.NET v SQL Serveru

NDBI039

Jan Drozen

http://www.ms.mff.cuni.cz/~drozenj

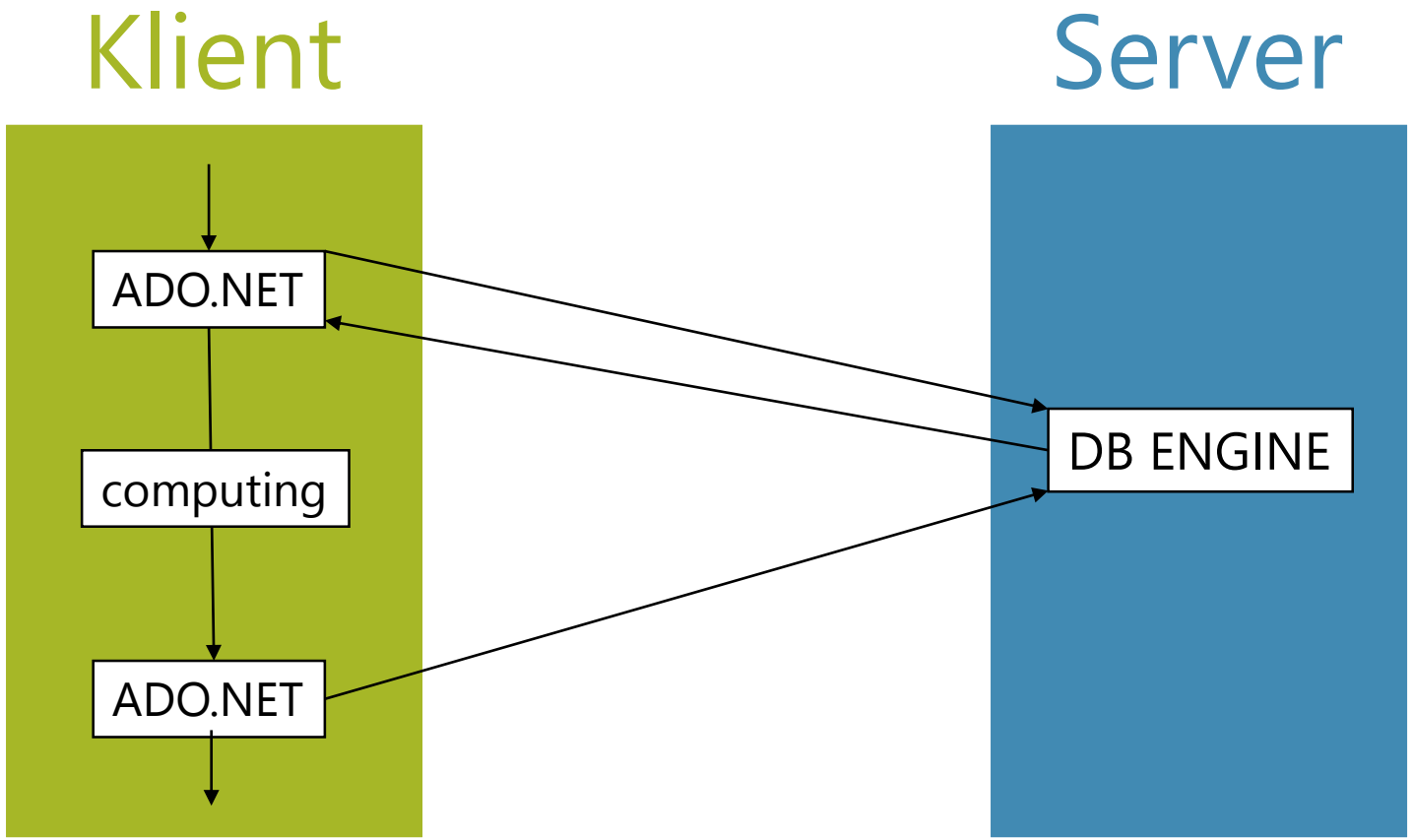
Platforma .NET

- poprvé v roce 2002 v1.0
- jednotná softwarová platforma pro PC/web/mobilní zařízení
- managed (garbage collected) prostředí
- CLR – Common Language Runtime
- jazyky podle CLS
  - C#, VB.NET, J#, F#,…

