Лекция № 5

I T E A

IT Education Academy

WWW.ITEA.UA

C# Base

Урок № 5

Entity_Framework.

План урока

- Общее представление о NuGet
- Подключение Nuget-пакетов к проекту
- Подключение к базе данных
- Основы работы с БД
- Подключение БД к проекту
- Создание миграций
- LINQ to SQL

Общее представление о NuGet

- NuGet - это расширение для Visual Studio, которое позволяет быстро и просто добавлять, удалять и обновлять сторонние библиотеки и инструменты для проектов, использующих .NET Framework.

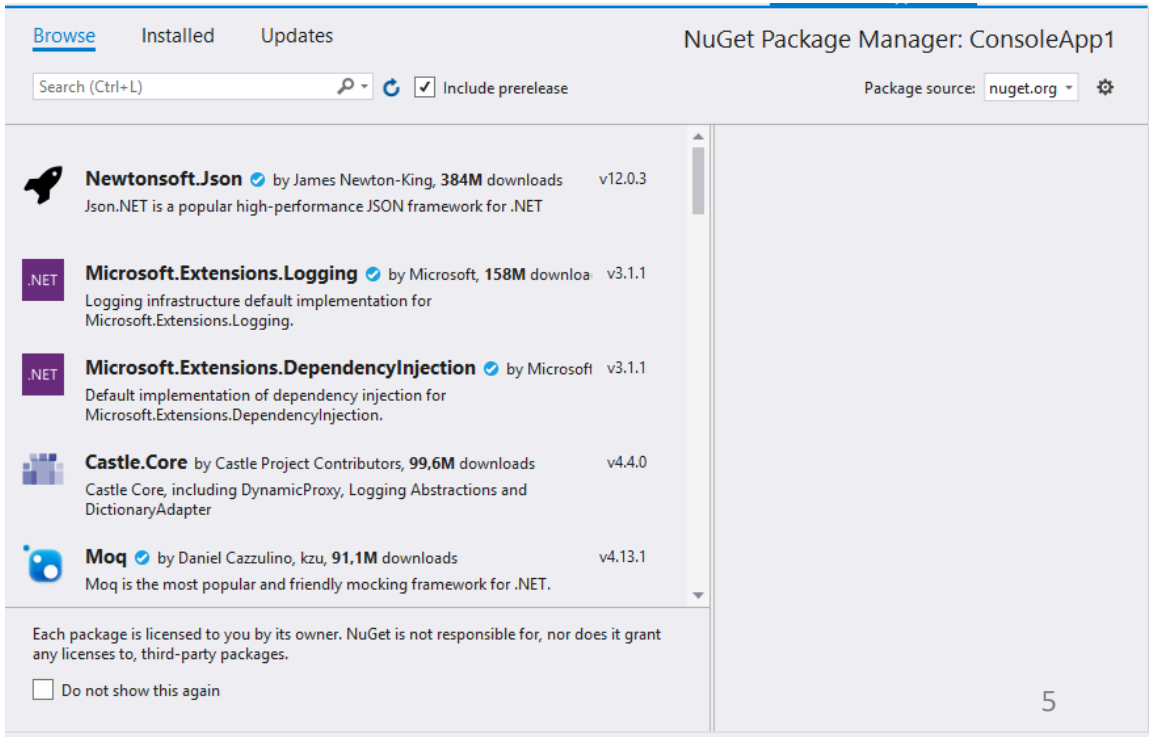
- Допустим, Вы сами разработали библиотеку и хотите предоставить ее другим разработчикам. Для этого нужно создать пакет NuGet и сохранить его в хранилище NuGet. Если Вы хотите использовать библиотеку или инструмент, созданные другими разработчиками, вы скачиваете пакет из хранилища и устанавливаете его в свой проект в Visual Studio.

- Во время установки пакета, NuGet копирует файлы в проект и автоматически делает необходимые изменения, такие как добавление ссылок и изменение файлов конфигурации app.config. Если вы решили удалить библиотеку, NuGet удаляет установленные файлы, а также отменяет изменения сделанные при установке, таким образом не остается никакого мусора и беспорядка

Использование NuGet в Visual Studio

- NuGet работает во всех версиях Visual Studio 2012. Найти, установить, удалить или обновить пакеты можно в диалоговом окне Manage Nuget Packages или через командную строку PowerShell в Package Manager Console. Все это встроено в Visual Studio
- и доступно через главное меню и либо через
- контекстное меню в Solution Explorer

- На картинке показан диалог управления пакетами NuGet. Вкладка Online показывает все доступные пакеты на официальном фиде.

