Advanced Programming C#

Lecture 11
part 2

dr inż. Małgorzata Janik

malgorzata.janik@pw.edu.pl

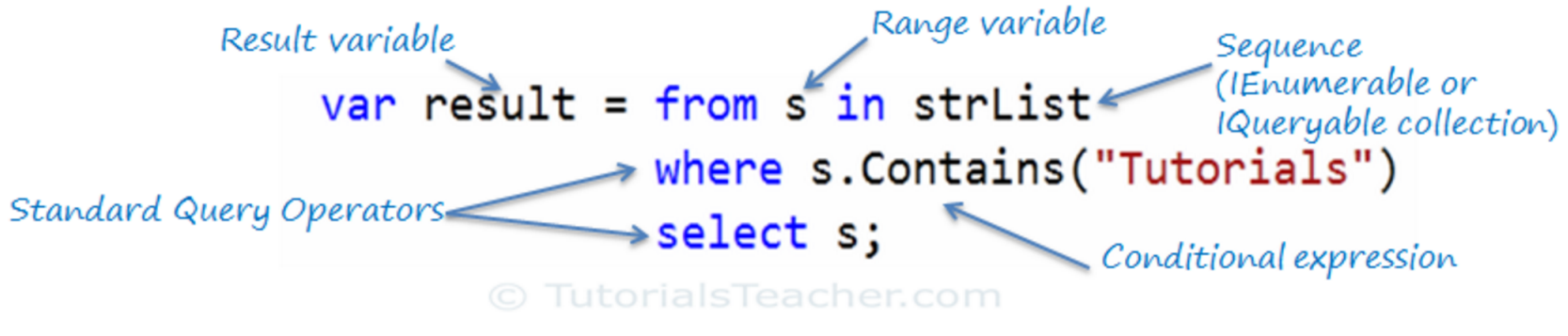
Winter Semester 2020/2021


WYDZIAŁ FIZYKI
POLITECHNIKA WARSZAWSKA
POLITECHNIKA WARSZAWSKA
WARSAW UNIVERSITY OF TECHNOLOGY

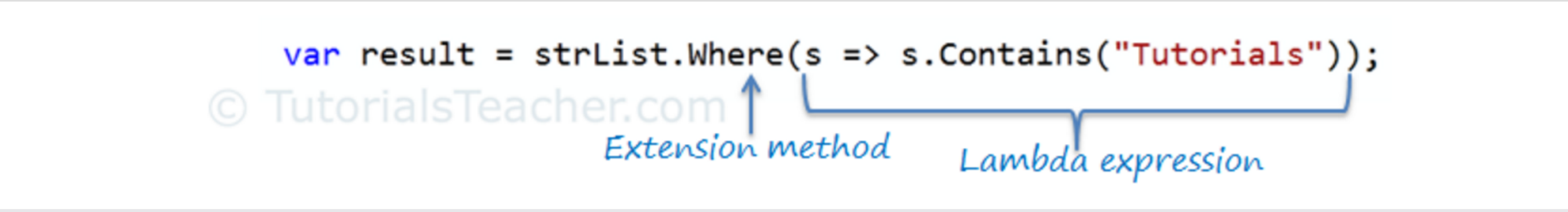
LINQ
(part 2)

LINQ

- Previous lecture...



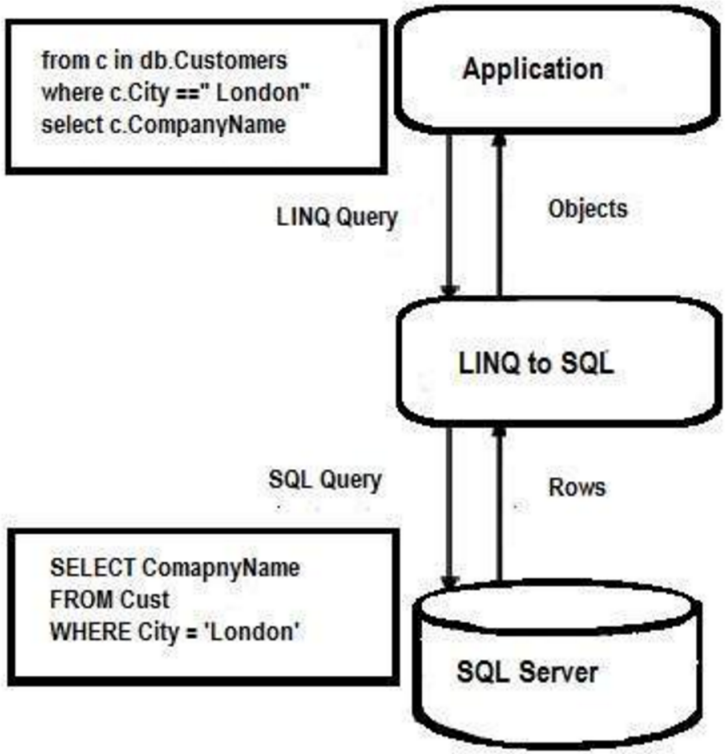
LINQ Query Syntax



LINQ Method Syntax Structure

LINQ to SQL

Architecture of LINQ to SQL.



