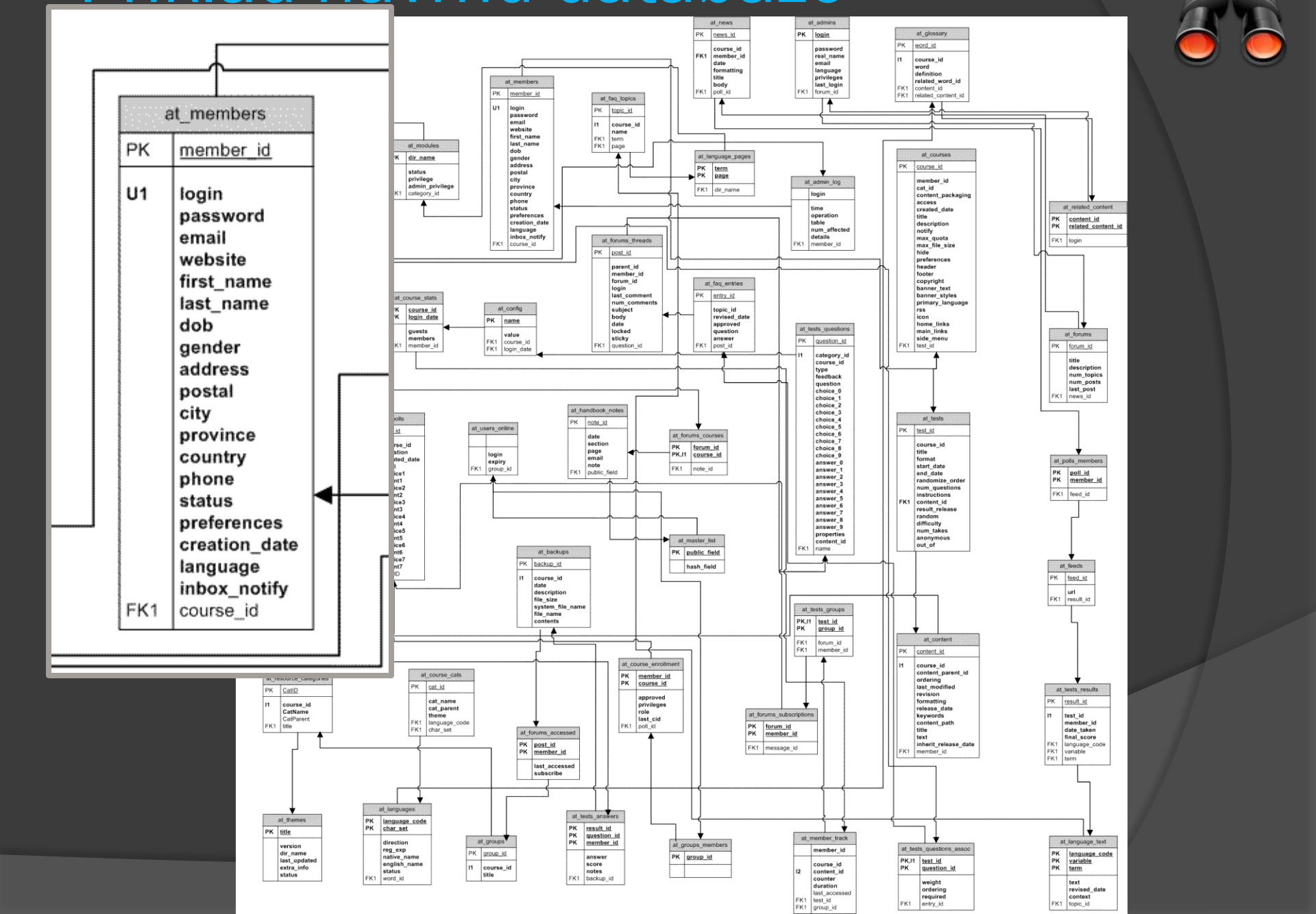
C# - Databáze úvod, ADO.NET

Centrum pro virtuální a moderní metody a formy vzdělávání na Obchodní akademii T.G. Masaryka, Kostelec nad Orlicí

Co je to databáze ?

- Databáze je určitá uspořádaná množina informací uložená na datovém médiu
- Data jsou v databázi uložena v tabulkách, uložená data mají většinou pevně stanovena kritéria, kterým musí vyhovovat (např. Jméno osoby je typu text o maximální délce ....)
- Jednotlivé tabulky jsou propojeny do logické struktury a návrh databáze je klíčovou součástí korektně vytvořené aplikace a musí splňovat přísná kritéria
- Součástí databáze nemusí být jen konkrétní data uložena v tabulkách ale i množina funcí a procedur, které je možné nad těmito daty definovat.
- K manipulaci s jednotlivými databázemi existují buď SW nástroje (Access) nebo v případě větších databází DB servery, které umožňují pokročilou správu spojenou s provozem DB
- K získávání výsledků dotazů na DB slouží univerzální jazyk SQL