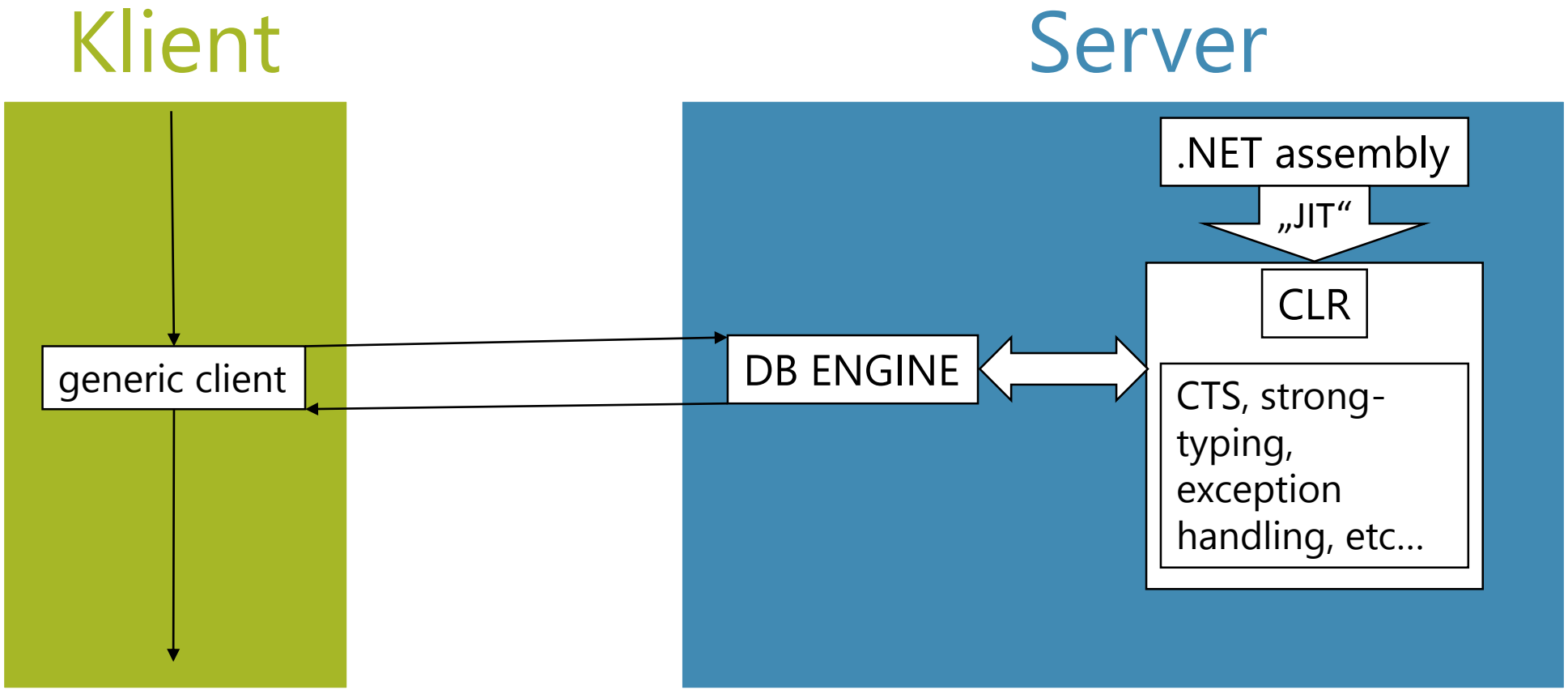
- přednášky
  - Jazyk C# a platforma .NET
  - Pokročilé programování pro platformu .NET I, II