https://www.tutorialspoint.com/linq/linq_sql.htm

LINQ to SQL

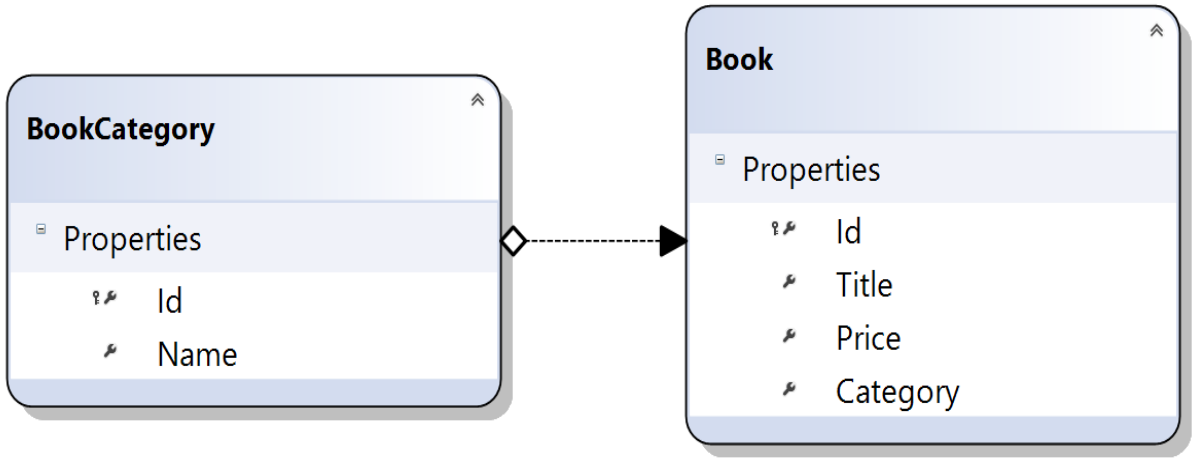
- DataContext
  - This class will handle connecting to the database and declaring each of the tables you'll be connecting to
- Entity Classes
  - Classes representing the SQL tables

https://www.tutorialspoint.com/linq/linq_sql.htm

https://msdn.microsoft.com/en-us/library/bb425822.aspx

https://www.codeproject.com/Articles/43025/A-LINQ-Tutorial-Mapping-Tables-to-Objects