Příklad návrhu databáze



K zamyšlení

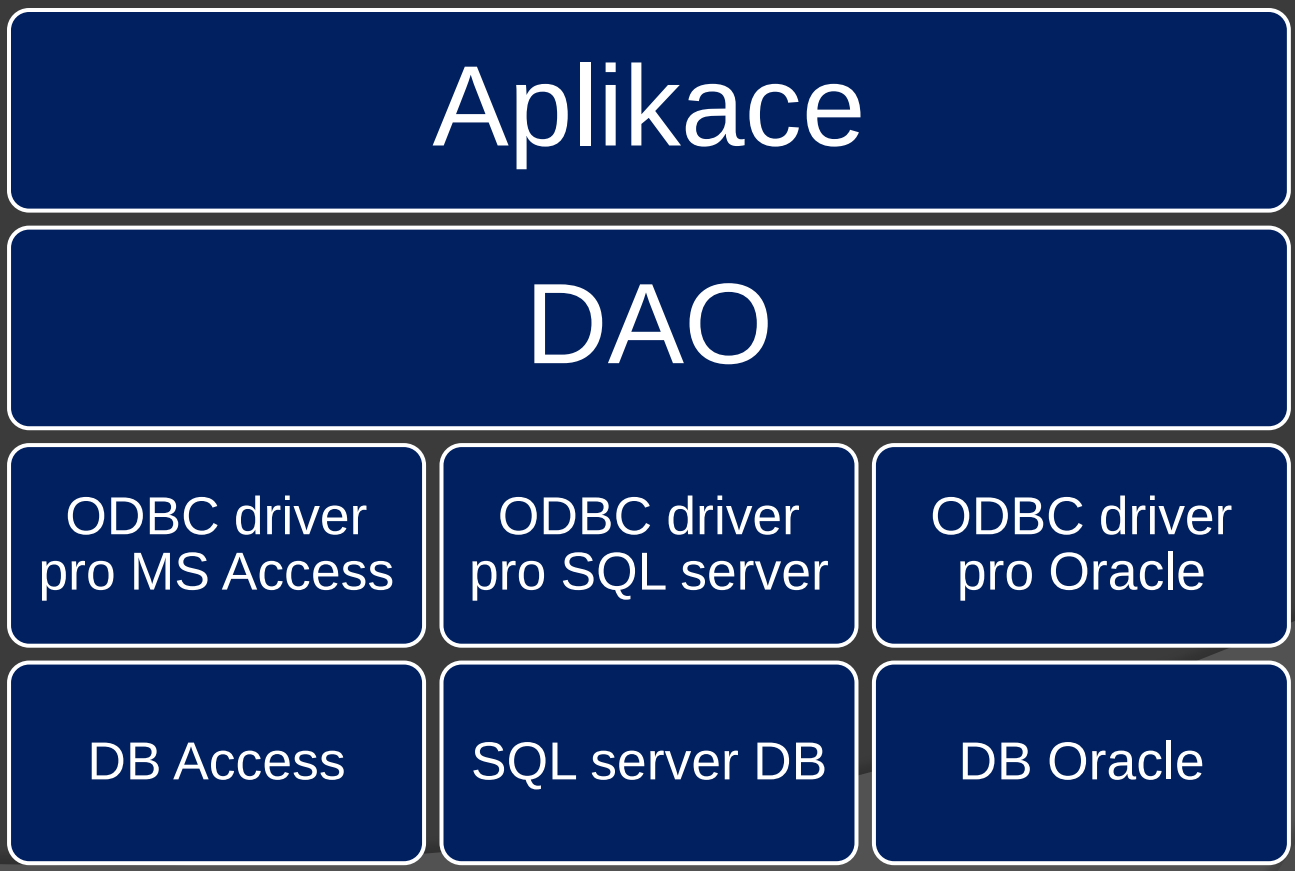
- Zamyslete se například nad tím jak by mělo být v DB reprezentováno PSČ (jaký datový typ a délku by jste zvolili)
- Zamyslete se nad tím co může provádět procedura na DB serveru a jestli je výhodnější použít pro manipulaci s daty proceduru, která je vykonávána na DB serveru, nebo proceduru, která je na úrovni psané aplikace (např. V C#) – srovnejte obě možnosti

Řešení

- Uvědomte si, že ne všude je PSČ reprezentováno číslem, je běžné v jiných státech, že PSČ může být např. BR58SE – pokud by jste reprezentovali a napsali aplikaci s DB, která bere pouze čísla, byla by při rozšíření na zahraniční adresy nepoužitelná a nutná přeprogramovat, s čímž souvisí nejen přepracování DB (záloha dat,...), ale i přepracování aplikační vrstvy
- Vezměte si například že na základě nějakých kritérií potřebujete zjistit kdo z DB 1000000 lidí daným kritériím vyhovuje a tyto lidi vypsat – pokud daný výběr provedete na DB serveru může po síti běžet pouze výsledek, narozdíl od tisíce záznamů, které jsou následně zpracovávány na klientu.