Motivace

CLR
CTS, strong-typing, exception handling, etc…



CLR integrace



Výhody integrace CLR

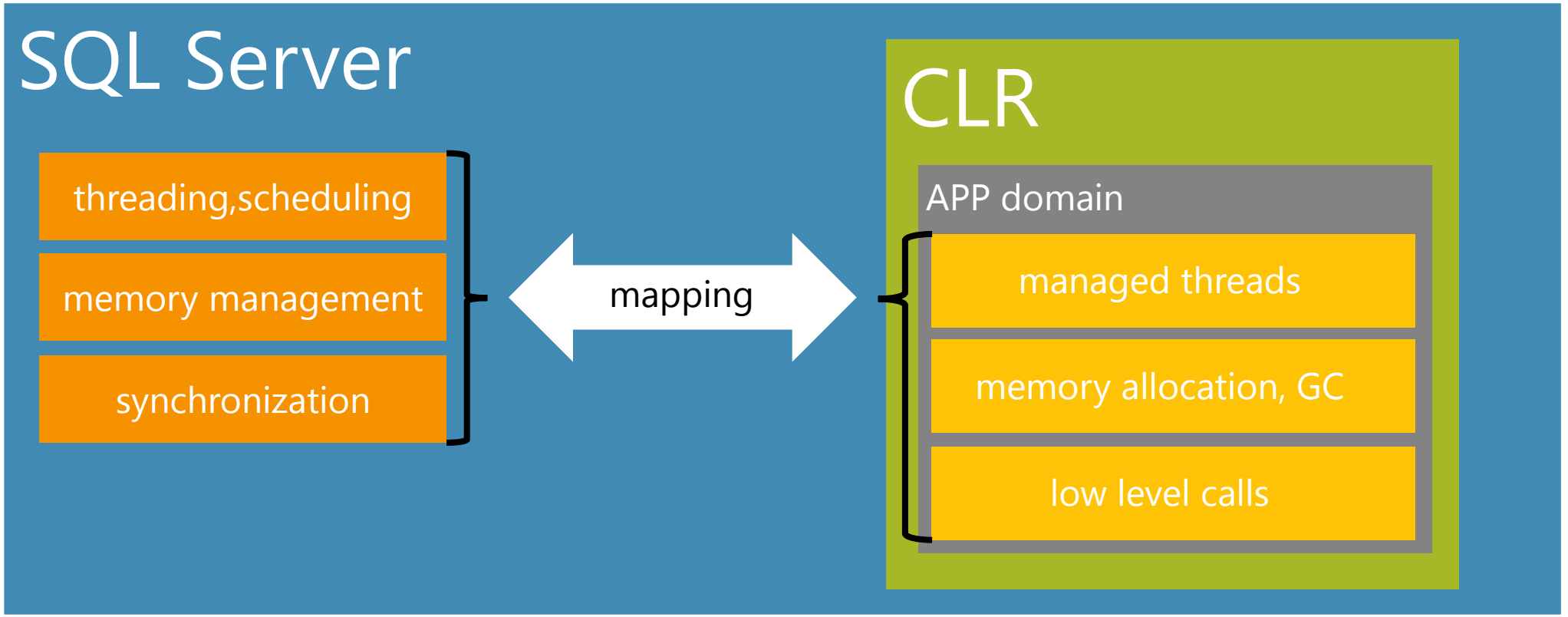
- SQL Server 2005: nahrazení Extended stored procedures (C/C++)
- větší výrazová síla
- možnost volby TSQL/.NET
- možnost volby jazyka .NET
- ušetřený traffic
- využívá se kapacita serveru

Použití CLR SQL

- typicky pro složitější výpočty
- možnost multi-thread zpracování
- objektový přístup
- možnost připojení k další DB

- pomalejší při použití, které je přirozené pro TSQL

Pod pokličkou



CLR SQL objekty

- na CLR můžeme provozovat:
  - stored-procedures
  - scalar-valued functions
  - table-valued functions
  - aggregate functions
  - user-defined types
  - triggers

CLR .NET objekty

- objektová reprezentace:
  - stored-procedures <- `public static` method
  - scalar-valued functions <- `public static` method
  - table-valued functions <- `public static` method
  - aggregate functions <- `class, structure`
  - user-defined types <- `class, structure`
  - triggers <- `public static` method

Co je třeba - I

- SQL Server >= 2005
- povolené použití CLR:

```
EXEC sp_configure 'clr enabled' , '1'
GO
RECONFIGURE
GO
```

Co je třeba - II

- vytvořit assembly podle dobře formátovaného kódu
- cílové platformy .NET frameworku:

<2012 implicitně nejspíš 2.0

SQL Server	.NET Framework
SQL Server 2005	2.0
SQL Server 2008 express	2.0 SP2
SQL Server 2008, WS 2003 (64b), IA-64	2.0 SP2
SQL Server 2008 (zbytek)	3.5 SP1
SQL Server 2012	4.0

Dobrman:

	name	value
1	directory	C:\Windows\Microsoft.NET\Framework64\v2.0.50727\
2	version	v2.0.50727
3	state	CLR is initialized

- možnost zjistit z: `sys.dm_clr_properties`
  - další: `sys.dm_clr_appdomains`, `sys.dm_clr_loaded_assemblies`,`sys.dm_clr_tasks`

Co je třeba - III

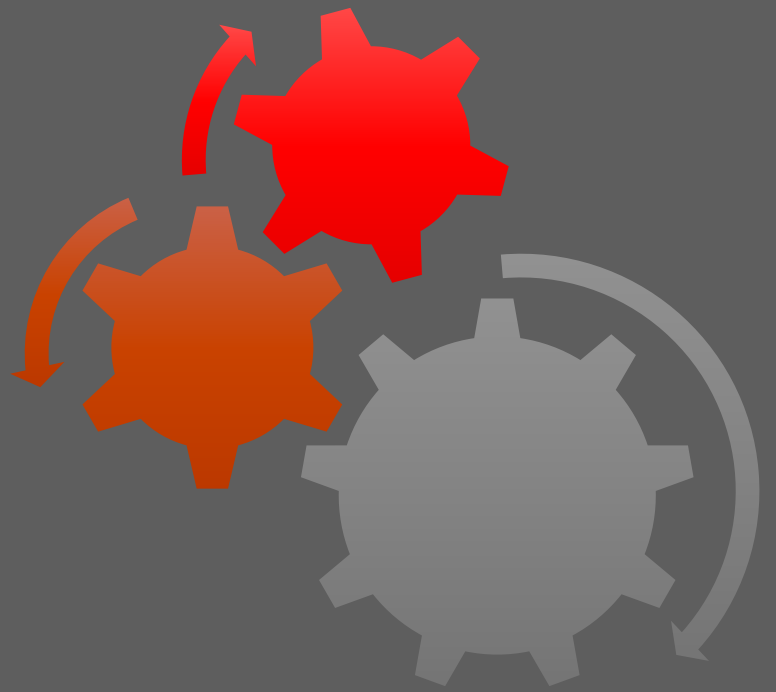
- kvůli bezpečnosti mají assembly definovanou úroveň
- standardně
  - **SAFE**
- pokud přistupují k externím zdrojům nebo unmanaged kódu, jsou označeny jako
  - **EXTERNAL_ACCESS**
    - přístup k systémovým zdrojům, jako např. soubory, síť, proměnné prostředí, registry
  - **UNSAFE**
    - přístup ke zdrojům i mimo server, volání nativního kódu
- pro takové assembly musí být v databázi povolení:
  - `ALTER DATABASE [dbname] SET TRUSTWORTHY ON`
- nebo
  - assembly podepsat
  - mít autorizovaný login

Co je třeba - IV

- nahrát assembly na server
  - VS provádí automaticky
  - klikacím způsobem přes SQL MS
  - TSQL:
    - `CREATE ASSEMBLY [jméno] FROM [cesta] WITH PERMISSION_SET [povolení]`

Co je potřeba - V

- definovat procedury, fce, atd. z assembly
  - pokud nahráváme přes VS, tak nemusíme
  - jinak TSQL:
    - `CREATE PROCEDURE [jméno] (@arg1 typ1,…) AS EXTERNAL NAME [assembly.object]`