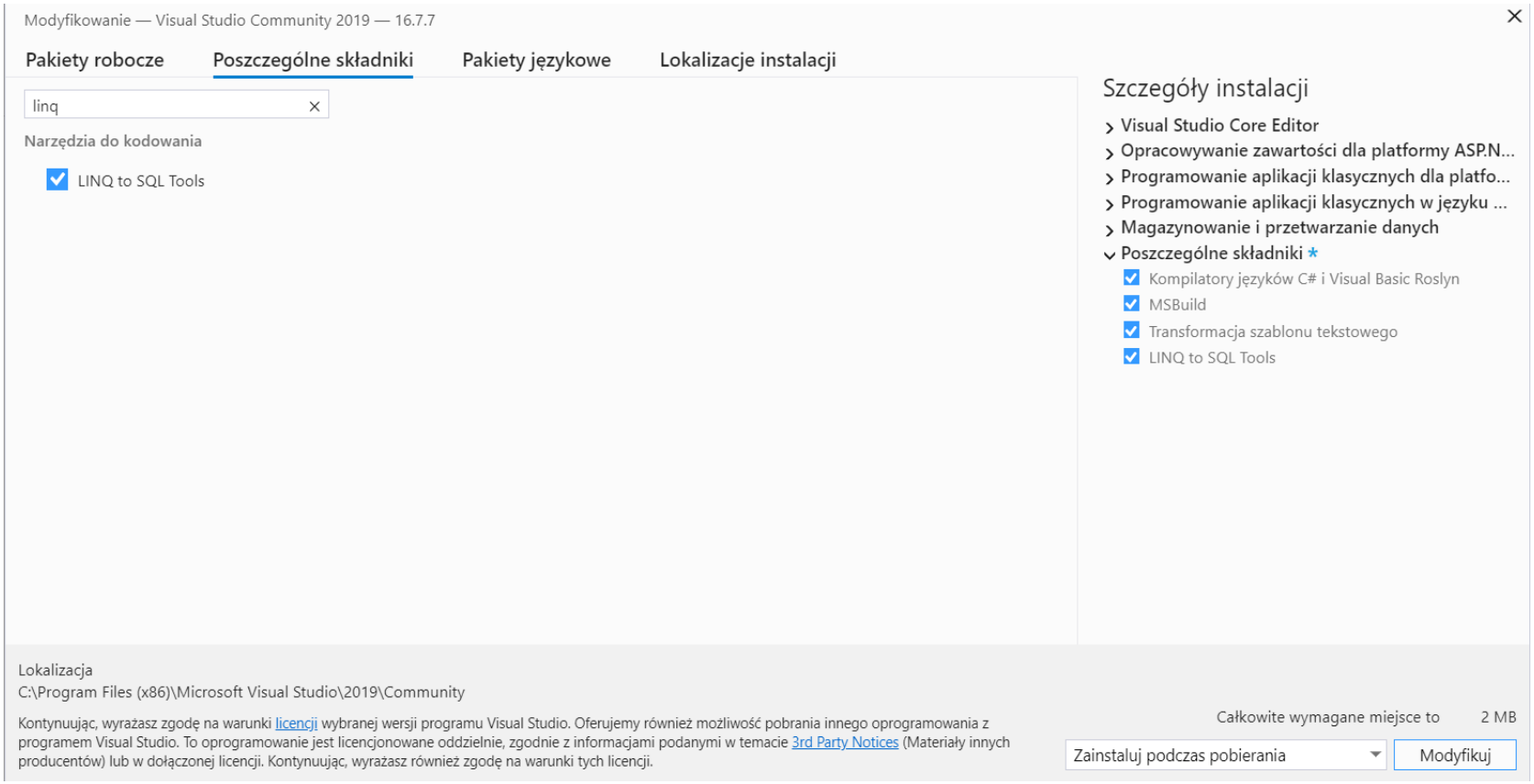
Download SQL database



www.if.pw.edu.pl/~majanik/data/Csharp/BookCatalog.mdf

Package needed in VS2019

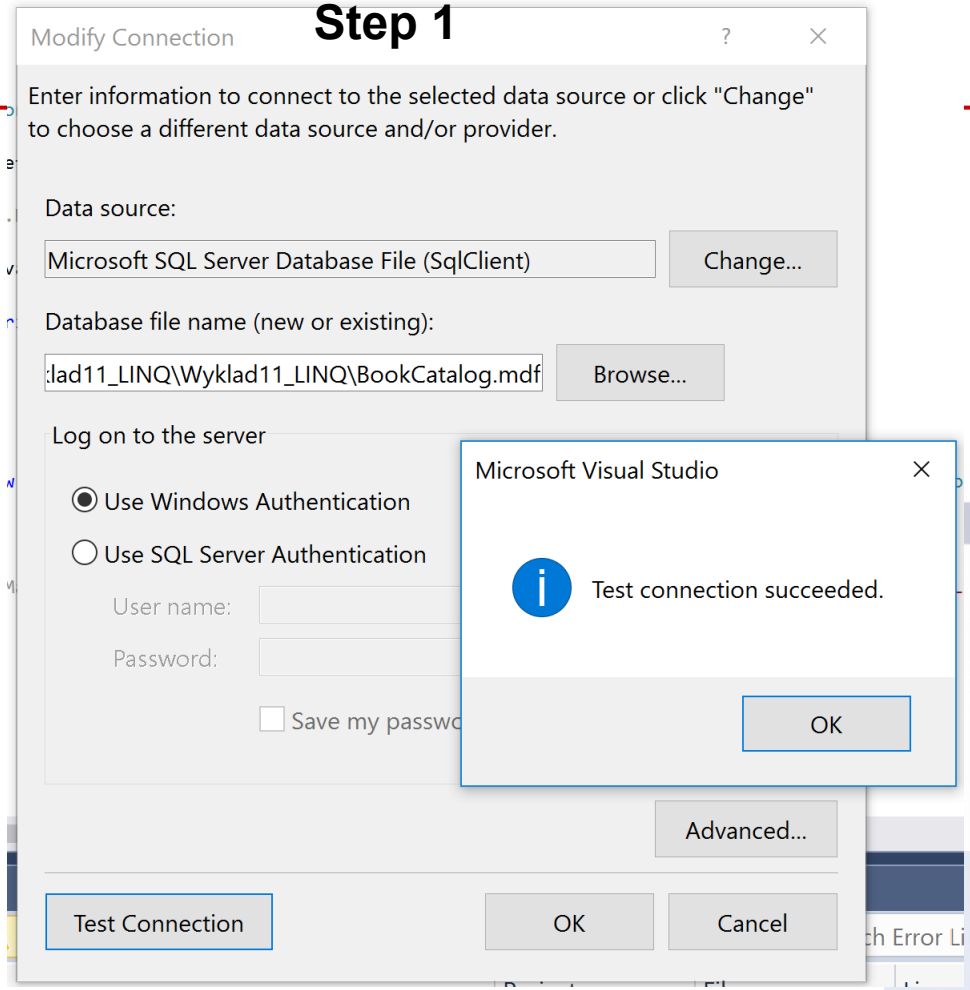
- Additional modificaton: **LINQ to SQL Tools**



How to Use LINQ to SQL?