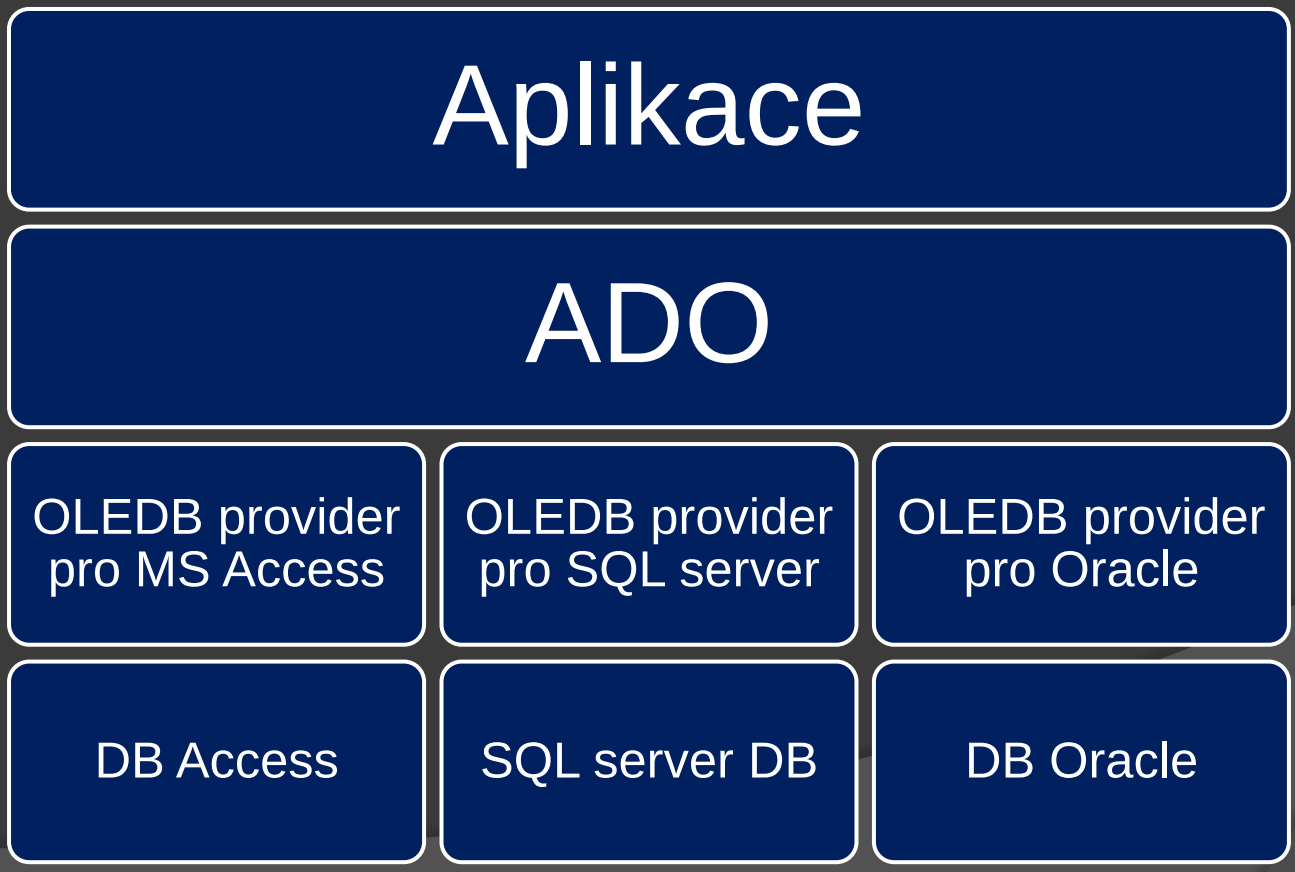
Trocha historie DAO

- Data access object
- Přístup přes ODBC driver



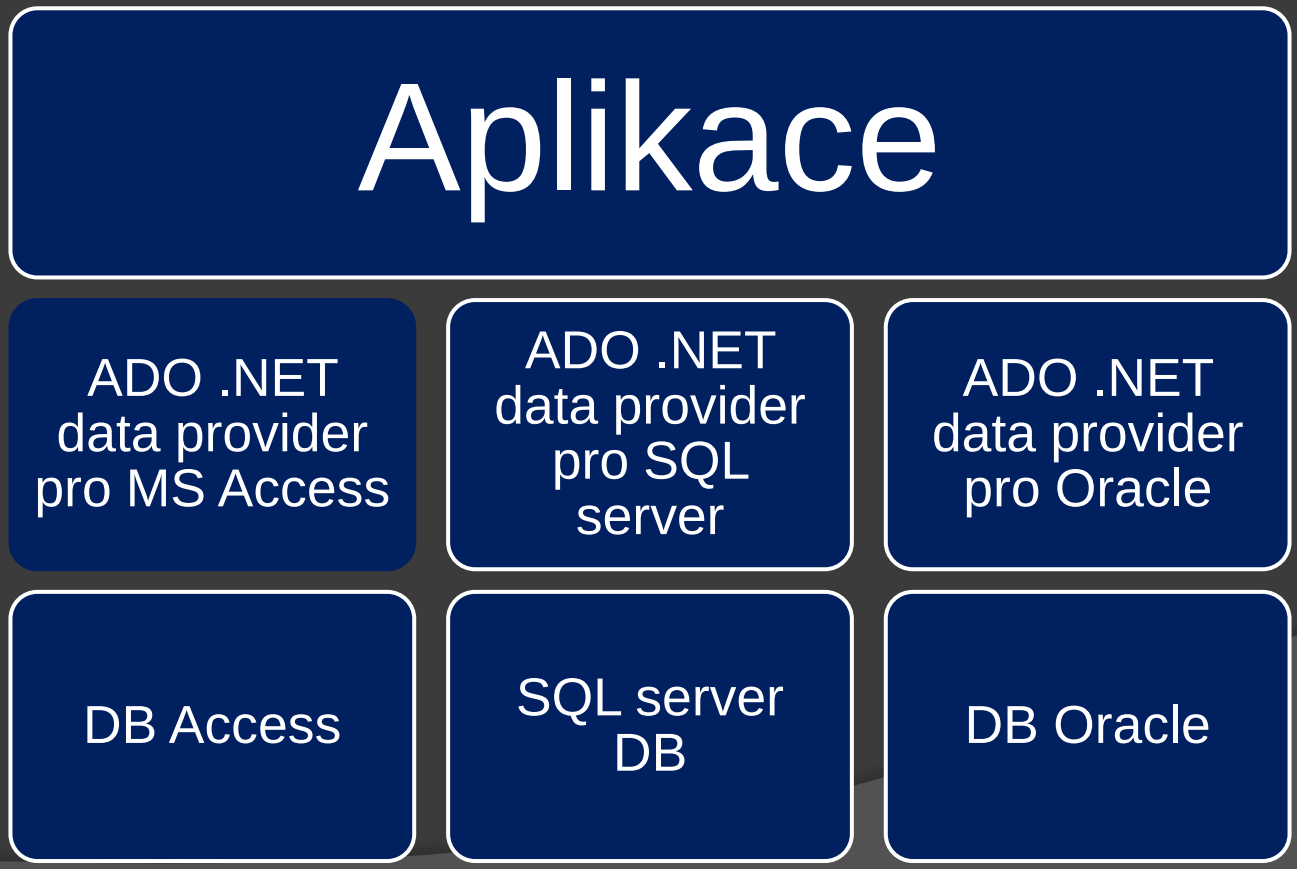
Trocha historie ADO

- Access data object
- Přístup přes OLE DB providery
- Využívá technologii ActiveX a COM (možnost linkování různých knihoven napsaných v různých programovacích nástrojích



ADO .NET

- Úplně přepracované
- Rychlejší než předchozí
- Efektivnější objekty (konec nabubřelého Recordsetu)



Co je to ADO.NET ?

- Existuje mnoho aplikací, které ke své činnosti potřebují data z různých datových zdrojů (databází)