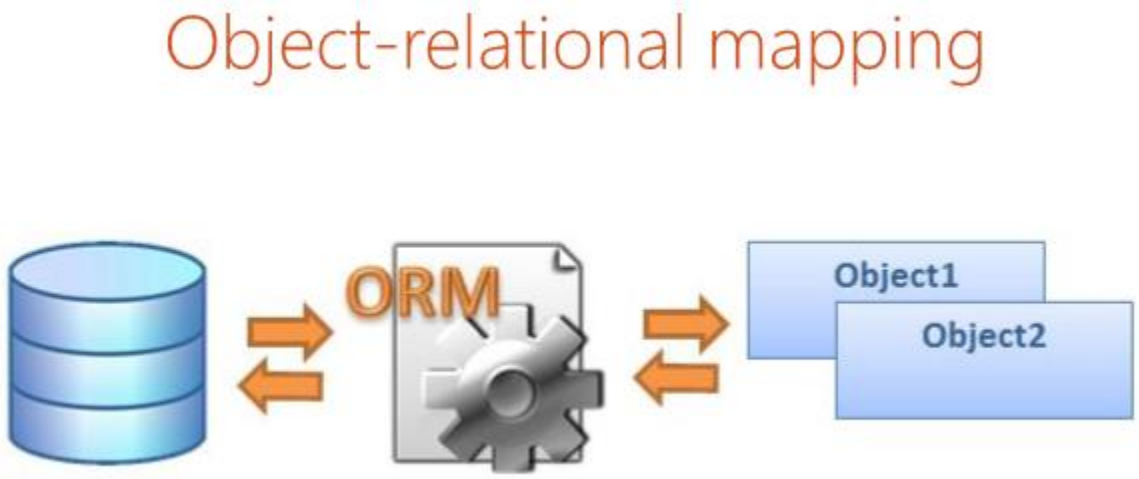


Entity Framework

- **Entity Framework** представляет специальную объектно-ориентированную технологию на базе фреймворка .NET для работы с данными. Если традиционные средства ADO.NET позволяют создавать подключения, команды и прочие объекты для взаимодействия с базами данных, то Entity Framework представляет собой более высокий уровень абстракции, который позволяет абстрагироваться от самой базы данных и работать с данными независимо от типа хранилища. Если на физическом уровне мы оперируем таблицами, индексами, первичными и внешними ключами, но на концептуальном уровне, который нам предлагает Entity Framework, мы уже работает с объектами.
- Центральной концепцией **Entity Framework** является понятие сущности или **entity**. Сущность представляет набор данных, ассоциированных с определенным объектом. Поэтому данная технология предполагает работу не с таблицами, а с объектами и их наборами.

Object-relational mapping

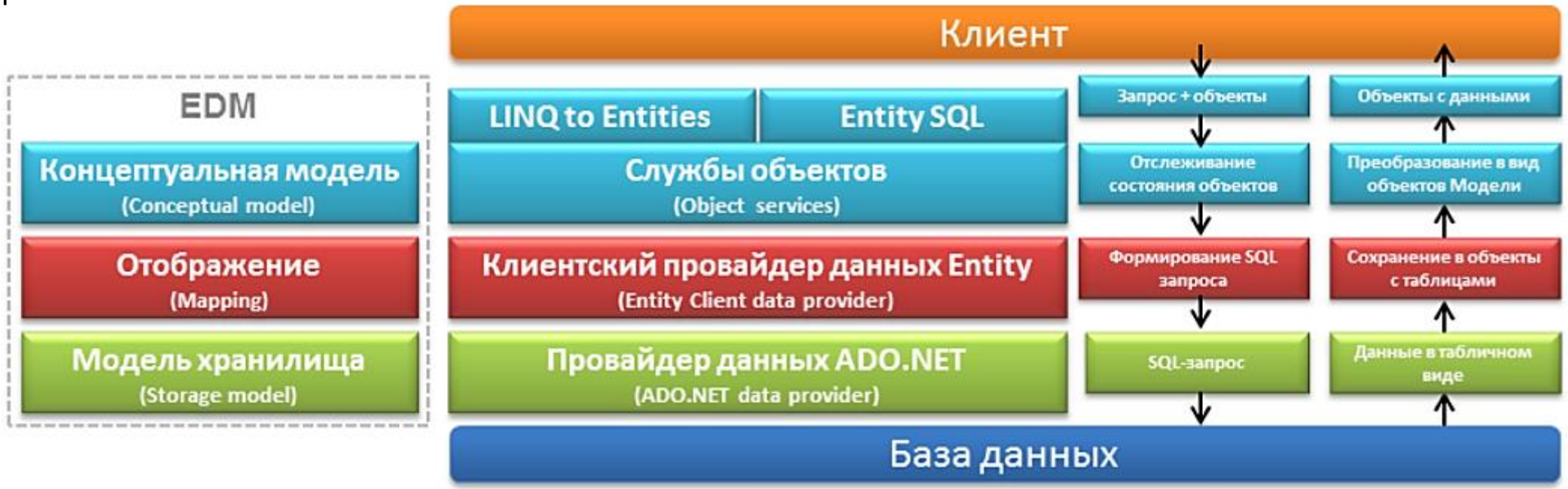
- **object-relational mapping (ORM)** решением для .NET Framework от Microsoft. Предоставляет возможность взаимодействия с объектами как посредством LINQ
- Объектно-реляционное отображение (**ORM**) технология программирования, для автоматического сопоставления и преобразования данных между реляционными СУБД и объектами из мира ООП.