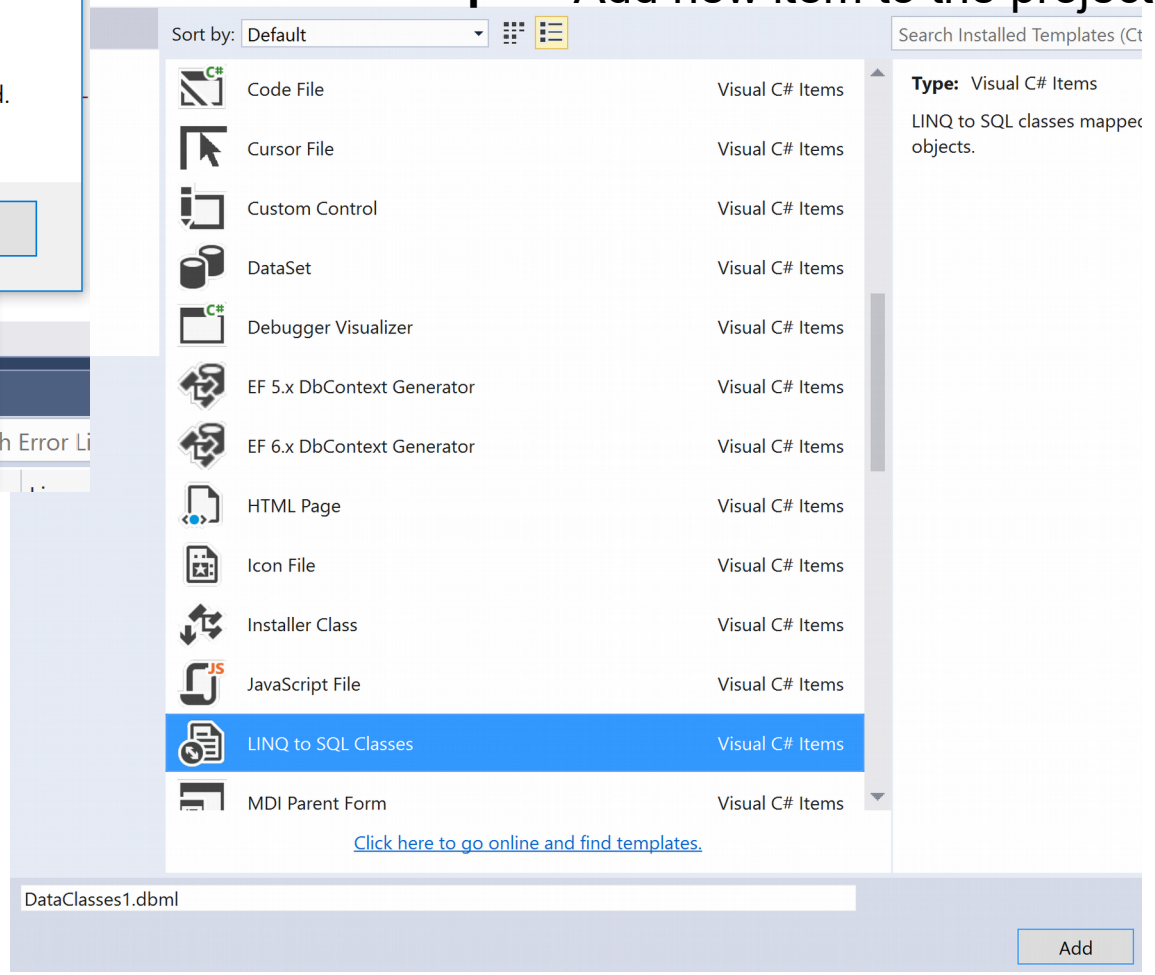
- Prerequisites: create new **Console Application Project**
- **Step 1**: Make a new “Data Connection” with database server.
  - *View → Server Explorer → Data Connections → Add Connection → Microsoft SQL Server Database File*
- **Step 2**: Add *LINQ To SQL classes* file.
- **Step 3**: Select tables from database and drag and drop into the new LINQ to SQL class file.