- K přístupu k těmto datům z aplikací tvořených na platformě Windows slouží komponenta ***ADO.NET***
- Komponenta ***ADO*** existuje už několik let ale její kompletní předělání na platformu .NET přineslo několik výhod a rozšíření ve způsobu jejího použití pro přístup k datům
- ADO je souborem tříd, rozhraní, struktur a enumerátorů, které umožňují přístup a práci s daty z prostředí .NET

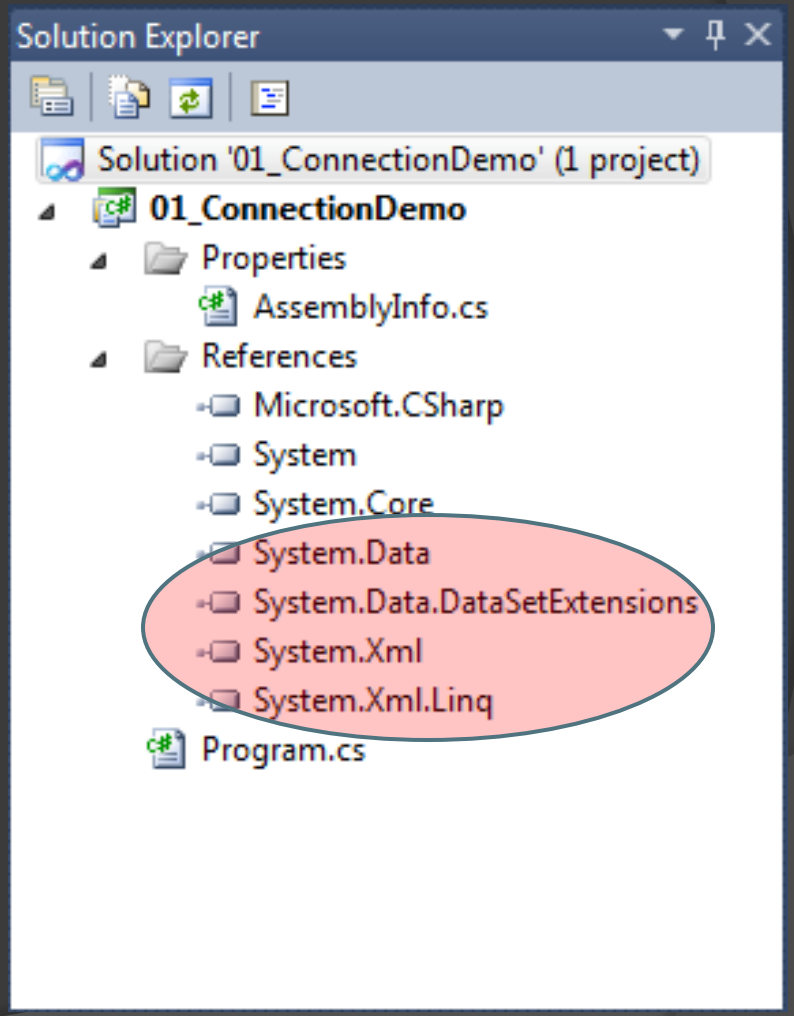
Podporované datové zdroje

- ***Relační DB***
  - Oracle, MS Access, SQL Server a další
- ***Hierarchické***
  - XML
- ***Strukturovaná nehierarchická data***
  - CSV, soubory MS Excel, soubory Active directory