Entity Framework

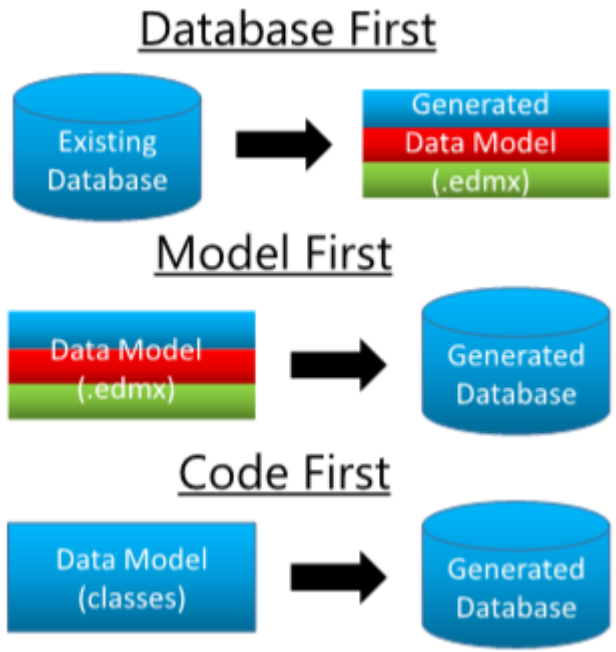
Архитектура Entity Framework

EF -набор технологий ADO.NET, обеспечивающих разработку приложений, связанных с обработкой данных.



Обзор EDM (Entity Data Model)

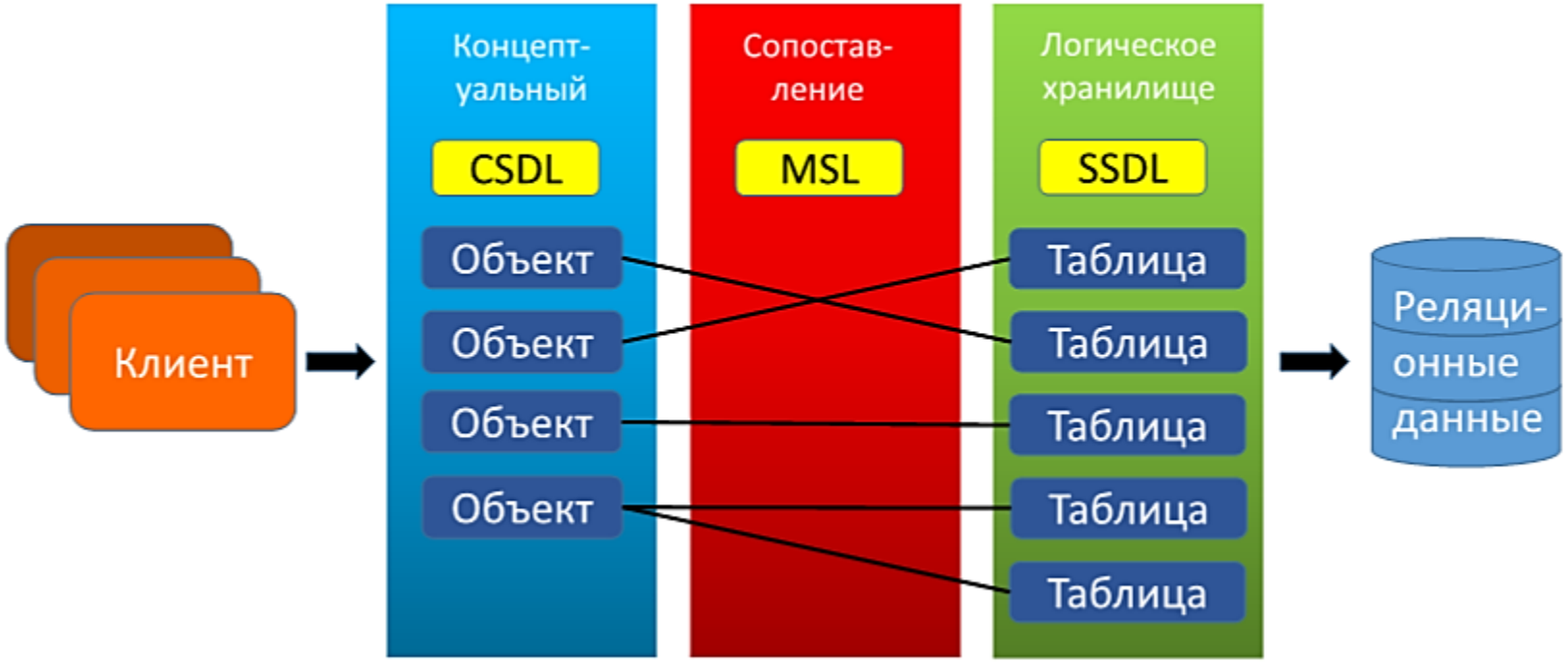
- **Entity Data Model (EDM)** – модель сущностей или концептуальная модель между объектной моделью и БД, согласно которой определяются правила соответствия объектов элементам базы данных.
- **Модель EDM** – это наборосновных понятий,которые описывают структуру данных не зависимо от формы хранения. Модель EDM решает проблемы, возникающие из необходимости хранить данные в различных формах.
- Создать эту модель можно несколькими способами:**«Database First», «ModelFirst», «CodeFirst».**