DEMO

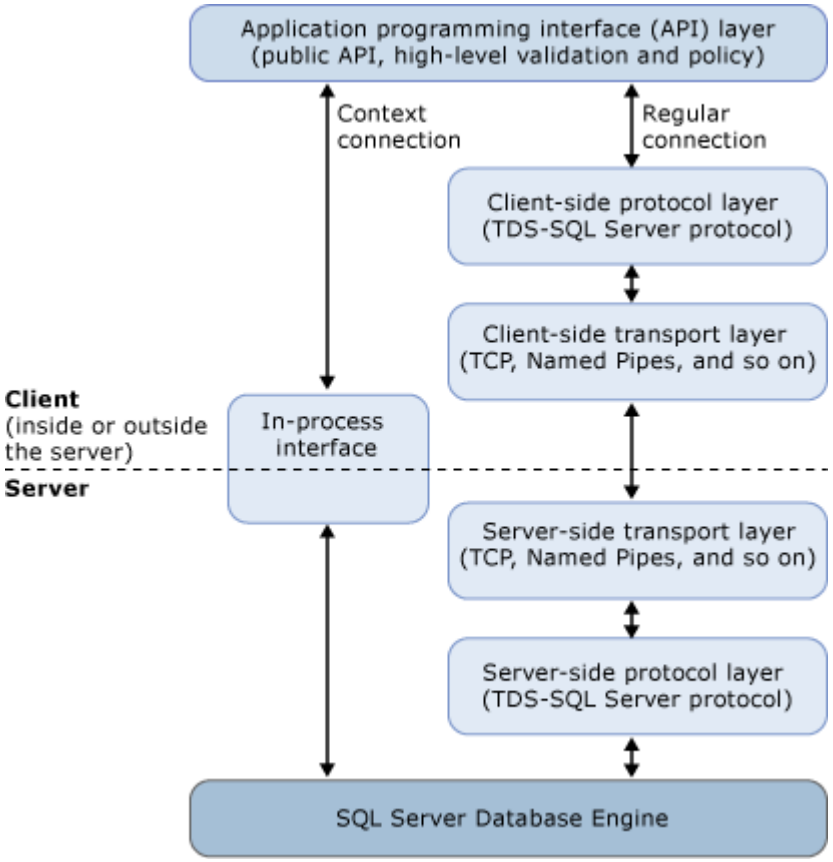
Stored procedure

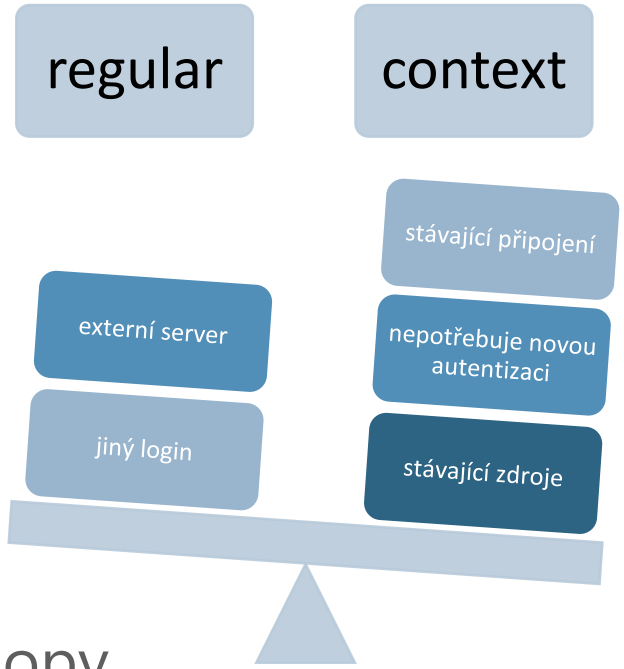
Připojení

- používáme instanci SqlConnection
  - klasické ADO.NET
  - ctor `public SqlConnection(string connectionString)`
  - použití unmanaged zdrojů, implementuje `IDisposable`
    - proto použití `using` bloku

- connection string?
  - buď úplný (*regular connection*)
    - `DATA SOURCE=[server];INITIAL CATALOG=[db] ;USER ID=[uname];PASSWORD=[passwd]`
    - `DATA SOURCE=[server];INITIAL CATALOG=[db] ;INTEGRATED SECURITY=SSPI`
    - `DATA SOURCE=[server];INITIAL CATALOG=[db] ;TRUSTED_CONNECTION=TRUE`
  - nebo použití stávajícího připojení (*context connection*)
    - `CONTEXT CONNECTION=TRUE`

