Databázové systémy I.

Off-line aplikace

OFF-LINE Aplikace

- Aplikace nejsou trvale spojeny s databází
- K synchronizaci dochází na základě příkazu
- Minimalizována komunikace mezi klientem a DB
- Výhodné při umístění DB na jiném stroji
- Problematické při vyšším počtu uživatelů (velký počet kolizí při update)

OFF-LINE Aplikace

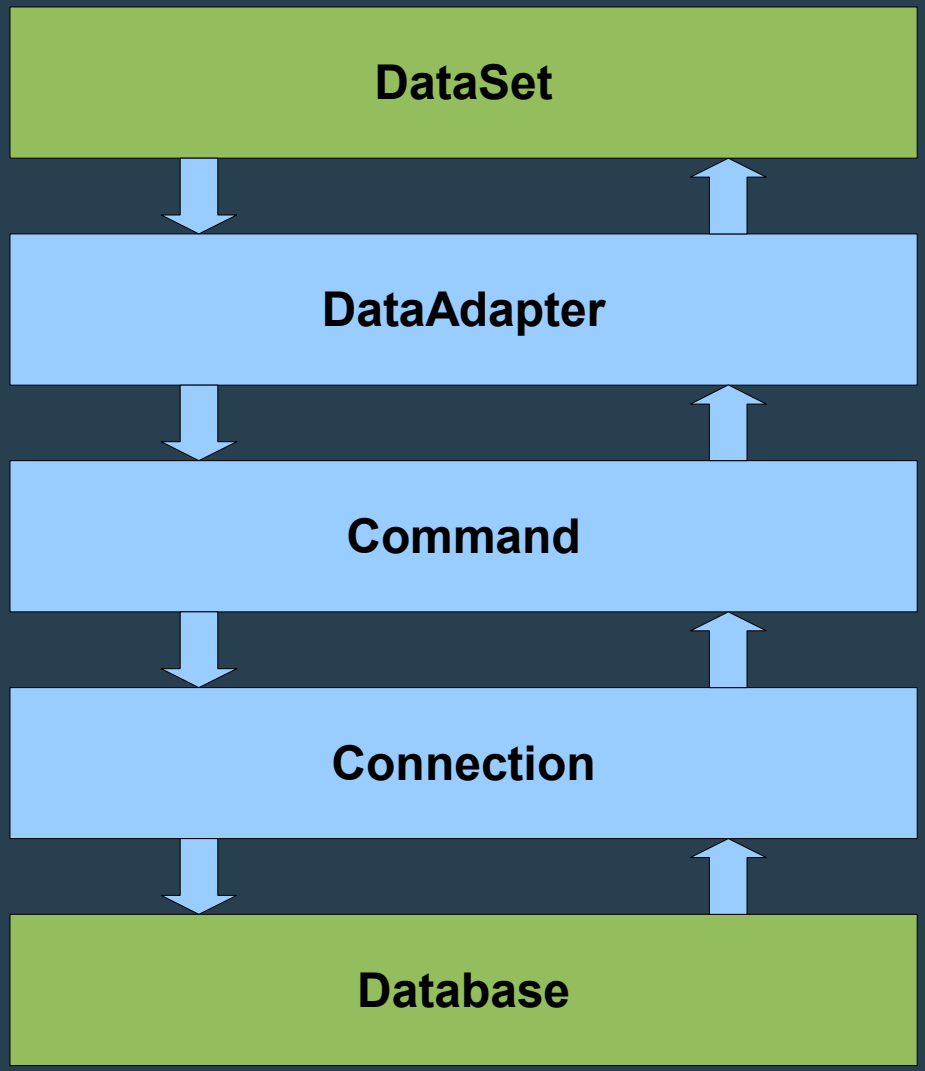
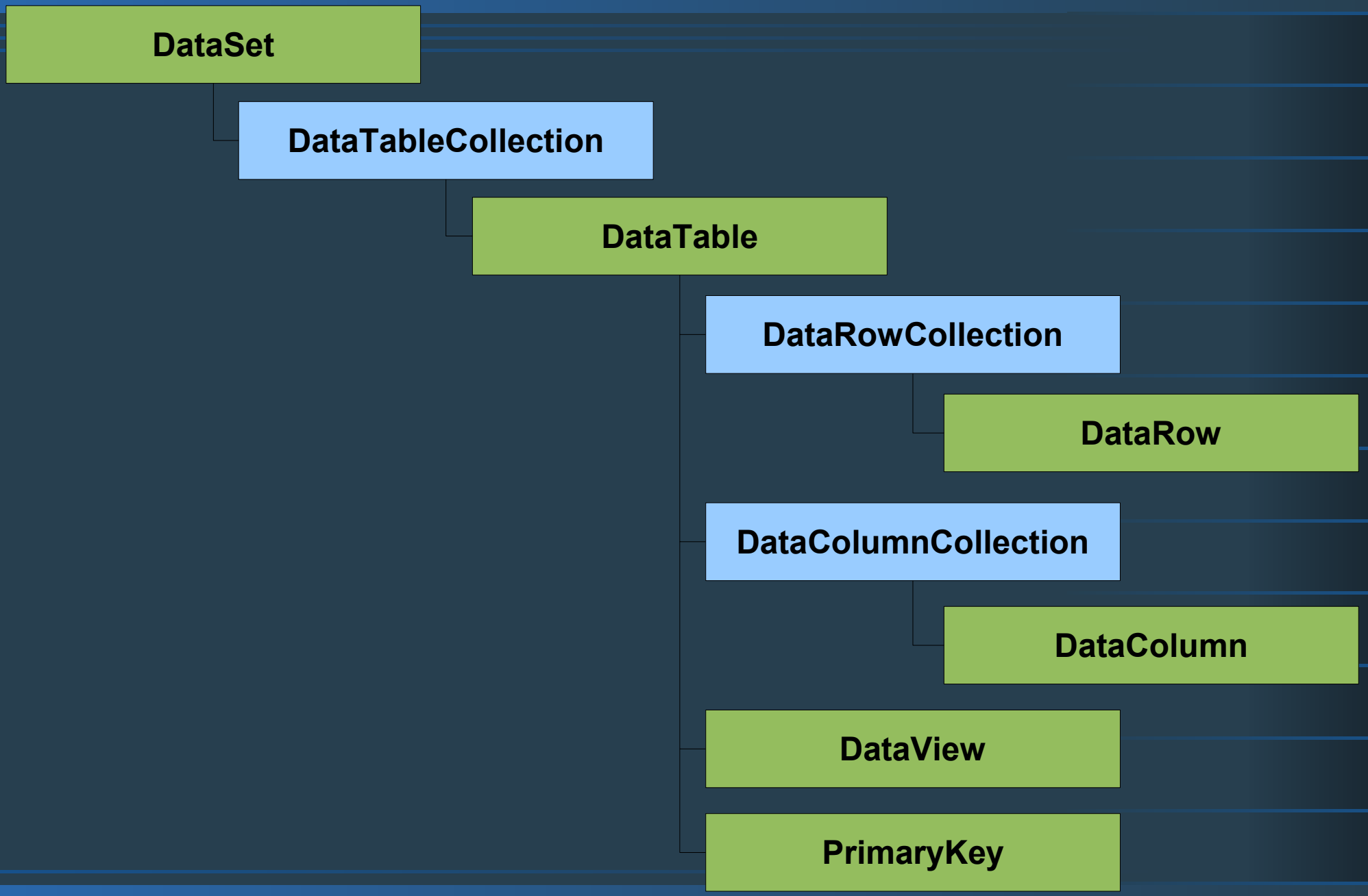


Schéma DataSet (dílčí)