ADO .NET knihovna

- ADO.Net knihovna je umístěna v **System.Data.dll**
- ***Poznámka:***

  Všimněte si, že tato knihovna integrována s knihovnou **System.Xml.dll**



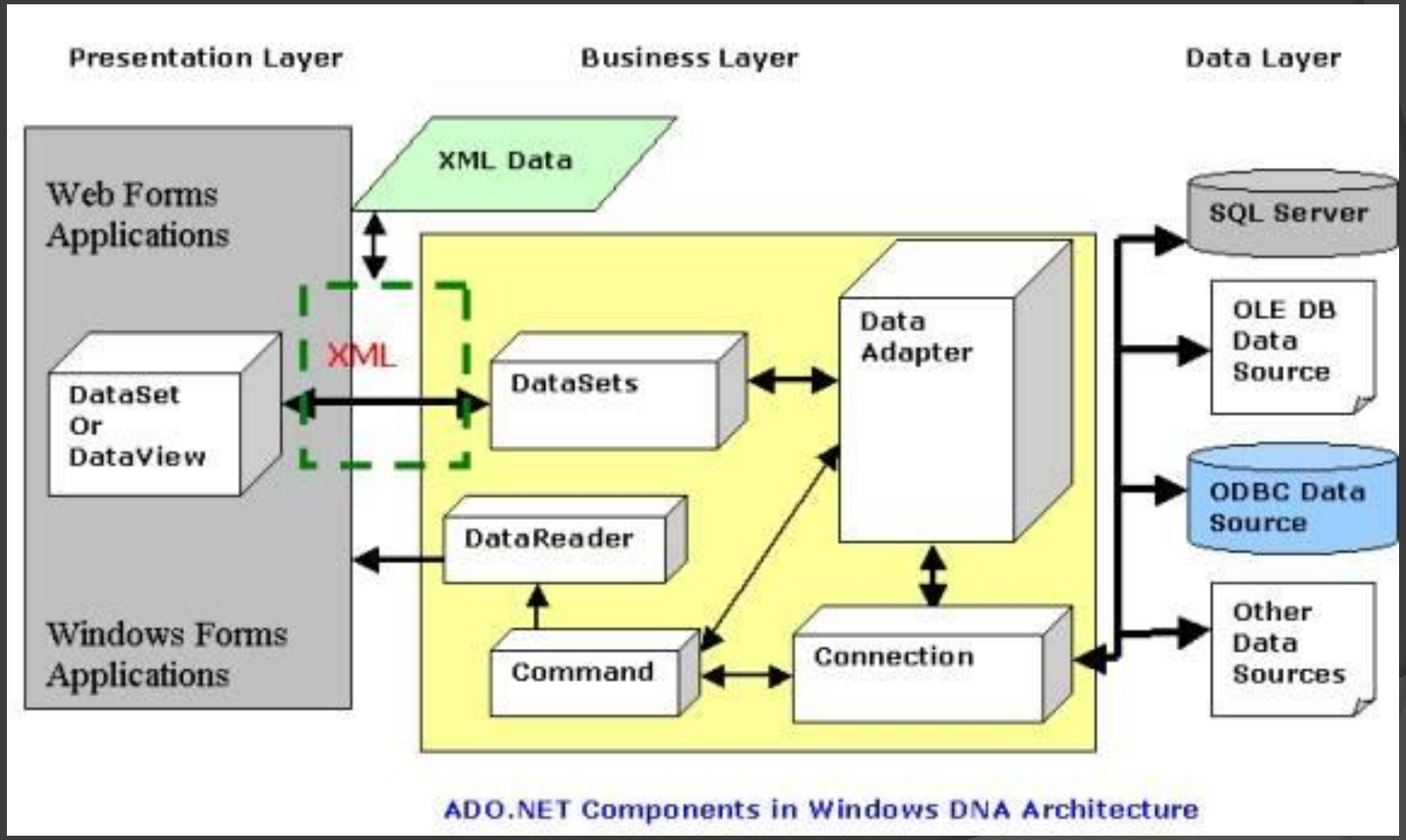
Poskytovatelé dat (data providers)

- Jak již bylo zmíněno pro uložení DB a jejich manipulaci existuje mnoho různých aplikací a technologií
- Tyto různé zdroje mají svá určitá ***rozhraní*** a s tímto rozhraním je schopný ADO.NET komunikovat
- Jednotlivý zprostředkovatelé přístupu k datovým zdrojům se nazývají **ADO.NET data providers**
- Obrovskou výhodou při psaní aplikací je pak to, že jeli naše aplikace napsána v ADO.NET je při přechodu na jiný datový zdroj (SQL server - > Oracle) je nutné změnit providera připojení ale chod aplikace zůstane neměnný
- Několik nejpoužívanějších providerů :
  - **Databáze Microsoft SQL Server** (7.0 nebo vyšší) – jmenný prostor System.Data.SqlClient
  - **ODBC datové zdroje** – jmenný prostor System.Data.Odbc
  - **OLE DB datové zdroje** – jmenný prostor System.Data.OleDb
  - **Databáze Oracle** – jmenný prostor System.Data.Oracle
- Existují i nezabudované providery třetích výrobců

Custom provider

- Každý z providerů poskytuje programátorovi stejné třídy se stejným rozhraním
  - Connection
  - ConnectionStringBuilder
  - Command
  - DataReader
  - Parametr
  - Transaction
- Každý z providerů má svůj vlastní ***namespace*** – nutné ho použít v sekci using:
  - **System.Data.SqlClient**
  - System.Data.OracleClient
  - System.Data.Odbc
  - System.Data.OleDb