Regular vs. context connection

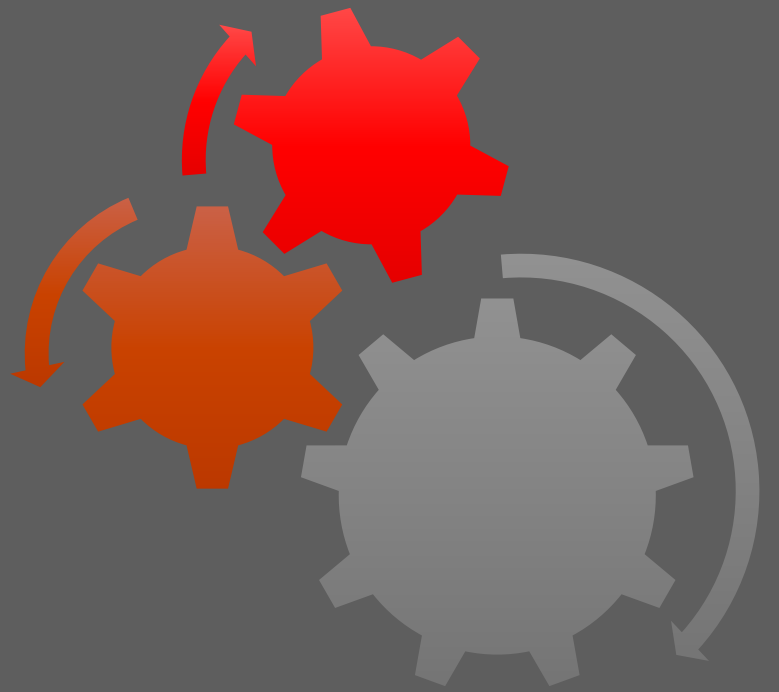




- omezení
  - context
    - nemůžeme použít SqlBulkCopy
    - žádné další upřesnění connection stringu
    - DataSource vrací null
    - a další
  - regular
    - nepodporuje asynchronní zpracování vzhledem k internímu serveru

∑ stored procedures

- jsou statické
- atribut `Microsoft.SqlServer.Server.SqlProcedure`
- návratová hodnota void

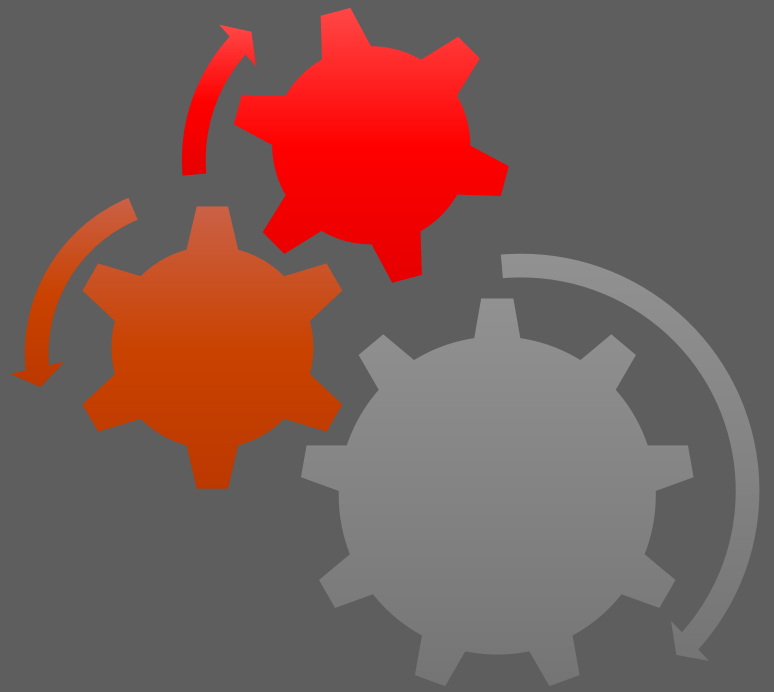


DEMO

user defined function (scalar)

∑ user defined functions (scalar)

- jsou statické
- atribut `SqlFunction(DataAccess = DataAccessKind.Read)`
- návratová hodnota .NET typu odpovídající nějakému SQL typu