Обзор EDM (Entity Data Model)

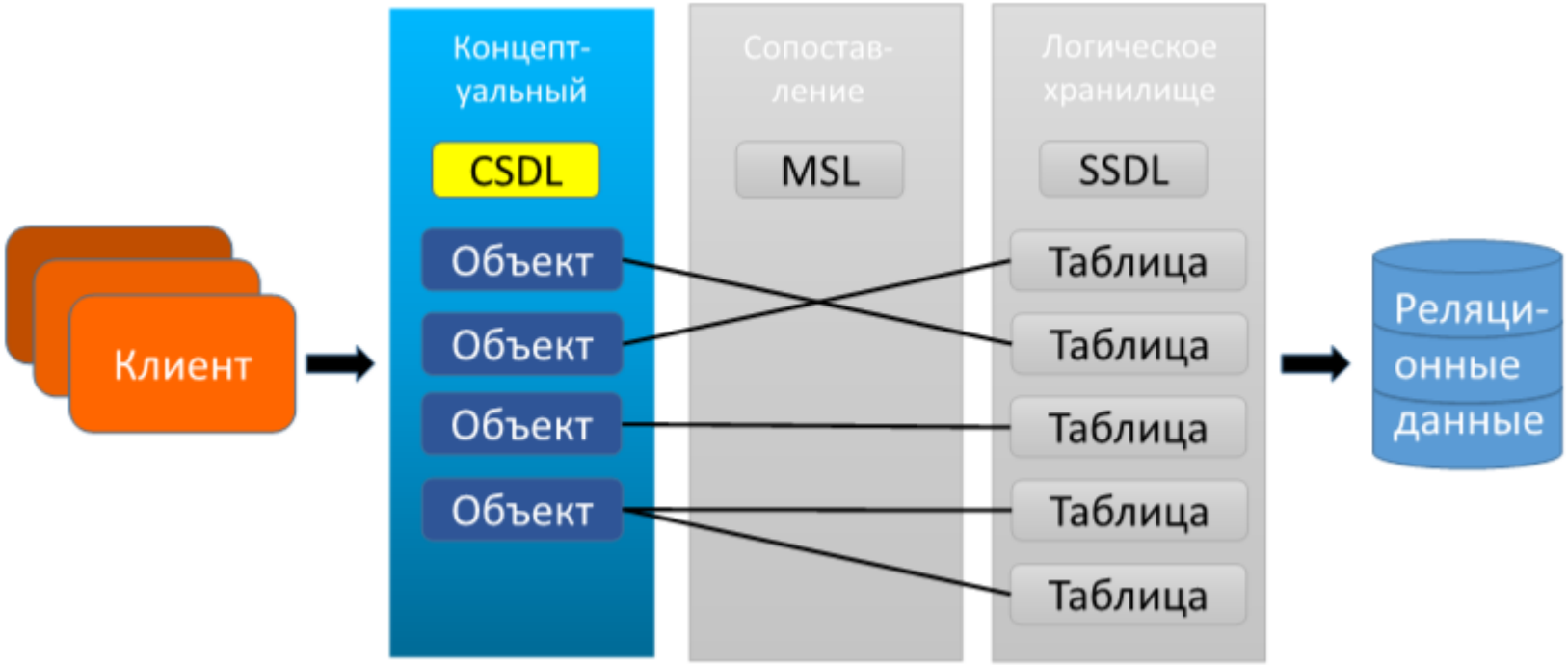
Модель EDM состоит из трёх слоев :

Концептуальный слой.(CSDL) |Слой сопоставления.(MSL) | Логический слой.(SSDL)



Обзор EDM (Entity Data Model)

