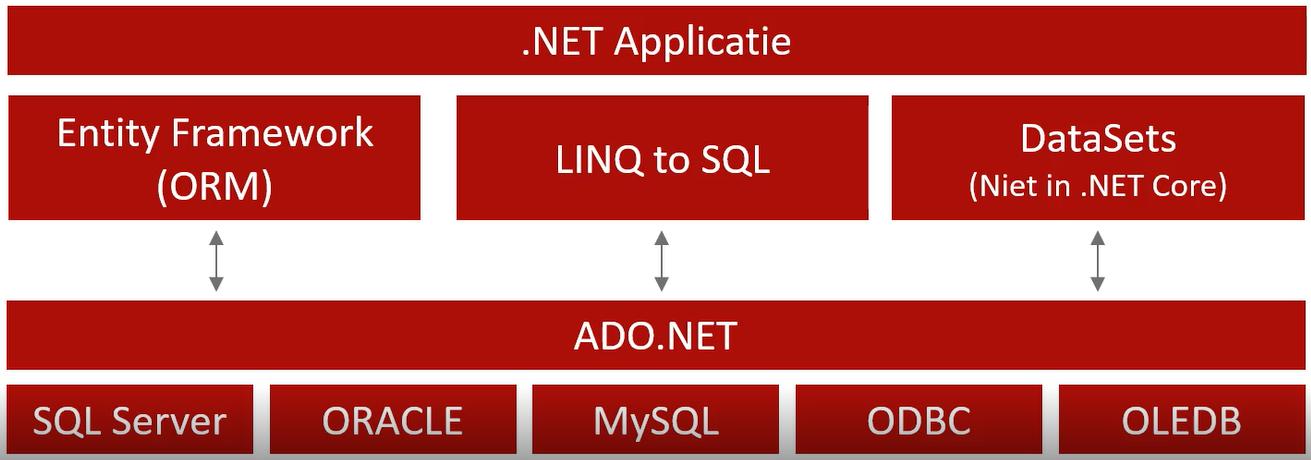
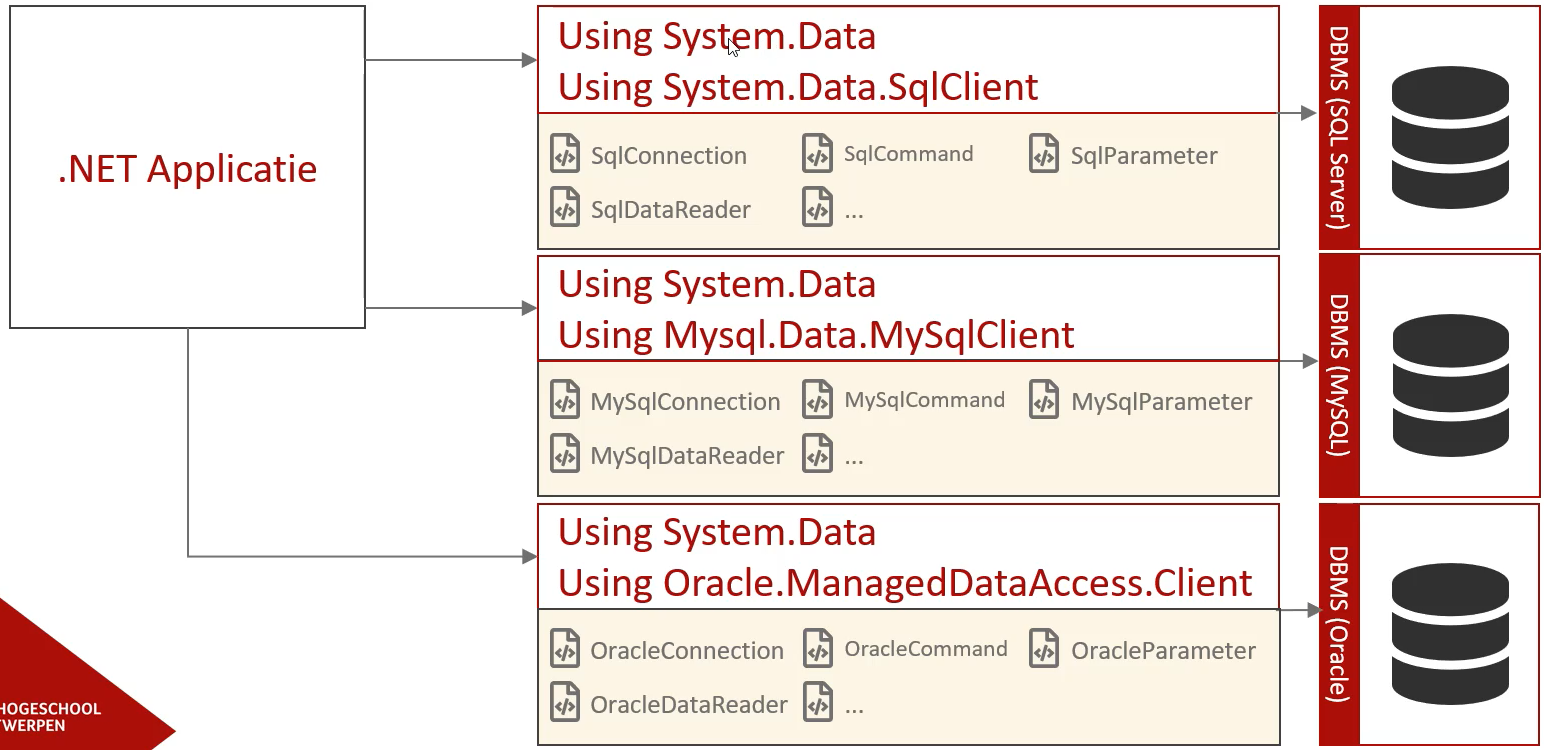
**Deel 9 - ADO.NET introductie**