DEMO

user defined function (table valued)

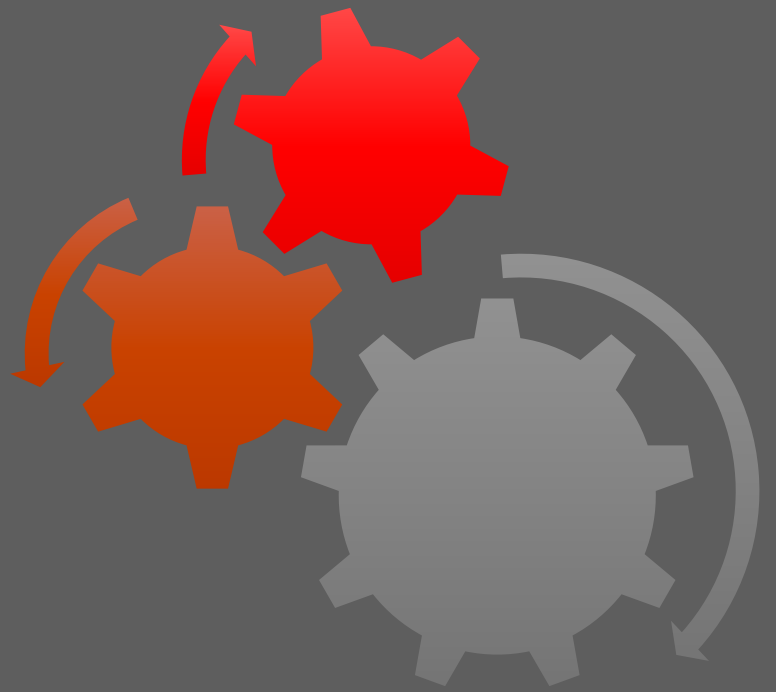
∑ user defined functions (table valued)

- jsou statické
- atribut

```
SqlFunction(
        DataAccess = DataAccessKind.Read,
        FillRowMethodName = “method",
        TableDefinition=“table_def")
```

- návratová hodnota .NET typu implementujícího `IEnumerable`
- potřeba mít statickou metodu `methodName(Object obj, out col1, out col2,…)`