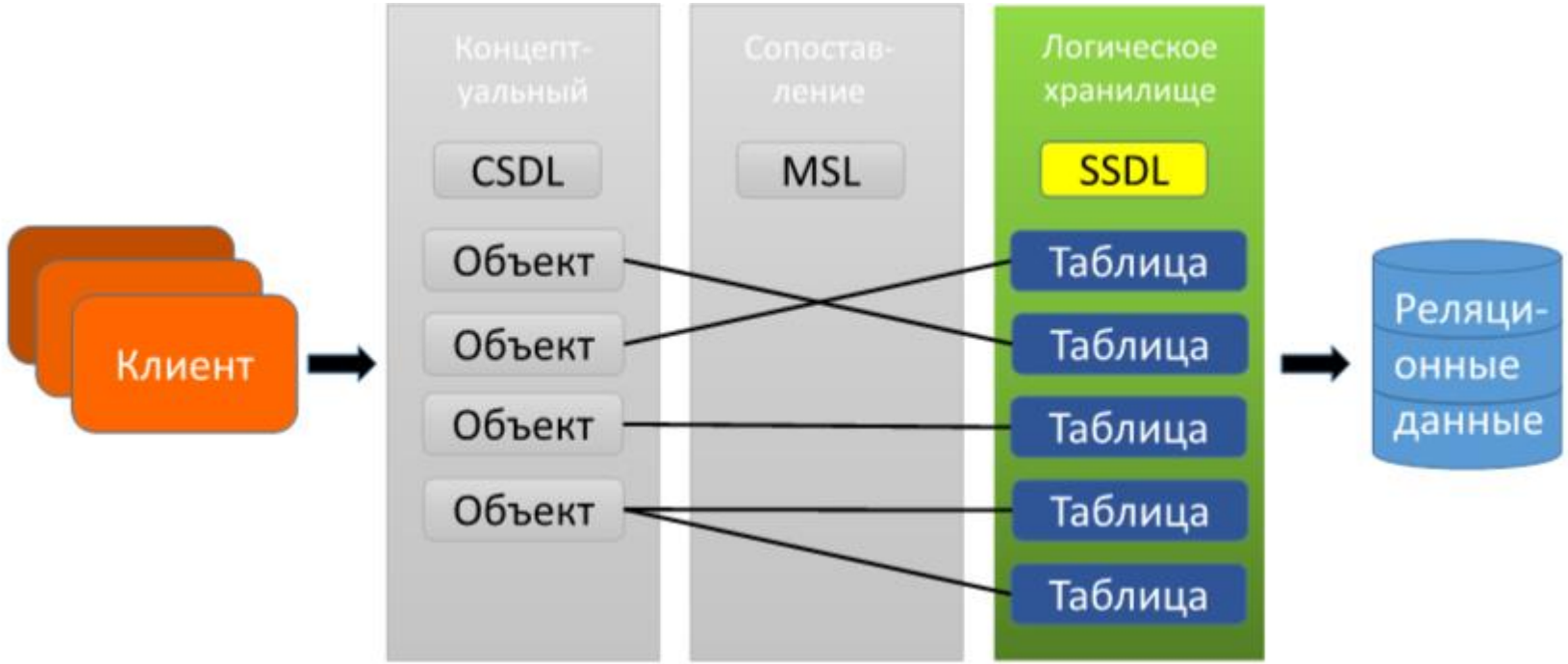
Conceptual Schema Definition Language (CSDL)



Язык CSDL —это язык на основе XML, описывающий сущности, связи и функции, составляющие концептуальную модель управляемого данными приложения.

Обзор EDM (Entity Data Model)