DataSet

- Zdroj dat pro aplikaci
- Obsahuje tabulky a jejich omezení (constraints)
- Data mohou být z několika zdrojů (databází)

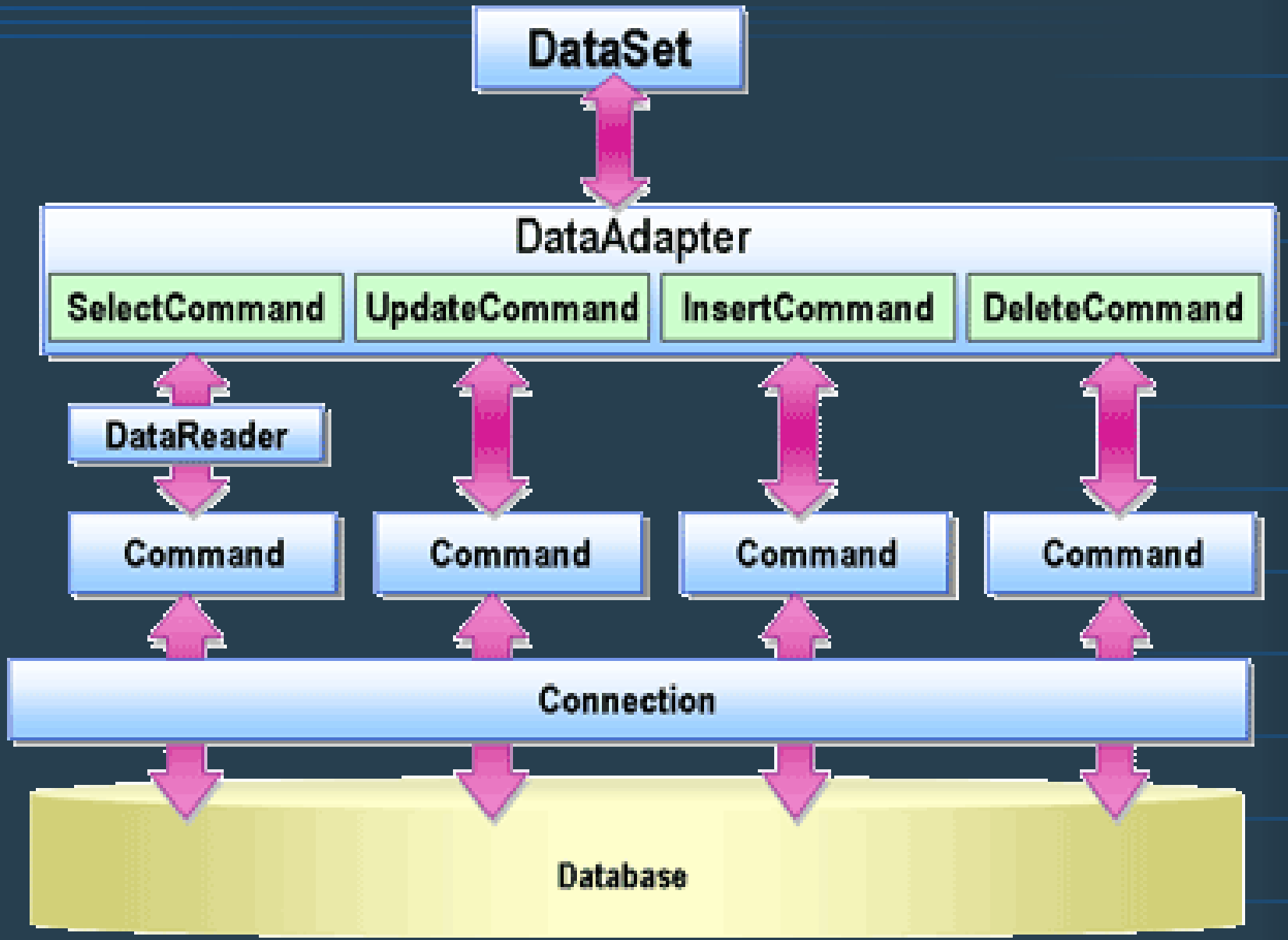
DataTable

- Odpovídá zhruba tabulce v databázi
- Jednotlivé DataTable nemusí být z jedné databáze
- Obsahuje podobné vlastnosti jako tabulky v běžných databázích (reference, klíče atd.)
- Slupce dostupné v kolekci Columns
- Řádky dostupné v kolekci Rows