**Architectuur**

* Subset van het .NET ecosysteem
* Voorziet functionaliteit om toegang tot data te krijgen



* Database provider
* Meerdere fabrikanten van databases
* Elke fabrikant voorziet eigen bibliotheek die interfaces implementeert



**Connectionstring**

* Bevat alle logica om connectie te kunnen maken met een SQL Server Database
* Instellingen van connectie via Connectionstring property
* Connectionstring opgebouwd uit case insensitive keywords en values, elk paar gescheiden door een ‘;’

Default Poort SQL Server = Poort 1433

* Waar Connectionstring bewaren?
  + Centrale plaats kiezen

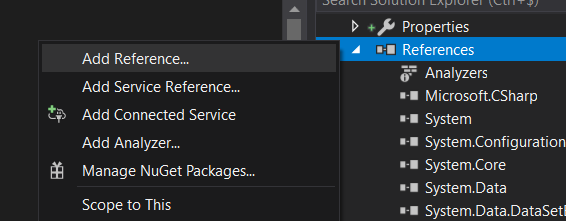
=> Slechts op 1 plaats aanpassen indien nodig

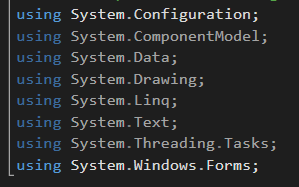
=> Niet in code want elke wijziging vereist hercompilatie

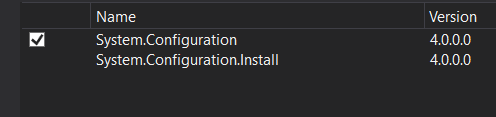
* + Centrale plaats is .config file
    - Web.config voor web-based applicaties
    - App.config voor desktop applicaties
    - (appsettings.json(.NET Core))
* Hoe connectionstring bewaren
  + <connectionStrings> sectie toevoegen
  + In deze sectie
    - <clear/> element als eerste toevoegen indien je geen COnnectionstrings wil overnemen uit overkoepelende .config files
    - Per Connectionstring een element toevoegen met de tag <add
    - Twee properties in dit element instellen: “name” en “connectionString”



* Hoe Connectionstring ophalen
  + Reference toevoegen in project naar System.Configuration
  + Using statement toevoegen voor System.Configuration
  + Gebruik ConnectionStrings property van ConfigurationManager object







* Connectionstring kan gevoelige informatie bevatten:
  + Poortnummers
  + Credentials (in geval van SQL Auth)
* Hoe connectionstring op een veilige manier bewaren
  + In geval bewaard in .config file is alles leesbaar indien file access
  + Connectionstring in code? ILDASM (Tool) kan dit uit .exe of .dll bestand lezen
  + Oplossing is **encryptie**!