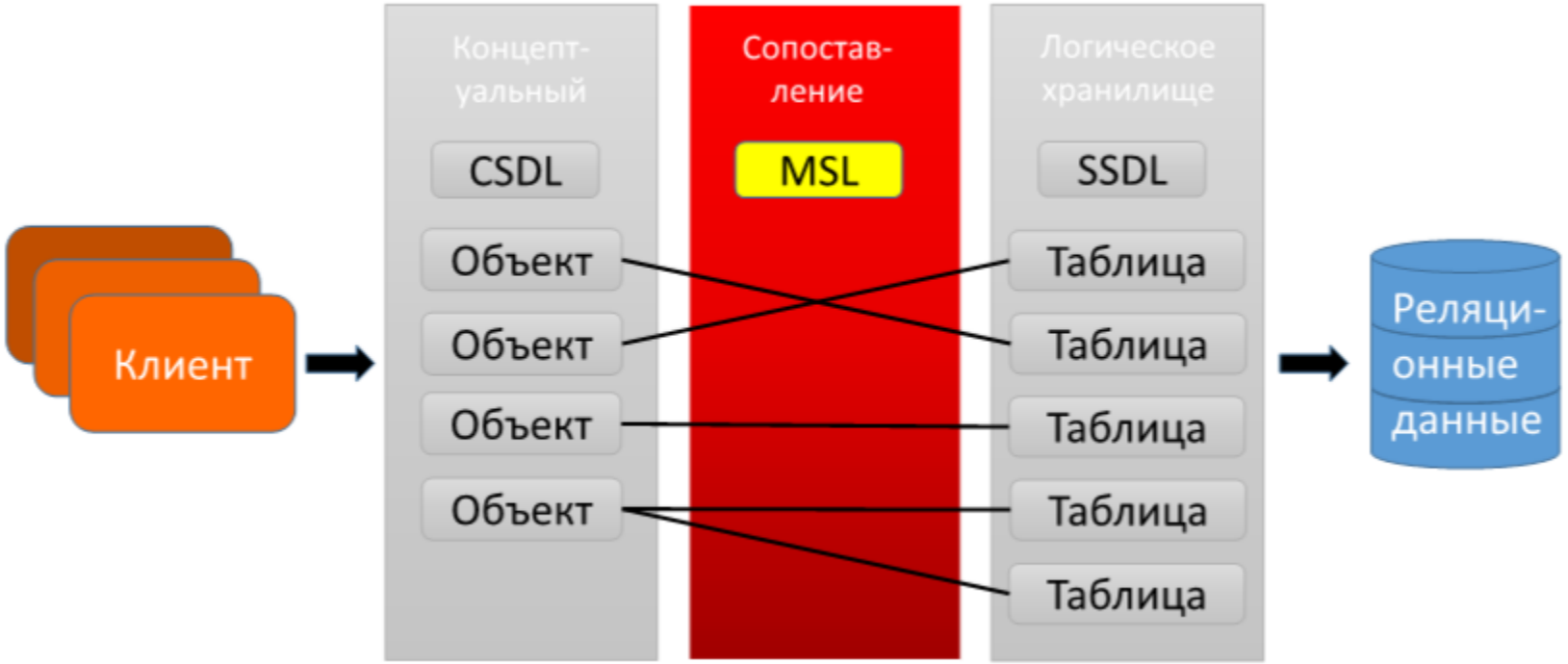
Storage Schema Definition Language SSDL



SSDL (логический слой) - представляет собой язык на основе XML, на котором описывается модель хранения в приложениях Entity Framework

Обзор EDM (Entity Data Model)

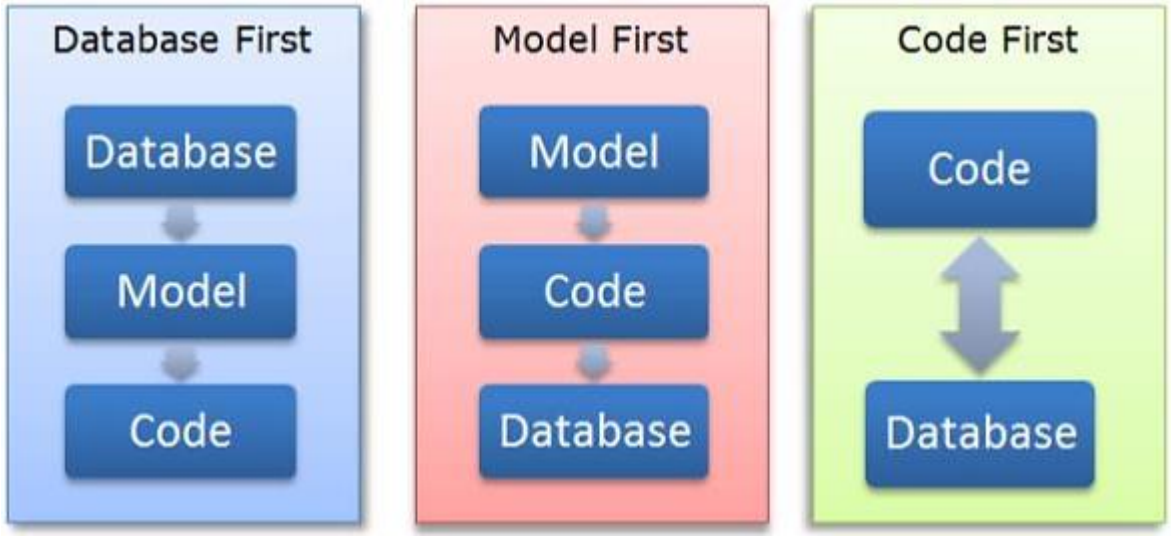
Mapping Schema Language (MSL)



Язык MSL —это язык на основе XML, описывающий сопоставление между концептуальной моделью и моделью хранения приложения Entity Framework

Способы создания моделей

- **Database first**: Entity Framework создает набор классов, которые отражают модель конкретной базы данных
- **Model first**: сначала разработчик создает модель базы данных, по которой затем Entity Framework создает реальную базу данных на сервере.
- **Code first**: разработчик создает класс модели данных, которые будут храниться в бд, а затем Entity Framework по этой модели генерирует базу данных и ее таблицы.



