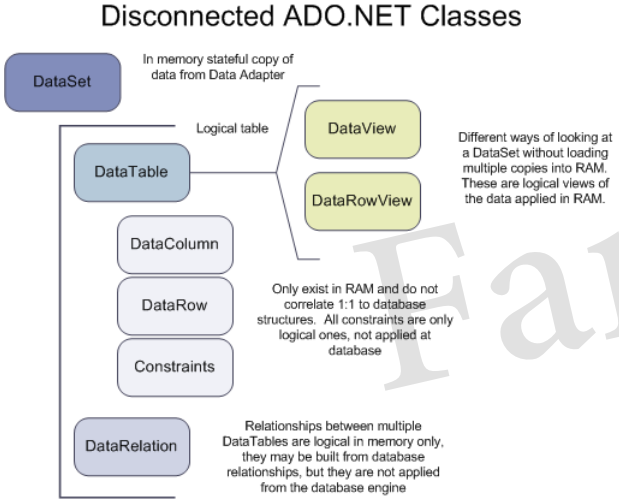
با استفاده از متد **Fill** از کلاس **DataAdapter** می‌توان دستور **SQL** موجود در خاصیت **SelectCommand** را در بانک اطلاعاتی اجرا کرده و سپس داده‌های برگشتی از اجرای این دستور را درون یک **DataSet** در حافظه قرار دهیم. هنگامی که با استفاده از **DataAdapter** داده‌هایی را درون یک **DataSet** قرار می‌دهید، ابتدا یک شئ جدید از نوع **DataTable** ایجاد شده، داده‌ها را درون آن قرار داده میشوند و به **DataSet** اضافه میشوند.

تصور ساده‌ای از کلاس‌های ADO.NET



کلاس‌های موجود در ADO.NET به حدی زیاد هستند که نمی‌توانیم تمام آنها را در این بخش معرفی کنیم. با این وجود تمام کلاس های مهم که برای مثال کاربردی در ادامه نیاز داریم در این بخش معرفی شده اند و ارتباط آنها را در شکل مقابل مشاهده می کنید. .





استفاده ADO.NET از معماری غیرمتصل

ADO.NET برای دسترسی به داده‌ها از معماری غیرمتصل استفاده می‌کند. غیرمتصل به این معنی است که ابتدا برنامه به موتور بانک اطلاعاتی مورد نظر متصل شده، داده‌های مورد نیاز خود را از بانک اطلاعاتی دریافت کرده و آنها را در حافظه کامپیوتر ذخیره می‌کند. سپس اتصال برنامه از بانک اطلاعاتی قطع می‌شود و تغییرات مورد نظر خود را در داده‌های موجود در حافظه انجام می‌دهد. هر زمان که لازم باشد تغییرات ایجاد شده در بانک اطلاعاتی ذخیره شوند، برنامه یک اتصال جدید را با بانک اطلاعاتی ایجاد کرده و از طریق این اتصال تغییراتی که در داده‌ها اعمال کرده بود را در جدول اصلی پیاده‌سازی می‌کند. شیی اصلی که داده های دریافتی از بانک اطلاعاتی را در حافظه نگهداری می‌کند، شی DataSet است.

استفاده از کلاس‌های ADO.NET در عمل

تا این قسمت از کار تنها بصورت تئوری کلاس های موجود در ADO.NET را معرفی کردیم. در ادامه با طراحی و پیاده‌سازی یک دفترچه تلفن مطمئن می‌شویم که نحوه استفاده از این کلاس‌ها، متدها، خاصیت‌هـاو... را درست درک کرده‌ایم.