Objektový model ADO.NET



Připojené vs. odpojené aplikace

- Základní otázkou při vytváření DB aplikace je zdali půjde o aplikaci ***ONLINE*** – tzn. Permanentně připojenou k DB, nebo ***OFFLINE*** – aplikace bude komunikovat s DB pouze někdy a většinu času pracuje s offline obrazem dat
- Zamyslete se nad výhodami a nevýhodami obou řešení a uvěďte příklady obou typů aplikací z reálného světa

Odpojené aplikace

- Část dat z centrální DB je nakopírována z centrálního úložiště a zpracovávána nezávisle na spojení s DB. Změny jsou pak provedeny po znovupřipojení

+

- Můžete pracovat nezávisle na stálém datovém spojení
- Nezatěžujete trvalým připojením výkon datového zdroje ani připojení
- Zvyšuje se výkon lokální aplikace

-

- Data nejsou vždy aktuální
- Musíte řešit konfliktní situace
- Nehodí se pro velký objem dat

Připojené aplikace

- Stále udržují spojení s datovým úložištěm

+

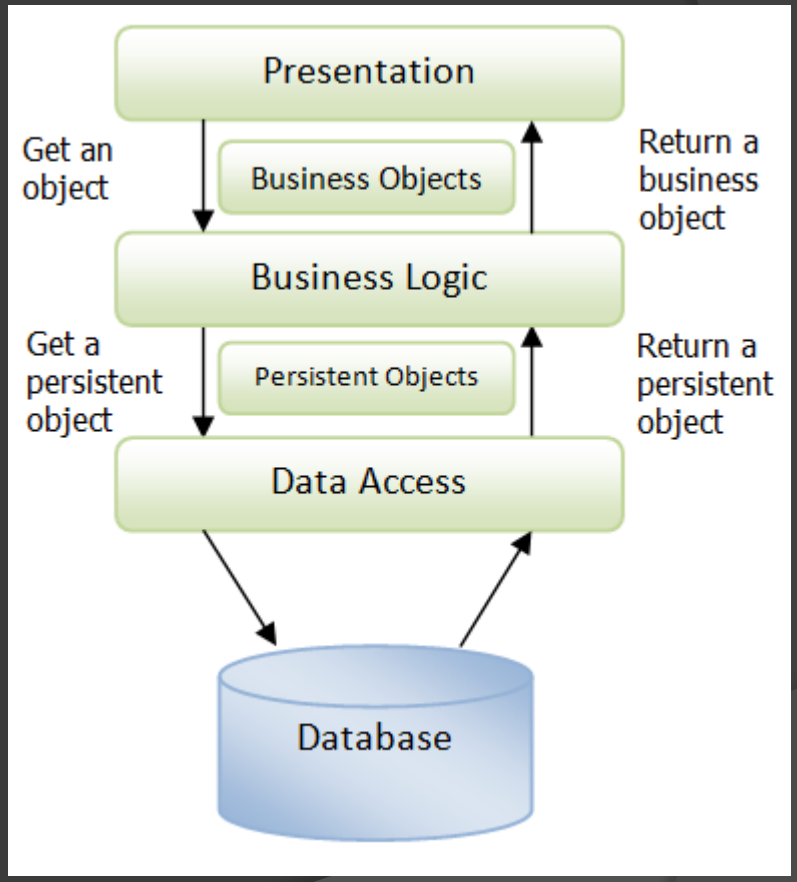
  - Stále aktuální data

-

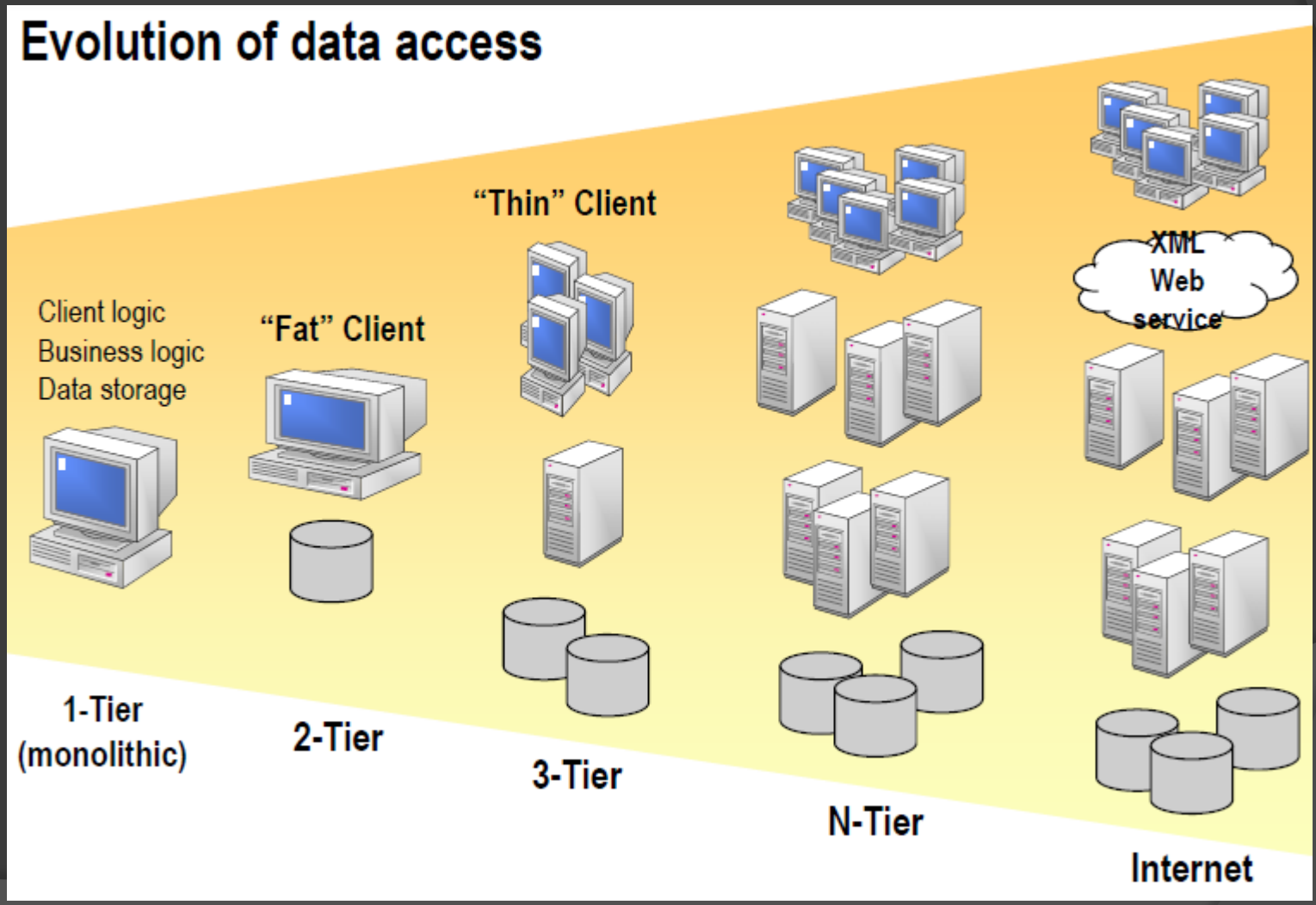
  - Mnoho připojení na jedno datové úložiště
  - Zpomalení celého systému
  - Výpadek spojení = konec aplikace