Step 1

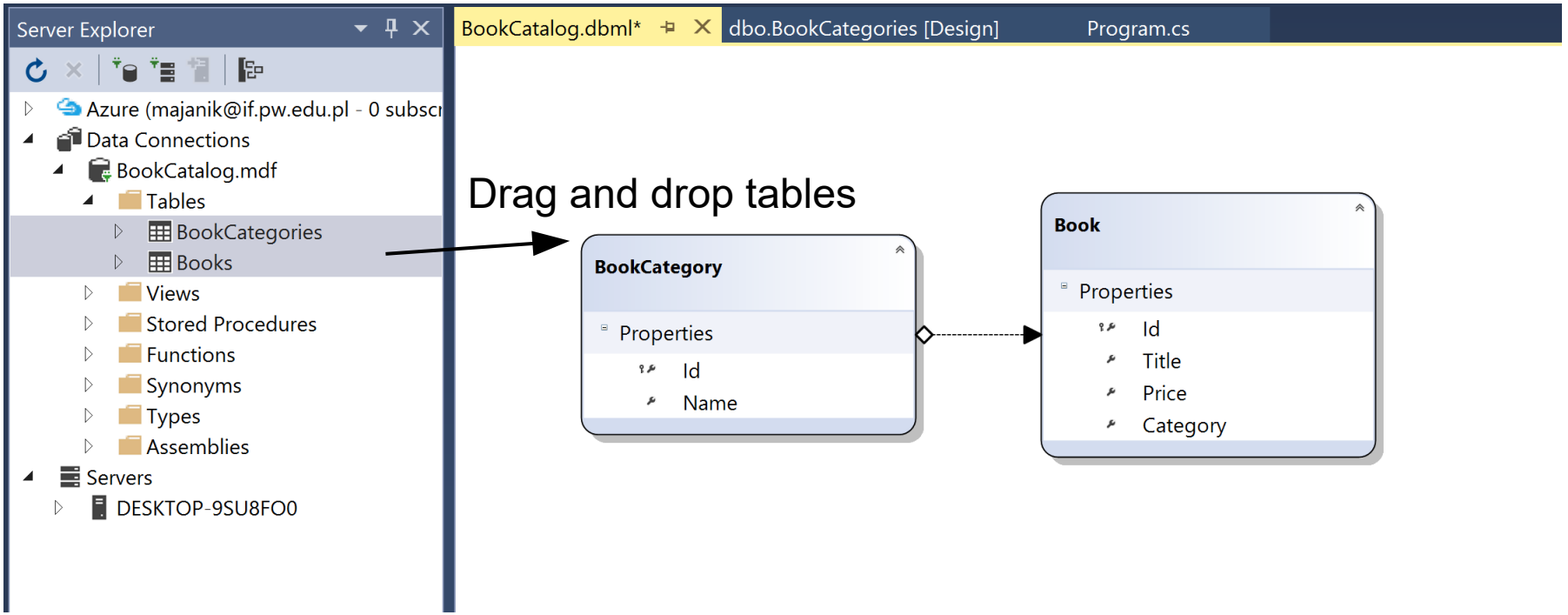


Create new connection.
Browse for the database file.

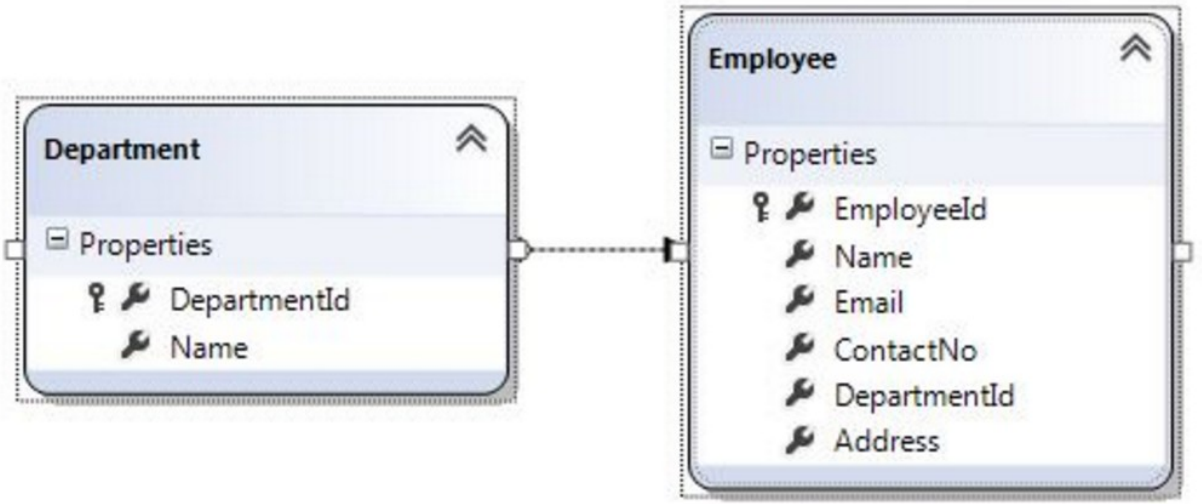
Step 2 Add new item to the project.



Step 3


Server Explorer
BookCatalog.dbml*
dbo.BookCategories [Design]
Program.cs
Azure (majanik@if.pw.edu.pl - 0 subscr
Data Connections
BookCatalog.mdf
Tables
BookCategories
Books
Views
Stored Procedures
Functions
Synonyms
Types
Assemblies
Servers
DESKTOP-9SU8FO0
Drag and drop tables
BookCategory
Properties
Id
Name
Book
Properties
Id
Title
Price
Category

All examples below for database:



Print table content

- Task 1: Print Books table

  To do that:

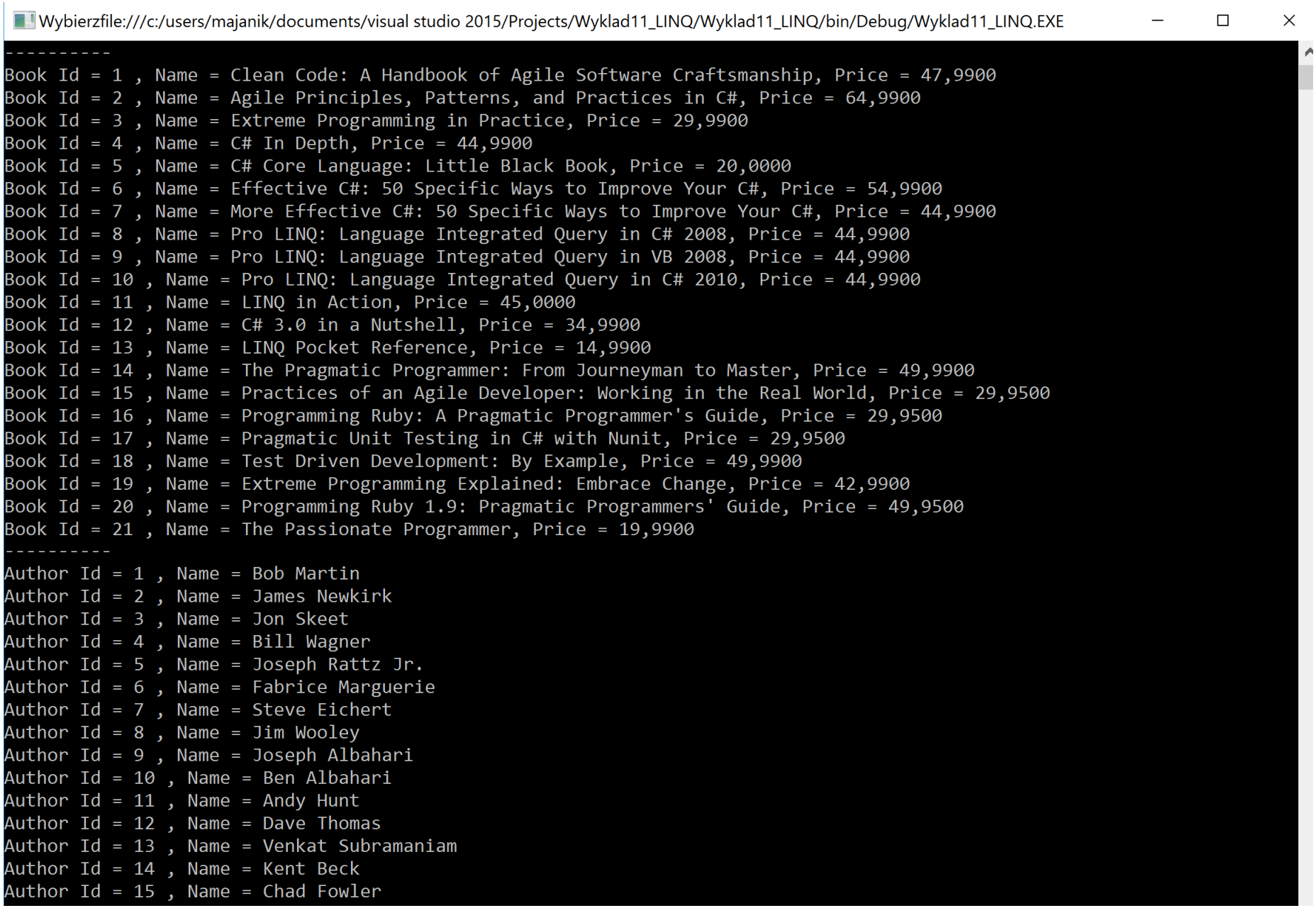
  - (1) Create DataContext:

```
LINQEmployeesDataContext db = new LINQEmployeesDataContext();
```

  - (2) use foreach statement to loop through the data and print all information from Books table

```
//Get All Employee from Database
        var employeeList = db.Employees;
        foreach (Employee employee in employeeList)
        {
           Console.WriteLine("Employee Id = {0} , Name = {1}, Email = {2}, ContactNo = {3}",
                  employee.EmployeeId, employee.Name, employee.Email, employee.ContactNo);
        }
```

Example output



Queries

- You do not need to Open / Close connection (LINQ to SQL does it for you)
- Create a query and execute it.
- Try first with SingleOrDefault(); command
- Example:

```
var query = from t in db.Department
            where t.DepartmentId == 2
            select t.Name;
var departmentName = query.SingleOrDefault();
```

Or:

```
var departmentName =
  db.Department.SingleOrDefault(t => t.DepartmentId == 2 )
```

In this case instead of the above query it is possible also to simply use `t.Department.Name`.
Still - try using SingleOrDefault query as a exercise.

Queries

- Whenever you use **SingleOrDefault**, you clearly state that the query should result in at most a single result. SingleOrDefault returns the only element of a sequence, or a default value if the sequence is empty; this method throws an exception if there is more than one element in the sequence.
- Task 2: Add to the previous listing information about the book category. Try both implementations: using query syntax and using lambda syntax. Use SingeOrDefault query to get category name.