=> Zelf uitwerken voor desktop applicaties, web applicaties hebben bepaalde libraries die dat voor jou doen

**SqlConnection object**

**Werken met**

* Te gebruiken om verbinding te maken met database
* Connectie moet geopend en gesloten worden
* Openen via **.Open()** method
* Sluiten via **.Close()** method
* Correcte afhandeling van vrijgeven connectie met behulp van het **using** statement
* Statement voor correcte vrijgave van resource gespecificeerd in het statement
* Geen nood meer aan **.Close()** of **.Dispose()** call

**Connectionpooling**

* **Connection pool** is een “vijver” met beschikbare connectie objecten
* Elke **.Open()** call zal in deze vijver gaan vissen
* Elke **.CLose()** call zal de connectie terugzetten in deze vijver ipv effectief de verbinding te sluiten
* Pooling staat default aan maar kan afgezet worden via de Connectionstring *(Pooling keyword)*
* Maximum en minimum aantal connecties van de pool ook instelbaar

*(Max Pool Size/Min Pool Size keywords)*

**SqlCommand object**

* Representeert een SQL statement of Stored Procedure call naar een SQL Server db
* Instellen van volgende properties:
  + **CommandText**

=> Bevat de uit te voeren query of tabelnaam of naam Stored Procedur

* + **CommandType**

=> Text, TableDirect of StoredProcedure

* + **Connection**

=> SqlConnection instance

**Data ophalen**

* Eén enkele (scalaire) waarde ophalen
  + Gebruik de **.ExecuteScalar()** method
* Een resultaatset (1 of meerdere records) ophalen
  + Gebruik **.ExecuteReader()** method
  + Geeft een SqlDataReader object terug
  + SqlDataReader = Forward only stream van records
  + Itereren over records
  + Toegang tot kolommen van een record via index of kolomnaam
* Een DML instructie uitvoeren (bv. insert)
  + Gebruik **.ExecuteNonQuery()** method
  + Geeft geen resultaat terug

**Inline SQL vs. Stored Procedure**

* Inline SQL
  + SQL statements in code verwerkt
  + **CommandType** property instellen op **CommandType.Text**
  + Query toewijzen aan **CommandText** property
* Stored Procedure
  + SP gebruiken ipv letterlijk SQL instructies
  + **CommandType** property instellen op **CommandType.StoredProcedure**
  + Naam van stored procedure toewijzen aan **CommandText** property

**SqlParameter object**

* Dynamisch maken van SQL instructies (bv. filteren)
* Niet doen dmv string concatenatie (SQL injection!)
* Parameters gebruiken in SQL instructies
  + Inline SQL
  + Stored procedures
* SqlParameter object gebruiken aan .NET kant
* SqlParameter object voorzien per gebruikte parameter in SQL instructie
* SqlParameter object correct configureren
* Toewijzen aan **Parameters** property van **SqlCommand** object
* Configuratie SqlParameter object:
  + Direction
    - Input => Om waarde naar database te sturen
    - Output => Om waarde van database terug te krijgen
    - InputOutput => Kan in 2 richtingen terug te krijgen
    - ReturnValue => Komt default terug van een stored procedure/function
  + SqlDbType
    - DataType van de parameter
    - Indien gebruikt voor stored procedure moeten datatypes matchen
  + Value
    - De waarde die meegegeven wordt aan de parameter
    - Lege value <> Null value in database => **DBNull.Value** gebruiken