Vrstvy databázových aplikací

- Při vývoji DB aplikací je vhodné dodržet oddělení databázové, aplikační a prezentační logiky.

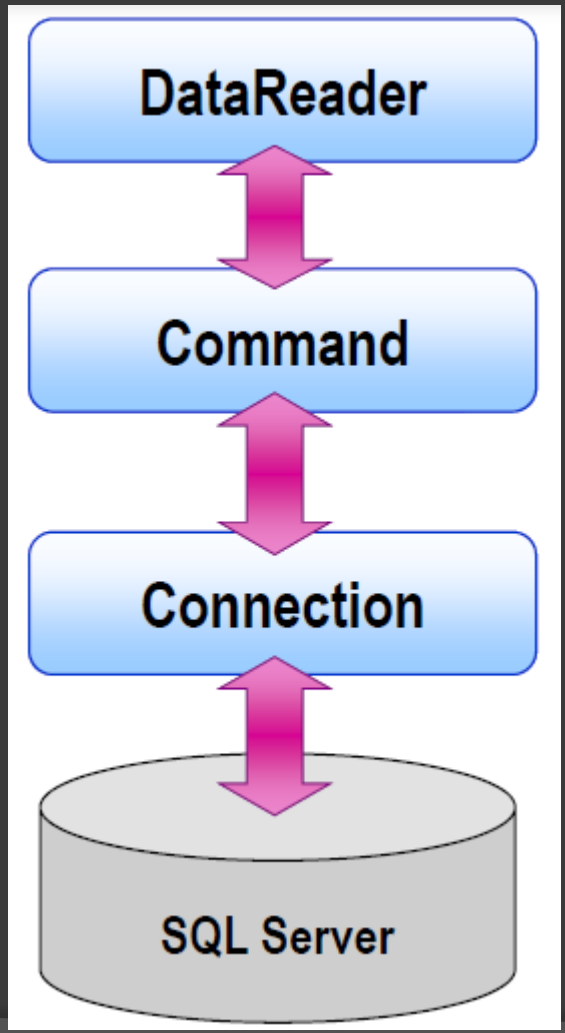


Modely DB aplikací



Scénář připojené aplikace

- A použité objekty

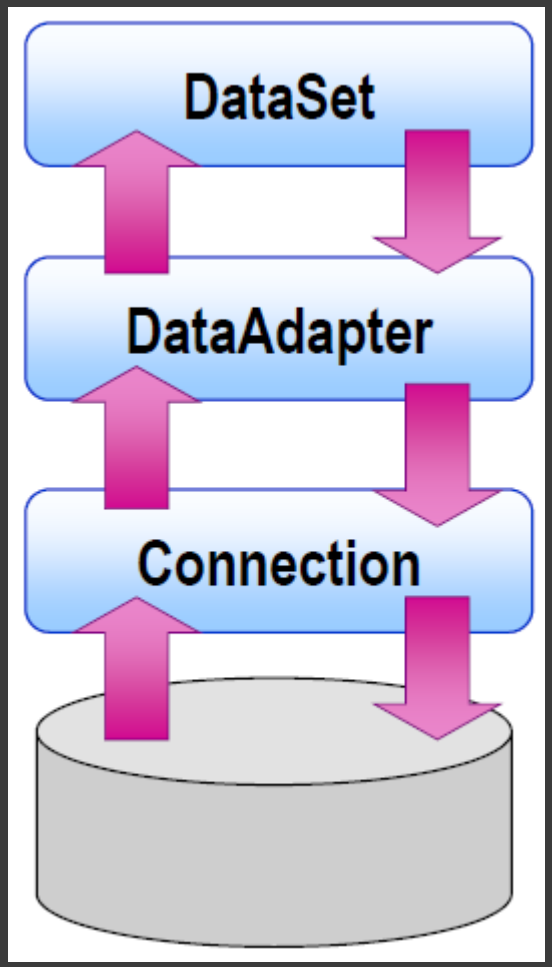


- V připojeném scénáři jsou zdroje dat až do uzavření spojení na DB úložišti (SQL serveru)