Example output

```
Book Id = 1 , Name = Clean Code: A Handbook of Agile Software Craftsmanship, Price = 47,9900, Category = Programming Practices
Book Id = 2 , Name = Agile Principles, Patterns, and Practices in C#, Price = 64,9900, Category = Programming Practices
Book Id = 3 , Name = Extreme Programming in Practice, Price = 29,9900, Category = Programming Practices
Book Id = 4 , Name = C# In Depth, Price = 44,9900, Category = C#
Book Id = 5 , Name = C# Core Language: Little Black Book, Price = 20,0000, Category = C#
Book Id = 6 , Name = Effective C#: 50 Specific Ways to Improve Your C#, Price = 54,9900, Category = C#
Book Id = 7 , Name = More Effective C#: 50 Specific Ways to Improve Your C#, Price = 44,9900, Category = C#
Book Id = 8 , Name = Pro LINQ: Language Integrated Query in C# 2008, Price = 44,9900, Category = LINQ
Book Id = 9 , Name = Pro LINQ: Language Integrated Query in VB 2008, Price = 44,9900, Category = LINQ
Book Id = 10 , Name = Pro LINQ: Language Integrated Query in C# 2010, Price = 44,9900, Category = LINQ
Book Id = 11 , Name = LINQ in Action, Price = 45,0000, Category = LINQ
Book Id = 12 , Name = C# 3.0 in a Nutshell, Price = 34,9900, Category = C#
Book Id = 13 , Name = LINQ Pocket Reference, Price = 14,9900, Category = LINQ
Book Id = 14 , Name = The Pragmatic Programmer: From Journeyman to Master, Price = 49,9900, Category = Programming Practices
Book Id = 15 , Name = Practices of an Agile Developer: Working in the Real World, Price = 29,9500, Category = Programming Practices
Book Id = 16 , Name = Programming Ruby: A Pragmatic Programmer's Guide, Price = 29,9500, Category = Ruby
Book Id = 17 , Name = Pragmatic Unit Testing in C# with Nunit, Price = 29,9500, Category = Unit Testing
Book Id = 18 , Name = Test Driven Development: By Example, Price = 49,9900, Category = Unit Testing
Book Id = 19 , Name = Extreme Programming Explained: Embrace Change, Price = 42,9900, Category = Programming Practices
Book Id = 20 , Name = Programming Ruby 1.9: Pragmatic Programmers' Guide, Price = 49,9500, Category = Ruby
Book Id = 21 , Name = The Passionate Programmer, Price = 19,9900, Category = Programming Practices
```

Insert

- You can use LINQ to SQL to insert new data into database
- Use „InsertOnSubmit” and „SubmitChanges” methods (see next slide)
- Task 3: Add (in the program code) new book titled „C#. Praktyczny kurs” with price „49.00”, and category „C#”
- List the books & authors again to see the changes

```
namespace LINQtoSQL
{
 class LinqToSQLCRUD
 {
  static void Main(string[] args)
  {
   string connectString =
System.Configuration.ConfigurationManager.ConnectionStrings["LinqToSQLDBConnectionString"].ToStri
ng();

   LinqToSQLDataContext db = new LinqToSQLDataContext(connectString);

   //Create new Employee
   Employee newEmployee = new Employee();
   newEmployee.Name = "Michael";
   newEmployee.Email = "yourname@companyname.com";
   newEmployee.ContactNo = "343434343";
   newEmployee.DepartmentId = 3;
   newEmployee.Address = "Michael - USA";

   //Add new Employee to database
   db.Employees.InsertOnSubmit(newEmployee);

   //Save changes to Database.
   db.SubmitChanges();

   //Get new Inserted Employee
   Employee insertedEmployee = db.Employees.FirstOrDefault(e =>e.Name.Equals("Michael"));

   Console.WriteLine("Employee Id = {0} , Name = {1}, Email = {2}, ContactNo = {3}, Address =
{4}",
            insertedEmployee.EmployeeId, insertedEmployee.Name, insertedEmployee.Email,
            insertedEmployee.ContactNo, insertedEmployee.Address);

   Console.WriteLine("\nPress any key to continue.");
   Console.ReadKey();
  }
 }
```

```
----------
----------
Book Id = 1 , Name = Clean Code: A Handbook of Agile Software Craftsmanship, Price = 47,9900, Category = Programming Pra
ctices
Book Id = 2 , Name = Agile Principles, Patterns, and Practices in C#, Price = 64,9900, Category = Programming Practices
Book Id = 3 , Name = Extreme Programming in Practice, Price = 29,9900, Category = Programming Practices
Book Id = 4 , Name = C# In Depth, Price = 44,9900, Category = C#
Book Id = 5 , Name = C# Core Language: Little Black Book, Price = 20,0000, Category = C#
Book Id = 6 , Name = Effective C#: 50 Specific Ways to Improve Your C#, Price = 54,9900, Category = C#
Book Id = 7 , Name = More Effective C#: 50 Specific Ways to Improve Your C#, Price = 44,9900, Category = C#
Book Id = 8 , Name = Pro LINQ: Language Integrated Query in C# 2008, Price = 44,9900, Category = LINQ
Book Id = 9 , Name = Pro LINQ: Language Integrated Query in VB 2008, Price = 44,9900, Category = LINQ
Book Id = 10 , Name = Pro LINQ: Language Integrated Query in C# 2010, Price = 44,9900, Category = LINQ
Book Id = 11 , Name = LINQ in Action, Price = 45,0000, Category = LINQ
Book Id = 12 , Name = C# 3.0 in a Nutshell, Price = 34,9900, Category = C#
Book Id = 13 , Name = LINQ Pocket Reference, Price = 14,9900, Category = LINQ
Book Id = 14 , Name = The Pragmatic Programmer: From Journeyman to Master, Price = 49,9900, Category = Programming Pract
ices
Book Id = 15 , Name = Practices of an Agile Developer: Working in the Real World, Price = 29,9500, Category = Programmin
g Practices
Book Id = 16 , Name = Programming Ruby: A Pragmatic Programmer's Guide, Price = 29,9500, Category = Ruby
Book Id = 17 , Name = Pragmatic Unit Testing in C# with Nunit, Price = 29,9500, Category = Unit Testing
Book Id = 18 , Name = Test Driven Development: By Example, Price = 49,9900, Category = Unit Testing
Book Id = 19 , Name = Extreme Programming Explained: Embrace Change, Price = 42,9900, Category = Programming Practices
Book Id = 20 , Name = Programming Ruby 1.9: Pragmatic Programmers' Guide, Price = 49,9500, Category = Ruby
Book Id = 21 , Name = The Passionate Programmer, Price = 19,9900, Category = Programming Practices
Book Id = 32 , Name = C#. Praktyczny kurs, Price = 49, Category = C#
```

