DataOps pentru utilizarea ”code managed” în baze de date

Conf.dr. Cristian KEVORCHIAN

Facultatea de Matematica si Informatica

ck@fmi.unibuc.ro

DevOps în context AGILE

- **Scop**: DevOps în Agile se concentrează pe integrarea dezvoltării software-ului și a operațiunilor IT pentru a accelera livrarea de software de înaltă calitate.
- **Practici cheie**: Integrare continuă/Implementare continuă (CI/CD), testare automată, infrastructură ca cod, monitorizare și feedback continuu.
- **Echipe implicate**: Dezvoltatori, operațiuni IT, asigurarea calității și echipe de securitate.
- **Beneficii**: Creșterea ritmului de implementare, reducerea timpului de transfer în producție, scăderea ratei de eșec a noilor versiuni și reducerea timpului de remediere a problemelor.
- **Automatizare**: Jenkins, Docker, Kubernetes, Ansible.

MLOps în context AGILE

- **Scop**: Automatizarea și gestionarea ciclului de viață al modelelor de învățare automată (ML) pentru a asigura performanța și acuratețea acestora în producție.
- **Practici cheie**: Gestionarea datelor, antrenarea modelelor, implementarea automată a modelelor, monitorizarea și întreținerea acestora.
- **Echipe implicate**: Ingineri de date, oameni de știință ai datelor, ingineri ML, IT și părți interesate din afaceri.
- **Instrumente**: TensorFlow Extins(TFX), MLflow, Kubeflow.

DataOps în context AGILE

- **Scop**: Aplicarea metodologiilor agile la analiza și gestionarea datelor pentru a îmbunătăți viteza și calitatea proceselor legate de date.
- **Obiective**: Reducerea timpului de ciclu al analizei datelor, îmbunătățirea calității datelor și asigurarea că datele sunt fiabile și accesibile pentru deciziile de afaceri.
- **Practici cheie**: Automatizarea fluxurilor de date, monitorizarea calității datelor, integrarea datelor și colaborarea între rolurile de data engineer, data scientist și dezvoltatorii de modele.
- **Echipe implicate**: data engineer, data science, data analist, utilizatori.
- **Instrumente**: Apache Airflow, dbt, instrumente de monitorizare a calității datelor.

Metodologii Specializate și Principiile Agile

DataOps – Definiție

- **DataOps** este o practică ce combină gestionarea datelor (Data Management) cu operațiunile (Operations) pentru a asigura procesarea datelor cu acuratețe și eficiență pe întregul lor ciclu de viață.
- DataOps nu este o metodologie sau un framework specific, precum Microsoft Solutions Framework (MSF). În schimb, DataOps reprezintă o abordare sau o practică ce se concentrează pe gestionarea și operaționalizarea datelor într-un mod eficient și precis.
- În contextul dezvoltării de cod în paradigme arhitecturale, cum ar fi MVC (Model-View-Controller), DataOps este orientat către managementul și operaționalizarea datelor în cadrul acestei structuri.

Interacțiunea DataOPS cu MVC

- **Modelul (Model)**: În cadrul DataOPS, ”Model-ul” nu se referă doar la structura și logica de gestionare a datelor, ci și la operațiunile cu date. Aici sunt incluse activități precum data cleaning, validarea datelor, procesarea datelor în timp real și asigurarea că datele sunt stocate și accesate într-un mod eficient și sigur. Modelul în cadrul DataOPS trebuie să fie proiectat și implementat în așa fel încât să îndeplinească cerințele specifice ale operațiilor asupra datelor.
- **Vizualizarea** (View): În contextul DataOPS, Vizualizarea poate include instrumente de vizualizare și raportare a datelor care sunt utilizate pentru a monitoriza și analiza datele în timp real. Aceste instrumente pot fi proiectate pentru a furniza utilizatorilor o înțelegere clară și intuitivă a datelor, permițându-le să ia decizii documentate.
- **Controlerul** (Controller): Controlerul în cadrul paradigmei MVC este responsabil pentru gestionarea interacțiunii utilizatorului cu aplicația și pentru coordonarea actualizării și manipulării modelelor și a vederilor. În cadrul DataOPS, Controlerul poate implementa logica necesară pentru a operaționaliza datele, asigurându-se că datele sunt procesate și tratate corespunzător în cadrul aplicației.