Основные операции для работы с данными

Большинство операций с данными представляют собой **CRUD-операции** (Create, Read, Update, Delete), то есть получение данных, создание, обновление и удаление. Entity Framework позволяет легко производить данные операции.

- **Create** – Добавление, для добавления применяется метод Add() у объекта DbSet.
- **Read** – Чтение.
- **Update** – Редактирование, контекст данных способен отслеживать изменения объектов, поэтому для редактирования объекта достаточно изменить его свойства и после этого вызвать метод SaveChanges().
- **Delete -** Удаление, для удаления объекта применяется метод Remove() объекта DbSet.

Связи отношений

Существует три вида связей:

- связь «Один-к-одному».
- связь «Один-ко-многим».
- связь «многие-ко-многим».

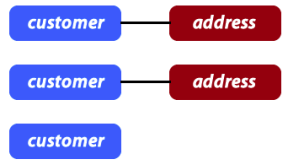
Одной из целей создания хорошей структуры базы данных является устранение избыточности (повторения) данных. Для этого нужно распределить данные по нескольким отдельным тематически организованным таблицам, чтобы каждый факт был представлен один раз.

Связи отношения "один к одному"

При установлении связи **"один к одному"** каждой строке таблицы А может соответствовать только одна строка таблицы Б и наоборот. Связь "один к одному" создается в том случае, когда оба связанные столбца являются первичными ключами или на них наложены ограничения уникальности. Такого рода отношения не очень распространены.

Addresses (информацию об адресе клиента)	
addresses_id	address
10	Проспект науки 1
11	Оболонский проспект 10

Customers (таблица для клиентов)		
customer_id	customer_name	addresses_id
20	Petrov Ivan	10
21	Ivanov Alexs	11



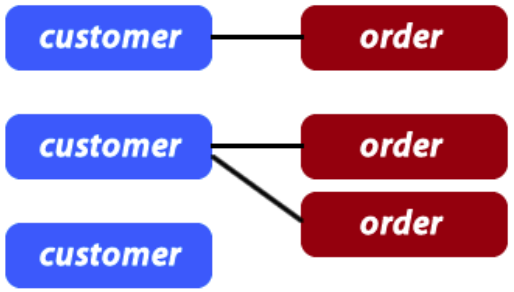
В таблице Customers есть поле с именем «address_id», которое ссылается на запись соответствия в таблице Address. Если каждый адрес может принадлежать только одному клиенту, это отношение **«Один к одному».**

Связи отношения "один к многим"

Связь "один ко многим" - наиболее распространенный вид связи. При такой связи каждой строке таблицы А может соответствовать множество строк таблицы Б, однако каждой строке таблицы Б может соответствовать только одна строка таблицы А.

Customers (таблица для клиентов)	
customer_id	customer_name
20	Petrov Ivan
21	Ivanov Alexs

Orders (таблица заказов)		
order_id	customer_id	amount
125	20	100
126	21	98
217	21	15



Например клиенты могут делать много заказов и заказы могут содержать много позиций - необходимо создать отношения «один ко многим». У каждого клиента может быть ноль, один или несколько заказов. Но заказ может принадлежать только одному клиент

Связи отношения "многие ко многим"

При установлении связи "многие ко многим" каждой строке таблицы А может соответствовать множество строк таблицы Б и наоборот. Такая связь создается при помощи третьей таблицы, называемой соединительной, первичный ключ которой состоит из внешних ключей, связанных с таблицами А и Б.

Orders (таблица заказов)		
order_id	customer_id	amount
125	20	100
126	21	98
217	21	15

Items	
items_id	item_name
200	Хочу заказ сегодня
201	Перезвонить

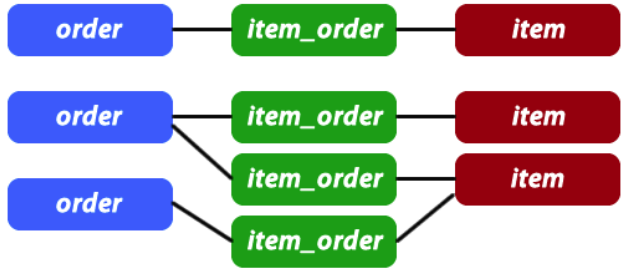


Таблица Items_Orders имеет только одну цель, а именно, чтобы создать отношение «многие ко многим» между элементами и заказами.