Update

- Updating rows in the database:
  - Get object from the database
  - Change its properties
  - Use SubmitChanges to apply the update
  - (See next slide for example)
- Task 4: Update price of the inserted book (49 → 39).
  Find it via its name: „C#. Praktyczny kurs”

```
using System;
using System.Linq;

namespace LINQtoSQL
{
 class LinqToSQLCRUD
 {
  static void Main(string[] args)
  {
   string connectString =
System.Configuration.ConfigurationManager.ConnectionStrings["LinqToSQLDBConnectionString"].T
oString();

   LinqToSQLDataContext db = new LinqToSQLDataContext(connectString);
   //Get Employee for update
   Employee employee = db.Employees.FirstOrDefault(e =>e.Name.Equals("Michael"));

   employee.Name = "George Michael";
   employee.Email = "yourname@companyname.com";
   employee.ContactNo = "99999999";
   employee.DepartmentId = 2;
   employee.Address = "Michael George - UK";

   //Save changes to Database.
   db.SubmitChanges();

   //Get Updated Employee
   Employee updatedEmployee = db.Employees.FirstOrDefault(e =>e.Name.Equals("George
Michael"));

   Console.WriteLine("Employee Id = {0} , Name = {1}, Email = {2}, ContactNo = {3}, Address
= {4}",
            updatedEmployee.EmployeeId, updatedEmployee.Name, updatedEmployee.Email,
            updatedEmployee.ContactNo, updatedEmployee.Address);

   Console.WriteLine("\nPress any key to continue.");
   Console.ReadKey();
```

Delete

- Similarly as for Update, first retreive object from the database
- To stage delete of an object use `DeleteOnSubmit(T)` function (see example on the next slide)
- Task 5: Delete all books with category „Ruby”

Notes:
- Use category name and not category ID!
- You may want to use Select (to get many books) instead of FirstOrDefault
- This task is not a copy-paste example & change names; needs some thinking

```
using System;
using System.Linq;

namespace LINQtoSQL
{
 class LinqToSQLCRUD
 {
  static void Main(string[] args)
  {
   string connectString =
System.Configuration.ConfigurationManager.ConnectionStrings["LinqToSQLDBConnectionString"]
.ToString();

   LinqToSQLDataContext db = newLinqToSQLDataContext(connectString);

   //Get Employee to Delete
   Employee deleteEmployee = db.Employees.FirstOrDefault(e =>e.Name.Equals("George
Michael"));

   //Delete Employee
   db.Employees.DeleteOnSubmit(deleteEmployee);

   //Save changes to Database.
   db.SubmitChanges();

   //Get All Employee from Database
   var employeeList = db.Employees;
   foreach (Employee employee in employeeList)
   {
    Console.WriteLine("Employee Id = {0} , Name = {1}, Email = {2}, ContactNo = {3}",
            employee.EmployeeId, employee.Name, employee.Email, employee.ContactNo);
   }
   Console.WriteLine("\nPress any key to continue.");
   Console.ReadKey();
  }
 }
}
```

References

- LINQ – SQL
  - https://www.tutorialspoint.com/linq/linq_sql.htm
- LINQ Tutorial
  - https://www.tutorialspoint.com/linq/
- LINQ Tutorials
  - http://www.tutorialsteacher.com/linq/linq-tutorials
- Mapping tables to objects:
  - https://www.codeproject.com/Articles/43025/A-LINQ-Tutorial-Mapping-Tables-to-Objects

THE END

dr inż. Małgorzata Janik
malgorzata.janik@pw.edu.pl