Fluxuri de operatii in procesare date

- Conectarea la sursele de date
- Pregatirea aplicatiei pentru a primi date
- Prelucrarea datelor in aplicatie
- Prezentarea datelor
- Editarea datelor in aplicatia dezvoltata
- Validarea datelor
- Salvarea datelor

Proiecte ”SQL Server” în context DataOPS

- **Modelarea datelor:** În Visual Studio, puteți utiliza instrumentele de proiectare pentru a modela bazele de date SQL Server. Aceasta implică proiectarea schemelor de bază de date, inclusiv tabelele, relațiile, indicii și alte obiecte. În contextul DataOPS, este important să aveți o proiectare corespunzătoare a bazelor de date care să permită o gestionare eficientă a datelor și să se alinieze cu cerințele operaționale.
- **Controlul versiunilor:** Utilizarea sistemelor de control al versiunilor, cum ar fi Git, pentru a gestiona schimbările în schema bazei de date și în scripturile SQL. Acest lucru asigură o gestionare coerentă și auditabilă a modificărilor în structura și conținutul bazei de date și facilitează colaborarea între membrii echipei de dezvoltare.
- **Automatizarea proceselor de implementare**: Utilizarea funcționalităților din Visual Studio, cum ar fi publicarea bazelor de date, pentru a automatiza procesele de implementare a modificărilor în medii de dezvoltare, testare și producție. Acest lucru ajută la menținerea consistenței între diferitele medii și reduce riscul de erori umane în timpul implementării.
- **Testarea și monitorizarea**: Integrarea testelor unitare și a testelor de integrare pentru a asigura calitatea datelor și pentru a detecta eventualele probleme într-un stadiu incipient. De asemenea, monitorizarea performanței și a disponibilității bazei de date este importantă pentru a asigura o funcționare stabilă și fiabilă a aplicației.
- **Securitatea datelor**: Implementarea măsurilor de securitate adecvate pentru a proteja datele împotriva accesului neautorizat și a pierderilor. Aceasta include gestionarea drepturilor de acces la date, criptarea datelor sensibile și implementarea altor controale de securitate în conformitate cu cerințele legale și de conformitate.

Procedura Stocata

```
CREATE PROCEDURE [dbo].[ProcCreareInserare]
AS
  BEGIN
  -- Verificăm dacă tabela există deja și o ștergem dacă este cazul
IF OBJECT_ID('ckMDSTab2024', 'U') IS NOT NULL
BEGIN
DROP TABLE ckMDSTab2024;
END
   -- Creare tabela noua
CREATE TABLE ckMDSTab (
        ID INT IDENTITY(1,1),
        K1 VARCHAR(20)
       );

       -- Inserăm 15 înregistrări în tabelă
       DECLARE @Kounter INT = 1;
       WHILE @Kounter <= 15
       BEGIN
           INSERT INTO ckMDSTab (K1) VALUES ('Randul' + CAST(@Kounter AS VARCHAR(2)));
           SET @Kounter = @Kounter + 1;
       END
   END
```

Proceduri stocate in CLR

- Executa business logic complex
- Procedurile stocate CLR asigura un management efficient al memoriei printr-o politica de tipuri sigura
- Management riguros al codului prin orientare catre OOP(polimorfism, incapsulare si mostenire)
- Poate fi scris in C# sau VB

CLR vs T-SQL

- C#-CLR

```
using System;
using System.Data;
using System.Data.SqlClient;
using Microsoft.SqlServer.Server;

public partial class StoredProcedures
{
    [SqlProcedure]
    public static void CreateckTable()
    {
        comanda SQL pentru crearea tabelei
        string sqlCommandText = @"
            CREATE TABLE ckTableCLR (
                ID INT PRIMARY KEY,
                K1  NVARCHAR(50),
               K2  INT
            )
        ";

        // Executa comanda SQL
        using (SqlConnection connection = new SqlConnection("context
connection=true"))
        {
            connection.Open();
            SqlCommand command = new SqlCommand(sqlCommandText,
connection);
            command.ExecuteNonQuery();
        }
    }
}
```

- T-SQL

```
CREATE TABLE ckTable (
    ID INT PRIMARY KEY,
    K1 NVARCHAR(50),
    K2 INT
);
```

Proceduri Stocate Standard vs. Proceduri Stocate CLR

- ”Business logic” de complexitate mare.
- Dacă fluxul de prelucrari necesită resurse CPU superioare atunci ”code managed” aduce avantaje.
- Taskuri imposibil de implementat in T-SQL cum ar fi accesul la servicii web.

Pasii pentru crearea unei proceduri stocate

- Database Project
- SQL/CLR Debugging
- Adaugarea urmatoarei secvente de cod metodei create implicit: “context connection=true”

```
using(SqlConnection connection = new
SqlConnection("context connection=true"))
```

- Procedura stocata CLR este parte a bazei de date create si nu este nevoie de conexiune externa
- Trebuie executat urmatorul query:

```
sp_configure ‘clr enabled’, 1
```

Avantaje arhitecturale

- Solutia trebuie sa fie sigura, usor de implementat si usor de implementat
- Bazat pe un model minimal “3-tire”, in care orice modificare pe un tire implica instalarea dll-ului afferent.(interfata utilizator+business logic+data layer)
- O singura arhitectura pentru desktop si web.
- Este posibila izolarea proceselor in dezvoltare. Un dezvoltator poate lucre cu baza de date altul poate lucre cu business logic sau interfetele utilizator.
- Nu este nevoie pentru scrierea de interogari specific(in-line queries)

Varianta “5-tiers”

