ASPNETCore WebAPI-ADO.NET STORE PROCEDURE

CRUD(read,create,update,delete)

Açıklama:

Bu yazımda C# ya da ASP.net ile veritabanı işlemlerinde Stored Procedure kullanmayı göstereceğim.Bütün select,insert,delete ve update işlemlerini gerçekleştirirken stored procedure kullanmak en faydalı yöntemdir.Bunun sebebi hem güvenlidir hemde Execute sistemini bir kere çalıştırıp bir daha kasmamasıdır.

VERİ TABANI

```
CREATE DATABASE ProductDB; db oluşturma

GO

-- Veritabanı kullanımı

USE ProductDB;

GO

-- Products tablosu oluşturma

CREATE TABLE Products (

   ProductId INT PRIMARY KEY IDENTITY(1,1),

   Name NVARCHAR(100) NOT NULL,

   Price DECIMAL(18,2) NOT NULL,

   Description NVARCHAR(MAX)

);
```

STORE PROCEDURES

sp_GetProducts

```
CREATE PROCEDURE dbo.GetProducts

AS

BEGIN

SELECT * FROM Products;

END
```

sp_GetProductById

```
CREATE PROCEDURE dbo.GetProductById
```

@ProductId INT

AS

BEGIN

SELECT * FROM Products WHERE ProductId = @ProductId; bu id ye göre GET işlemi yapar.

END

sp_InsertProduct

CREATE PROCEDURE dbo.InsertProduct

@Name NVARCHAR(100),

@Price DECIMAL(18, 2),

@Description NVARCHAR(MAX)

AS

BEGIN

INSERT INTO Products (Name, Price, Description)

VALUES (@Name, @Price, @Description); yeni değer ataması yapar.

END

sp_UpdateProduct

CREATE PROCEDURE dbo.UpdateProduct

@ProductId INT,

@Name NVARCHAR(100),

@Price DECIMAL(18, 2),

@Description NVARCHAR(MAX)

AS

BEGIN

UPDATE Products

SET Name = @Name, Price = @Price, Description = @Description property leri set etme değiştirme,güncelleme

```
  WHERE ProductId = @ProductId;
END
```

sp_DeleteProduct

```
CREATE PROCEDURE dbo.DeleteProduct
  @ProductId INT
AS
BEGIN
  DELETE FROM Products WHERE ProductId = @ProductId;
END
```

bu id ye göre silme işlemi yap

ADONET CRUD İŞLEMLER

```
// ProductsController.cs
using Microsoft.AspNetCore.Mvc;
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
namespace YourNamespace.Controllers
{
  [Route("api/[controller]")]
  [ApiController]
  public class ProductsController : ControllerBase
  {   //db bağlantısı start
    private readonly string _connectionString;
    public ProductsController(IConfiguration configuration)
```

Generic paketi LİST den gelir.

=> Kullanılan ado.net paketleri

```
{
                                                                    appSettings.json daki bağlantı adresinin adı
   _connectionString =configuration.GetConnectionString("DefaultConnection");
} //db bağlantısı end
[HttpGet]
// list,SqlCommand,CommandType,StoreProcedure,ExecuteReader,while
public IActionResult GetProducts()
{
        Product listesini al
   var products = new List<Product>();
   using (var connection = new SqlConnection(_connectionString))
   {
                                          StoreProcedure
      var command = new SqlCommand("sp_GetProducts", connection);
                                                                bağlantı
      command.CommandType = CommandType.StoredProcedure;
      connection.Open(); bağlantıyı aç
      using (var reader = command.ExecuteReader())
      {
         while (reader.Read())    verileri oku (Read)
         {
            var product = new Product
            {
               ProductId = Convert.ToInt32(reader["ProductId"]),    İnteger olarak oku
               Name = reader["Name"].ToString(),  string olarak oku
               Price = Convert.ToDecimal(reader["Price"]),
               Description = reader["Description"].ToString()
            };
            products.Add(product); okuduktun sonra ekle,getir
```

```
            }
          }
        }
        return Ok(products);  eklediğin,getirdiğin product ları dön
    }
    [HttpGet("{id}")]
//SqlCommand,CommandType,StoreProcedure,ExecuteReader,Parameters.
 AddWithValue,while
    public IActionResult GetProductById(int id)
    {
        using (var connection = new SqlConnection(_connectionString))
        {
                                                    StoreProcedure          bağlantı
            var command = new SqlCommand("sp_GetProductById", connection);
            command.CommandType = CommandType.StoredProcedure;
            command.Parameters.AddWithValue("@ProductId", id);
            connection.Open();
            using (var reader = command.ExecuteReader())
            {
                if (reader.Read())  verileri oku(Read)
                {
                    var product = new Product
                    {
                        ProductId = Convert.ToInt32(reader["ProductId"]),
                        Name = reader["Name"].ToString(),
                        Price = Convert.ToDecimal(reader["Price"]),
```

```
                Description = reader["Description"].ToString()
            };
            return Ok(product);
        }
        else
        {
            return NotFound();
        }
    }
}
```

bu id de bir product yoksa NotFound/404 döner.

```
[HttpPost]
//SqlCommand,CommandType,StoreProcedure,ExecuteReader,Parameters.
AddWithValue,ExecuteNonQuery,while
public IActionResult InsertProduct(Product product)
{
    using (var connection = new SqlConnection(_connectionString))
    {
        var command = new SqlCommand("sp_InsertProduct", connection);
        command.CommandType = CommandType.StoredProcedure;
        command.Parameters.AddWithValue("@Name", product.Name);
        command.Parameters.AddWithValue("@Price", product.Price);
        command.Parameters.AddWithValue("@Description", product.Description);
        connection.Open();
        command.ExecuteNonQuery();
```

StoreProcedure

bağlantı

Bağlantı üzerinde bir Transact-SQL deyimi yürütür ve etkilenen satır sayısını döndürür.

```
            return Ok();
        }
    }
    [HttpPut("{id}")]
//SqlCommand,CommandType,StoreProcedure,Parameters.
 AddWithValue,ExecuteNonQuery,while
    public IActionResult UpdateProduct(int id, Product product)
    {
        using (var connection = new SqlConnection(_connectionString))
        {
                                                    StoreProcedure
            var command = new SqlCommand("sp_UpdateProduct", connection);
            command.CommandType = CommandType.StoredProcedure;
            command.Parameters.AddWithValue("@ProductId", id);
            command.Parameters.AddWithValue("@Name", product.Name);
            command.Parameters.AddWithValue("@Price", product.Price);
            command.Parameters.AddWithValue("@Description", product.Description);
            connection.Open();
            command.ExecuteNonQuery();
            return Ok();
        }
    }
    [HttpDelete("{id}")]
//SqlCommand,CommandType,StoreProcedure,Parameters.
 AddWithValue,ExecuteNonQuery,while
    public IActionResult DeleteProduct(int id)
```

Bağlantı üzerinde bir Transact-SQL deyimi yürütür ve etkilenen satır sayısını döndürür.

```
    {
        using (var connection = new SqlConnection(_connectionString))
        {
            var command = new SqlCommand("sp_DeleteProduct", connection);
            command.CommandType = CommandType.StoredProcedure;
            command.Parameters.AddWithValue("@ProductId", id);
            connection.Open();
            command.ExecuteNonQuery();
            return Ok();
        }
    }
  }
}
```

StoreProcedure

Bağlantı üzerinde bir Transact-SQL deyimi yürütür ve etkilenen satır sayısını döndürür.