Лекция 14

Современные СУБД. Организация связи Microsoft SQL Server и Microsoft Visual Studio.

Цель

Изучить современные системы управления базами данных и освоить интеграцию Microsoft SQL Server с Microsoft Visual Studio для разработки приложений, работающих с базами данных.

Основные вопросы

1. Обзор современных СУБД;
2. Архитектура Microsoft SQL Server;
3. Интеграция SQL Server с Visual Studio;
4. Технологии доступа к данным: ADO.NET, Entity Framework;
5. Создание приложений с использованием базы данных;

Лекция

Современные СУБД

Современные СУБД можно классифицировать по моделям данных:

Реляционные СУБД (RDBMS)

- Microsoft SQL Server - корпоративная СУБД от Microsoft;
- Oracle Database - мощная СУБД для крупных предприятий;
- MySQL - популярная открытая СУБД;
- PostgreSQL - продвинутая открытая СУБД;

NoSQL СУБД

- MongoDB - документо-ориентированная СУБД;
- Redis - хранилище ключ-значение в памяти;
- Cassandra - распределенная колоночная СУБД;

NewSQL СУБД

- Google Spanner - глобально распределенная СУБД;
- CockroachDB - совместимая с PostgreSQL распределенная СУБД;

Архитектура Microsoft SQL Server

Компоненты SQL Server:

- Ядро СУБД - основной компонент для обработки запросов;
- SQL Server Agent - автоматизация задач;
- SQL Server Reporting Services (SSRS) - создание отчетов;
- SQL Server Integration Services (SSIS) - интеграция данных;
- SQL Server Analysis Services (SSAS) - аналитическая обработка;

Интеграция SQL Server с Visual Studio

Технологии доступа к данным

ADO.NET - базовая технология доступа к данным в .NET

Entity Framework - ORM (Object-Relational Mapping) для работы с данными как с объектами

Подключение к SQL Server из Visual Studio

Способы подключения:

- Server Explorer в Visual Studio;
- Строка подключения в коде;
- Конфигурационные файлы;

Строка подключения:

```
<connectionStrings>
   <add name="DefaultConnection"
        connectionString="Server=localhost;Database=MyDB;Integrated
Security=True;"
        providerName="System.Data.SqlClient" />

</connectionStrings>
```

Создание приложения с базой данных

Пример: Простое приложение с ADO.NET

```
using System;
using System.Data;
using System.Data.SqlClient;

class Program
{
    static void Main()
    {
        string connectionString = "Server=localhost;Database=MyDB;Integrated
Security=True;";

        using (SqlConnection connection = new SqlConnection(connectionString))
        {
            connection.Open();

            // Выполнение запроса
            string query = "SELECT * FROM Employees WHERE Department =
@Department";
            SqlCommand command = new SqlCommand(query, connection);
            command.Parameters.AddWithValue("@Department", "IT");

            SqlDataReader reader = command.ExecuteReader();

            while (reader.Read())
            {
                Console.WriteLine($"Name: {reader["FirstName"]}
{reader["LastName"]}, Salary: {reader["Salary"]}");
            }

            reader.Close();
        }
    }

}
```

Пример: Приложение с Entity Framework

Установка Entity Framework:

```
Install-Package EntityFramework
```

Создание модели:

```
public class Employee
{
    public int EmployeeID { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Department { get; set; }
    public decimal Salary { get; set; }
}

public class CompanyContext : DbContext
{
    public DbSet<Employee> Employees { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder
optionsBuilder)
    {
        optionsBuilder.UseSqlServer("Server=localhost;Database=MyDB;Integrated
Security=True;");
    }

}
```

Использование контекста:

```
using (var context = new CompanyContext())
{
    // Добавление сотрудника
    var newEmployee = new Employee
    {
        FirstName = "John",
        LastName = "Doe",
        Department = "IT",
        Salary = 50000
    };
    context.Employees.Add(newEmployee);
    context.SaveChanges();
```

```
    // Запрос сотрудников
    var itEmployees = context.Employees
        .Where(e => e.Department == "IT")
        .ToList();

    foreach (var employee in itEmployees)
    {
        Console.WriteLine($"{employee.FirstName} {employee.LastName}");
    }
}
```

Управление базами данных в Visual Studio

SQL Server Object Explorer

- Просмотр структуры базы данных;
- Выполнение запросов;
- Управление таблицами и данными;

Server Explorer

- Подключение к различным источникам данных;
- Просмотр схемы базы данных;
- Редактирование данных;

Развертывание приложений с базой данных

Методы развертывания:

- LocalDB для разработки и тестирования;
- SQL Server Express для небольших приложений;
- Полноценный SQL Server для производственных сред;

Миграции баз данных:

Entity Framework Migrations:

```
Add-Migration InitialCreate

Update-Database
```

Контрольные вопросы

1. Какие современные СУБД вы знаете и в чем их особенности?
2. Какова архитектура Microsoft SQL Server?
3. Какие технологии доступа к данным используются в .NET?
4. Как организовать подключение к SQL Server из Visual Studio?
5. В чем разница между ADO.NET и Entity Framework?
6. Как развернуть приложение с базой данных?

Литература

1. Microsoft Documentation: SQL Server and Visual Studio
2. Entity Framework Documentation
3. ADO.NET Programming Guide
4. Коннолли Т., Бегг К. Базы данных: проектирование, реализация и сопровождение. - Глава 2.