- TSQL definice:
  - ```
    CREATE FUNCTION [name] (@arg1,@arg2,…)
    RETURN TABLE
    (argg1,argg2,…)
    AS
    EXTERNAL…
    ```
- nemůže vracet ne-unicode varchar a timestamp
    ```



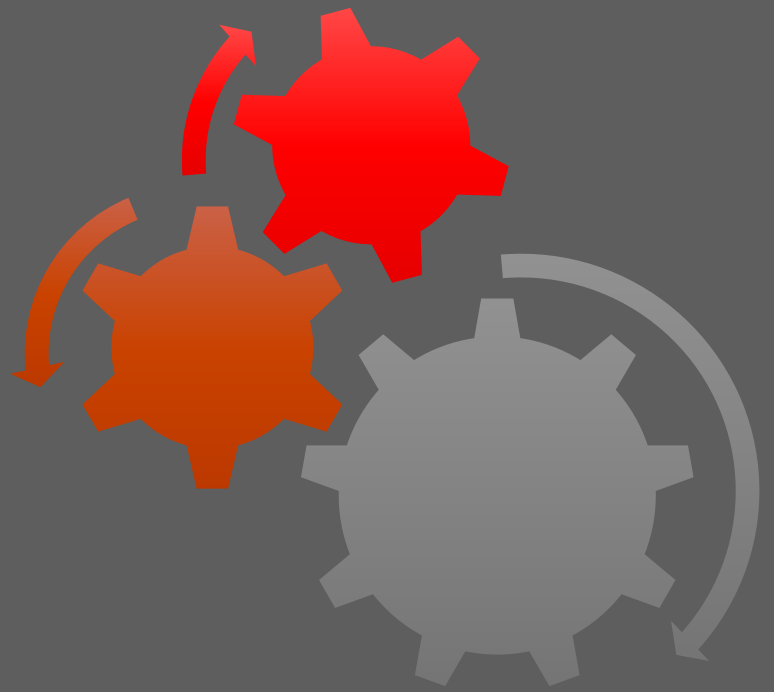
DEMO

user-defined aggregate

∑ user defined aggregates

- provádí výpočet nad množinou řádků a vrací jednu hodnotu
- atribut `SqlUserDefinedAggregate`
- 4 metody (instanční):
  - `public void Init`
  - `public void Accumulate`
  - `public void Merge`
  - `public type Terminate`

- TSQL definice:
  - `CREATE AGGREGATE [name] (@input [type]) RETURNS [type] EXTERNAL NAME…`



DEMO

user-defined type

∑ user defined types

- atribut `SqlUserDefinedType`
  - volitelně `Serializable`
- musí implementovat `INullable`
- musí implementovat `public` metody `Parse` a `ToString`
- musí implementovat `IXmlSerializable`
  - nebo všechny public položky musí být `XmlSerializable`
  - nebo musí být označeny `XmlIgnore` parametrem
- jména public položek kratší než 128 znaků a musí odpovídat konvencím SQL serveru
- statické položky musí být konstanty nebo read-only

Debuging

- dá se povolit z VS
- možnost i z MS
- při krokování postupuje přes příkazy posílané na server (dynamic SQL)
- nepodporuje SQL Server Express ☹

Shrnutí

- SQL server umožňuje implementovat v rámci procedur, UDF, atd. takřka cokoli
  - můžeme část aplikační logiky převést na server
- může ale nemusí být rychlejší
- managed prostředí
  - zdroje mapované na zdroje SQL serveru
- regular vs. context connection

LINQ

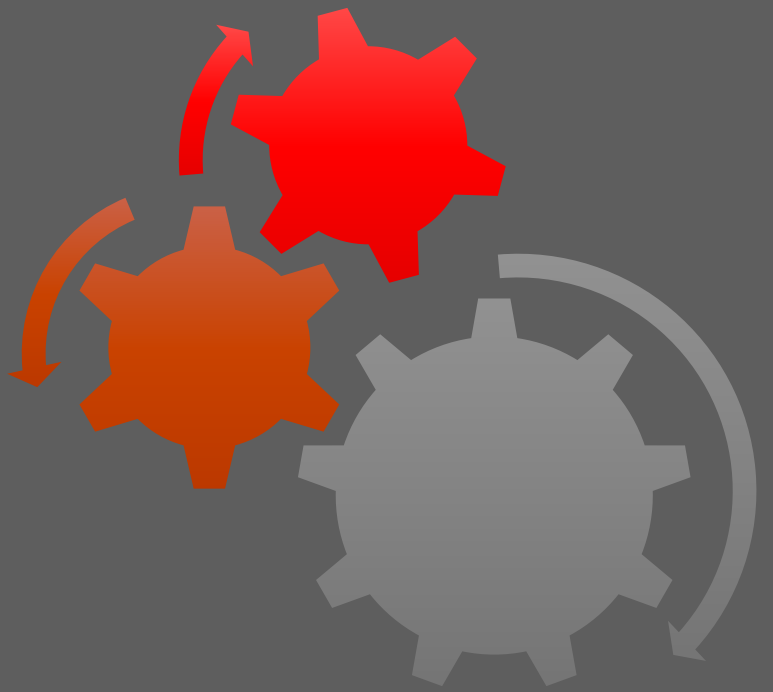
LINQ

- Language INtegrated Query
- dotazovací jazyk
- C# 3.0, VB9, .NET 3.5
- zkrácení enumerace
- varianty podle cíle dotazu
  - LINQ to
    - objects
    - SQL
    - XML
    - …
- implementace pomocí extension metod

LINQ to SQL

- odstíní programátora .NET od SQL
- není potřeba `SqlCommand`, apod.
- mapuje objekty na entity RD
  - metody na funkce a procedury
- potřeba mít podpůrné datové struktury
  - O/R Designer (VS)
  - atributy Table, Column

- více se používá entity-framework
  - LINQ to entities



DEMO

LINQ to SQL

Zdroje

- MSDN documentation
- SQL Server books online
- Programming Microsoft SQL Server 2012
  Lobel, Leonard G.
  Brust, Andrew J.
  TALLAN,2012

KONEC