# IEnumerable и IQueryable в Entity Framework

**Последнее обновление: 31.10.2015**

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**. Объект 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 располагается в пространстве имен **System.Linq**. Объект 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.

Для примера возьмем два вроде бы идентичных выражения. Объект IEnumerable:

```
IEnumerable<Phone> phoneIEnum = db.Phones;
phoneIEnum=phoneIEnum.Where(p => p.Id > id);
```

Здесь запрос будет иметь следующий вид:

```
SELECT
        [Extent1].[Id] AS [Id],
        [Extent1].[Name] AS [Name],
        [Extent1].[Company] AS [Company]
        FROM [dbo].[Phones] AS [Extent1]
```

Фильтрация результата, обозначенная с помощью метода `Where(p => p.Id > id)` будет идти уже после выборки из бд в самом приложении.

Чтобы совместить фильтры, нам надо было сразу применить метод Where: `db.Phones.Where(p => p.Id > id);`

Объект IQueryable:

```
IQueryable<Phone> phoneIQuer = db.Phones;
phoneIQuer=phoneIQuer.Where(p => p.Id > id);
```

Здесь запрос будет иметь следующий вид:

```
SELECT
        [Extent1].[Id] AS [Id],
        [Extent1].[Name] AS [Name],
        [Extent1].[Company] AS [Company]
```

```
        FROM [dbo].[Phones] AS [Extent1]
        WHERE [Extent1].[Id] >3
```

Таким образом, все методы суммируются, запрос оптимизируется, и только потом происходит выборка из базы данных.

Что же лучше использовать? Все зависит от конкретной ситуации. Если разработчику нужен весь набор возвращаемых данных, то лучше использовать IEnumerable, предоставляющий максимальную скорость. Если же нам не нужен весь набор, а то только некоторые отфильтрованные данные, то лучше применять IQueryable.

**Назад** **Содержание** **Вперед**