Naplnění DataSet daty

- Naplnění je prováděno přes DataAdapter
- Komunikace s datovým zdrojem je prováděna přes jednotlivé příkazy (SqlCommand)
- Pro každou operaci (sel/ins/upd/del) je definován vlastní příkaz (SqlCommand)

DataSet – DataAdapter - DB



na závěr...

- SqlConnectionStringBuilder
  - Vygeneruje korektní spojovací řetězec
  - Možno generovat jednoduše programově
- Nezapomeňte si přidat jmené prostory pro práci s daty
  - System.Data
  - System.Data.SqlClient

Databázové systémy II.

Insert, update, delete v OFF-LINE aplikacích

Východiska

- Nutno vytovřit příkaz pro každou z operací ins/upd/del
- Nutno určit, které řádky se budou aktualizovat
- Je nutno nějakým způsobem předat nová data do textového příkazu
- Nutno potvrdit aktualizace
- Po provedení aktualizací na straně serveru obnovit data na straně aplikace

Základní programová struktura

```
1.insertCmd = new SqlCommand("INSERT INTO osoby(jmeno)
                                    VALUES (@jmeno)", connection);
2.insertCmd.Parameters.Add("@jmeno", SqlDbType.VarChar,50,"jmeno");
3.adapter.InsertCommand = insertCmd;
```

1. Vytvoření objektu, který reprezentuje SQL příkaz. Příkazy mají poněkud zvláštní formu, aby bylo možné předat do nich nové hodnoty. Druhý parametr je spojení, na kterém má být dotaz vykonán
2. Vytvoření parametru, který bude předáván do SQL příkazu. Je nutno specifikovat, jaká data bude parametr obsahovat.
    1. Nazev parametru v SQL dotazu
    2. Datový typ
    3. Odkud se bude brát hodnota (jaký sloupec tabulky odpovídá hodnotě tohoto parametru
3. Přiřazení SQL příkazu k příslušnému příkazu adapteru

Přidání řádku

```
DataRow myNewRow = lTable.NewRow();
myNewRow["jmeno"] = "Evzen";
lTable.Rows.Add(myNewRow);
```