- Tier 1 – Nivelul prezentare(WebForms, WinForms). Datele din si spre nivelul de prezentare sunt filtrate de business logic(nu direct cu nivelul de date).
- Tier 2-Business Logic Layer, se spliteaza in : core business logic si data access logic
- Tier 3- Datele relative la operatiile dintre aplicatii si bazele de date vor fi realizate la acest nivel. Creat in LINQ
- Tier 4 - Proceduri stocate CLR.Instalat in SQL server.
- Tier 5 – Datele si obiectele destinate intretinerii acestora

ORM(Object Relational Mapping)

- Tehnica de programare pentru convertirea datelor, vazute ca obiecte, intre medii diferite cum ar fi sistemele de gestiune a bazelor de date(SQL Server, ORACLE, DB2, etc) si limbaje de programare OO cum ar fi (C#,VB.NET, Java, etc)
- Tehnica se bazeaza pe crearea unei baze de virtuale OO in vederea maparii cu obiectele limbajelor de programare.

LINQ

LINQ(Language-Integrated Query) este un concept introdus in Visual Studio 2008 si .NET Framework 3.5 care face legătura dintre “lumea obiectelor” si “lumea datelor”.

Componenta a platformei .NET, care ofera posibilitatea interogarii datelor direct din mediul de dezvoltare bazat pe limbaje .NET.

Defineste o familie de operatori pentru filtrarea componentelor unor vectori, colectii numarabile, fisiere XML, baze de date relationale.

Daca sursa de date nu stocheaza datele dupa un model OO trebuie facuta o mapare intre acestea si modelul obiectual.

Interogarile scrise folosind operatorii de interogare sunt executate fie prin intermediul motorului LINQ fie prin transmitere intr-un mediu diferit(SQL Server) in vederea executiei.

Rezultatul interogarii este o colectie de obiecte stocate in memorie care pot fi enumerate cu ajutorul unei functii de parcurgere, cum ar fi foreach din C#.

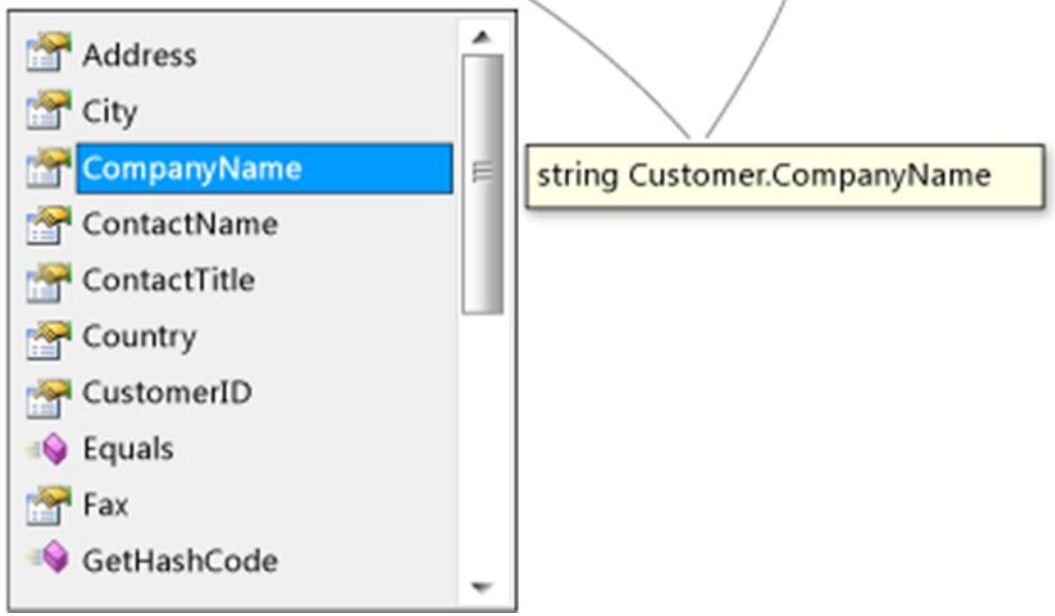
EXEMPLU

```
Northwnd nw = new
Northwnd(@"northwnd.mdf");

var companyNameQuery =
    from cust in nw.Customers
    where cust.
```

```
Dim nw As New _
Northwnd("c:\northwnd.mdf")

Dim companyNameQuery = _
    From cust In nw.Customers _
    Where cust.
```



LINQ to SQL

- LINQ to SQL este un provider dedicate aplicatiilor care stocheaza date in SQL Server.
- Nu este nevoie de un motor de procesare al interogarilor, acestea fiind “traduse” in limbaj SQL si transmise serverului de baze de date pentru procesare.
- **Obiectul DataContext – canalul prin care se obtin obiectele din baza de date si prin care se transmit modificarile catre baza de date.(similar conexiunii din ADO.NET)**
- **DataContext se initializeaza cu un obiect de tip conexiune sau un sir de caractere.**
- **DataContext implementeaza operatori specifici LINQ.**

Întrebări