EF

Всегда используйте профайлер !

Позволяет обнаружить проблемные места на ранней стадии

- Неоптимальный запрос (нужно посмотреть *execution plan*)
- Повторные вызовы одного и того же запроса
- Неправильное использование Lazy Loading
- т.д.

Использование *SaveChanges*

- Стараемся вызывать один раз *
- Unit Of Work паттерн

* не железное правило, каждый раз думаем

Bulk Operations – боль EF

- Сторонние библиотки (платные)
- Stored procs
- Raw SQL
- SqlBulkCopy

- Add vs AddRange
- AutoDetectChangesEnabled

AutoDetectChangesEnabled

```
using (var context = new BloggingContext())
{
    try
    {
        context.Configuration.AutoDetectChangesEnabled = false;

        // Make many calls in a loop
        foreach (var blog in aLotOfBlogs)
        {
            context.Blogs.Add(blog);
        }
    }
    finally
    {
        context.Configuration.AutoDetectChangesEnabled = true;
    }
}
```

Bulk Operations – DB side

```
private async Task SaveAutoFilledOnesAsync(List<Ones> autoFilledOnes)
{
    int counter = 0;
    do
    {
        var batchOnes = autoFilledOnes
            .Skip(counter * BatchSize)
            .Take(BatchSize)
            .ToList();

        if (batchOnes.Any())
        {
            _dashDbContext.Ones.AddRange(batchOnes);
            await _dashDbContext.SaveChangesAsync();
            counter++;
        }
        else
        {
            break;
        }
    } while (true);
}
```

Lazy Loading – Проблема N+1 запросов

```
var shipments = _shipmentsProvider.GetShipmentsQueryable().ToList();
var test = shipments.Where(s => s.Services.Any()).ToList();
```

```
public class Shipment
{
    public int Id { get; set; }
    public ICollection<Service> Services { get; set; }
}

public class Service
{
    public int Id { get; set; }
    public string Description { get; set; }
    public int ShipmentId { get; set; }
}
```

- Include
- Перестроить запрос

LINQ: отложенное выполнение (не связано с EF **изучить самостоятельно**)

Быстрое чтение

- AsNoTracking

```
var blogs = context.Blogs.AsNoTracking().ToList();
```

Проекции

Загружаем из базы только те поля, которые нам нужны

- Широкая таблица
- Неочевидный пример (join)

IQueryable vs IEnumerable

Различия

- IEnumerable doesn’t support lazy loading
- IQueryable support lazy loading
- Querying data from a database, IEnumerable execute a select query on the server side, load data in-memory on a client-side and then filter data.
- Querying data from a database, IQueryable execute the select query on the server side with all filters.

Асинхронность