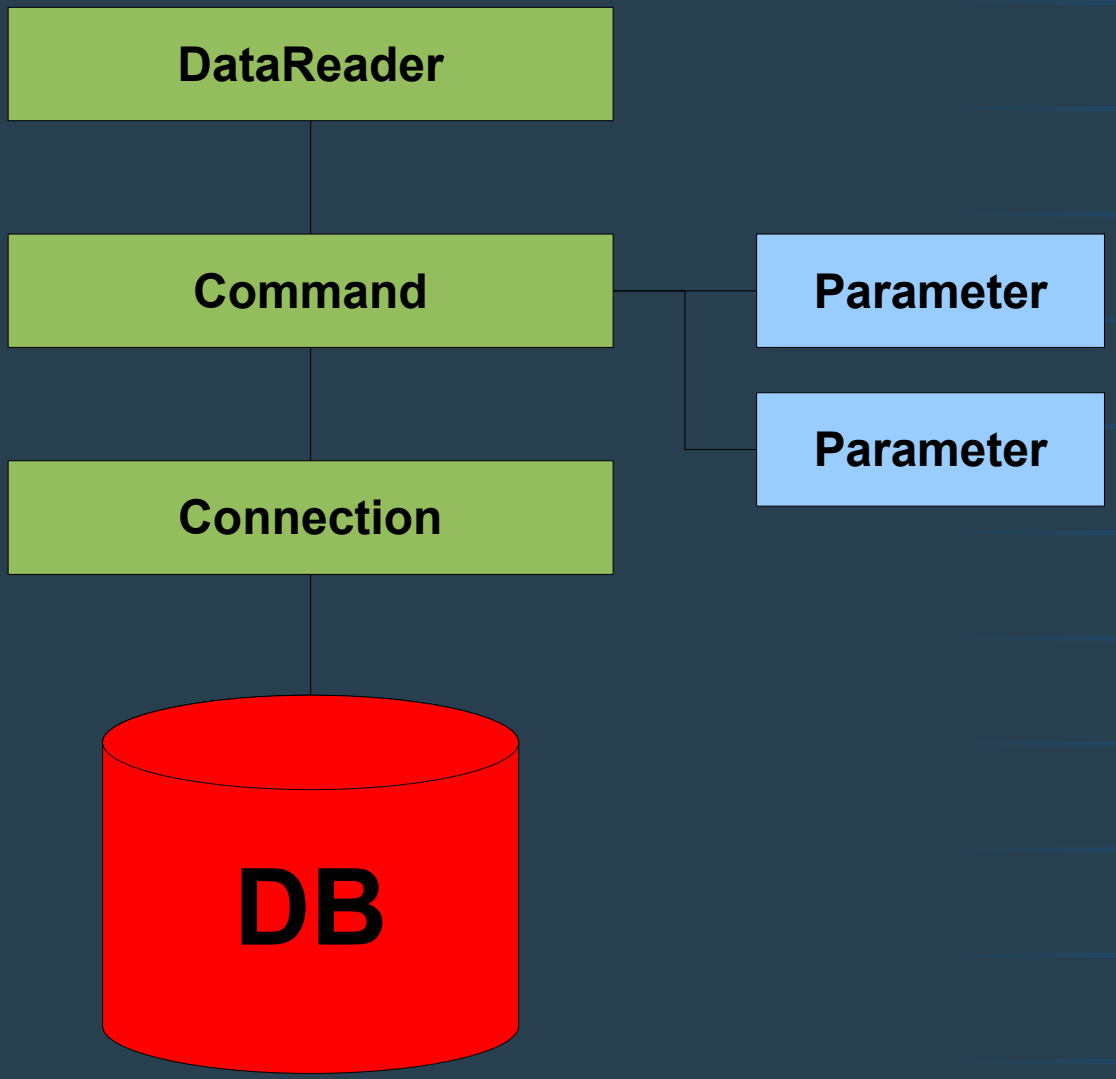
Potvrzení změn a aktualizace databáze

```
myNewRow.EndEdit();
adapter.Update(ds,"osoby");

// tento radek musi byt az PO adapter.Update()
ITable.AcceptChanges();

//obnoveni obsahu dataSetu (možno i částečně, např. jedna tabulka.)
adapter.Fill(ds);
```

Připojené (on-line) aplikace

Aplikace udržující stálé spojení s daty

Základní třídy pro on-line apliakce



Přesněji...

- Pro různé databáze existují specifické třídy (implementují společné rozhraní)
- Různé třídy pro přístup k různým druhům databází (SqlDataReader, OracleDataReader...)

Jaké jsou výhody?

- Data nejsou přenášena do lokálních kopií
- Pracujeme s jednotlivými řádky
- Jednoduchá implementace
- Kde jsme to už viděli?
  - Velice podobnou syntaxi používá PHP skript
- Podpora transakčního zpracování

Co můžeme využít?

- Klasické dotazy – ExecuteReader()
- Aktualizační příkazy – ExecuteNonQuery()
- Získání výsledků agregačních fcí. - ExecuteScalar()