Items_Orders	
order_id	items_id
125	200
217	200

LINQ to Entities

- **LINQ to Entities** обеспечивает поддержку LINQ при запросах к сущностям.
- Компонент позволяет разработчикам писать запросы к концептуальной модели Entity Framework на языке Visual Basic или Visual C#.
- Запросы к платформе Entity Framework представляются в виде дерева команд запроса, выполняемого на контексте объектов.
- **Технология LINQ to Entities** преобразует запросы Language-Integrated Queries (LINQ) в запросы в виде дерева команд, выполняет эти запросы на платформе Entity Framework и возвращает объекты, которые могут использоваться как платформой Entity Framework, так и технологией LINQ.

Методы LINQ to Entities

- Where()
- Find ()
- First () / FirstOrDefault ()
- Include ()
- Select ()
- OrderBy ()
- OrderByDescending ()
- ThenBy ()/ThenByDescending()
- Join ()
- GroupBy ()
- Union()
- Intersect()
- Except()
- Count ()
- Min(),Max()иAverage()
- Sum()

IEnumerable и IQueryable

Методы расширений LINQ могут возвращать два объекта: **IEnumerable** и **IQueryable**. С одной стороны, интерфейс **IQueryable** наследуется от **IEnumerable**, поэтому по идее объект **IQueryable** это и есть также объект **IEnumerable**. Но реальность несколько сложнее. Между объектами этих интерфейсов есть разница в плане функциональности, поэтому они не взаимозаменяемы.

Интерфейс **IEnumerable** находится в пространстве имен **System.Collections.**

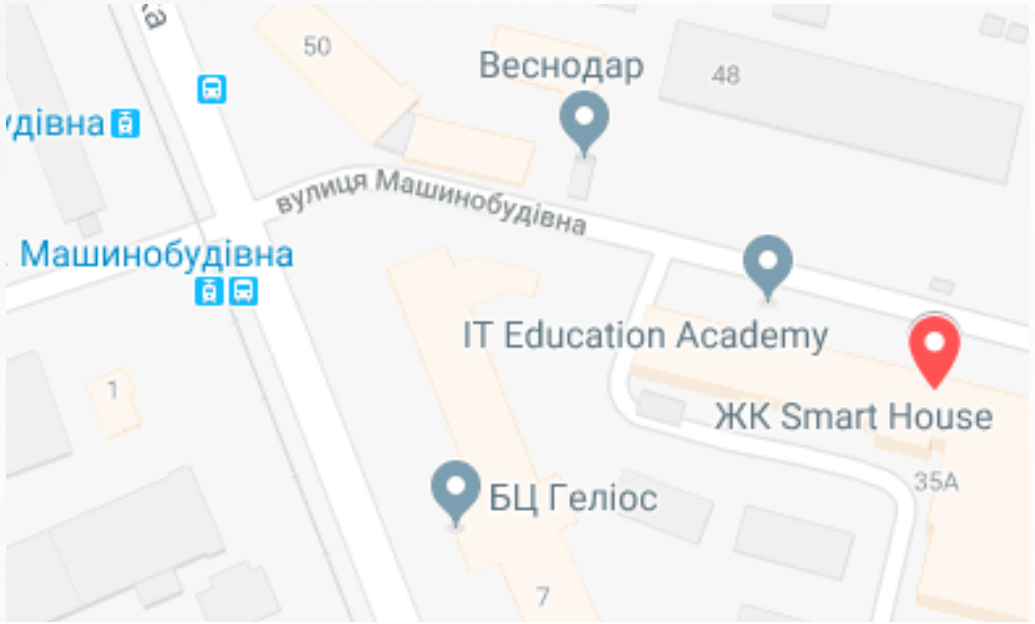
Интерфейс **IQueryable** располагается в пространстве имен **System.Linq.**

IEnumerable и IQueryable

IEnumerable представляет набор данных в памяти и может перемещаться по этим данным только вперед. Запрос, представленный объектом **IEnumerable**, выполняется немедленно и полностью, поэтому получение данных приложением происходит быстро.
При выполнении запроса **IEnumerable** загружает все данные, и если нам надо выполнить их фильтрацию, то сама фильтрация происходит на стороне клиента.

Интерфейс **IQueryable** предоставляет удаленный доступ к базе данных и позволяет перемещаться по данным как в прямом порядке от начала до конца, так и в обратном порядке. В процессе создания запроса, возвращаемым объектом которого является **IQueryable**, происходит оптимизация запроса. В итоге в процессе его выполнения тратится меньше памяти, меньше пропускной способности сети, но в то же время он может обрабатываться чуть медленнее, чем запрос, возвращающий объект **IEnumerable**.


Презентация окончена.
Спасибо за внимание!



I T E A

КОНТАКТНЫЕ ДАННЫЕ

ITEA

ЖК “Smart House”, ул. Машиностроительная, 41 (м.Берестейская)

ЖК «Корона» улица Срибнокильская,1 м. Позняки

+38 (044) 599-01-79
facebook.com/Itea
info@itea.ua
itea.ua