Min første program der tilgår en database

Ide:

Du skal programmere en REST service til en tabel i Hotel databasen, samt programmere en REST Konsumer.

Baggrund:

A) Databasen

Du har i faget SWD lavet en række tabeller i en Hotel Database med følgende 4 tabeller:

HOTEL: (Hotel_No, Name, Address)
ROOM: (Room_No, Hotel_No, Types, Price)
BOOKING: (Booking_Id, Hotel_No, Guest_No, Date_From, Date_To, Room_No)
GUEST: (Guest_No, Name, Address)

(Primary keys are underlined) – det kan være du har indført et bookingId, der er primærnøgle, i stedet for en sammensat nøgle.

Det forventes du har lavet disse 4 tabeller i en lokal database, samt initialiseret dem med nogle værdier, samt kender til SQL sætninger som SELECT, INSERT, UPDATE og DELETE.

Se evt. https://www.w3schools.com/sql/

B) C# Database tilgang

- SqlConnection : https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlconnection?view=dotnet-plat-ext-5.0&viewFallbackFrom=netcore-3.1

- SqlCommand: https://docs.microsoft.com/en-us/dotnet/api/system.data.sqlclient.sqlcommand?view=dotnet-plat-ext-5.0&viewFallbackFrom=netcore-3.1

Opgaven:

NU er det tid til at lave programmet. Til det skal du have model-klasser, samt selve ConsoleApp'en.

Trin 1 Model klasser

Du skal lave et 'class Library' (dll-fil)til model klasserne.

Du skal lave et projekt i VS2019 **'Class Library (.Net Core)'**

Du skal implementere de 4 model klasser:

- `Hotel  properties: HotelNr, Navn, Adresse, Rooms`
- `Room  properties: RoomNr, RoomType, Pris`
- `Guest  properties: GuestNr, Navn, Adresse`
- `Booking  properties: BookingId, HotelNr, Room, Guest, FraDato, TilDato`

Husk at alle klassene skal være public, samt de skal have en **default konstruktør**.
Du må gerne tilføje (overload) endnu en konstruktør til initialisering. Typerne skal følge typerne i databasen.

Husk at køre 'build' af dit class library.

Trin 2 Console Applikation

Du skal lave et nyt projekt i din solution, som skal være en ConsoleApp (.Net Core). Du kommer senere til at lave det til en Razor-applikation.

Lav en klasse DBWorker med en Start-metode og kald denne fra program-main metoden.

I DBWorker klassen skal du lave 5 metoder:

```
private List<Hotel> GetAllHotels()

private Hotel GetHotelByNr(int no)

private bool AddHotel(Hotel hotel)

private bool UpdateHotel(int no, Hotel hotel)

private Hotel DeleteHotelByNr(int no)
```

Du skal implementere de fem metoder og kalde dem fra Start-metoden.

For SELECT
Den skabelon som alle select metoder (her 2 metoder) er opbygget af er som følger:

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
    SqlCommand command = new SqlCommand(queryString, connection);
    command.Connection.Open();

    SqlDataReader reader = command.ExecuteReader ();

    … mere kode f.eks. læs alle rækker fra database tabellen
}
```

ConnectionString findes under properties for databasen.

For INSERT, UPDATE og DELETE
Mens insert, update og delete (og create, alter osv) er som følger:
Den skabelon som alle 5 metoder er opbygget af er som følger:

// sql: "select * from hotel where Hotel_No = @no";

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
    SqlCommand command = new SqlCommand(sqlString, connection);
    command.Connection.Open();
    // indsæt parametre i sqlString
    // sql.Parameters.AddWithValue("@no", no);


    int rows = command.ExecuteNonQuery();


    … mere kode f.eks. find ud af om en række er udført dvs. row == 1
}
```

ConnectionString findes under properties for databasen.

Trin 3 ekstra implementer for Rooms og Bookings

Trin 4 ekstra Lav det mere rigtigt mapning til objekter

Hotel model klassen har en liste af rooms og en booking har et objekt af guest samt et af room.

Implementer at et hotel objekt også henter alle rooms, hhv. at booking henter et guest objekt samt et room objekt.