1. Otevřu připojení – Open Connection
2. Vykonám příkaz – Execute Command
3. Zpracuji řádky v objektu DataReader
4. Zavřu Reader – Close Reader
5. Zavřu Spojení – Close Connection

Scénář odpojené aplikace

- A použité objekty

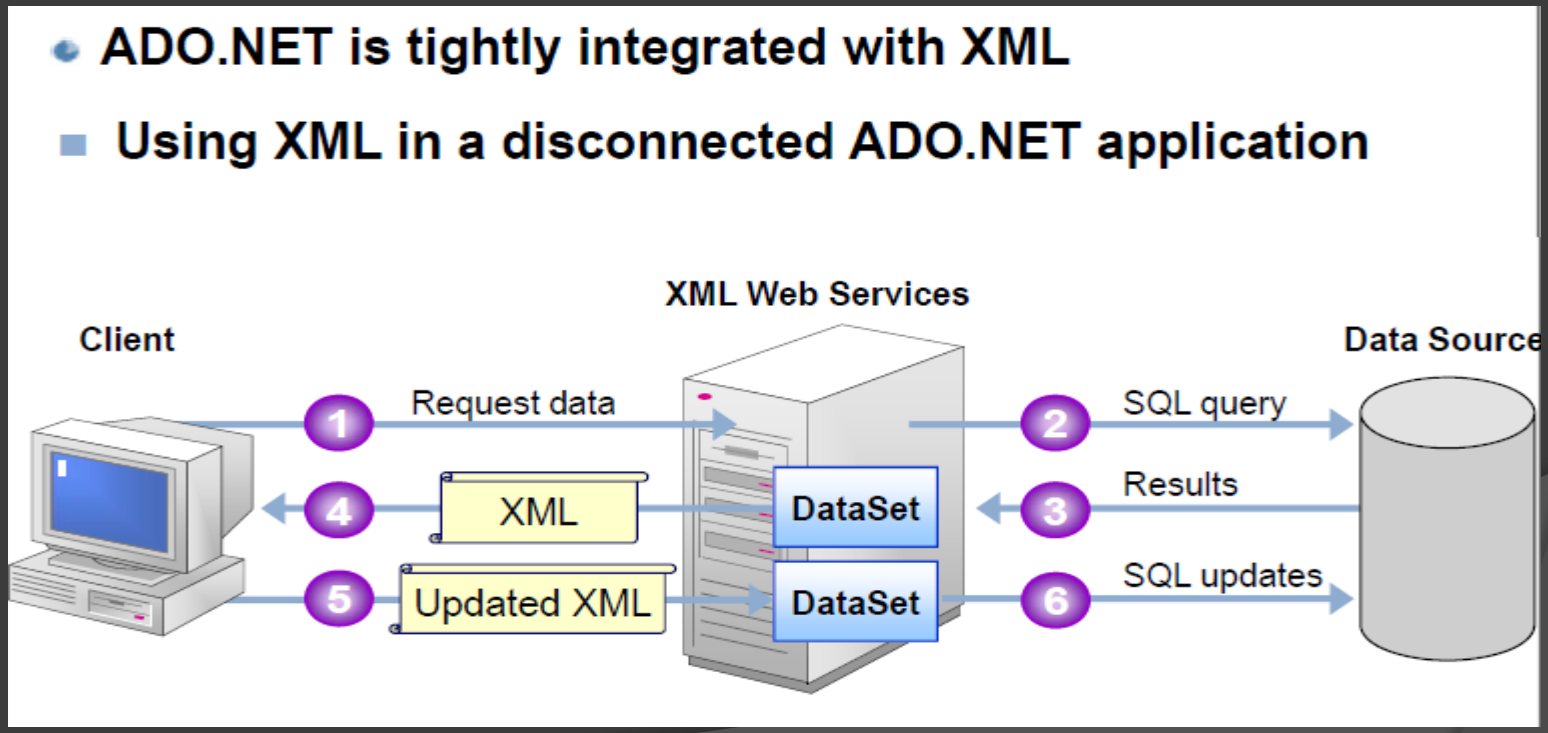


- V odpojeném scénáři nejsou zdroje dat na DB úložišti (SQL serveru) v momentě jejich zpracovávání

1. Otevřu připojení – Open Connection
2. Naplním Dataset – Fill Dataset
3. Zavřu Spojení – Close Connection
4. Otevřu připojení – Open Connection
5. Updatuji datový zdroj (DB)
6. Zavřu Spojení – Close Connection

Další možnosti a objekty ADO.NET

- ***ADO.NET*** nabízí k využití velké množství objektů a jejich metod např. pro práci s ***XML*** dokumenty objekt typu ***DataSet*** umožňující tvorbu „obrazu“ dat v naší aplikaci zejména u aplikací typu ***offline*** atd.



- Následovat budou prezentace věnované připojení k DB, Online aplikacím a offline aplikacím