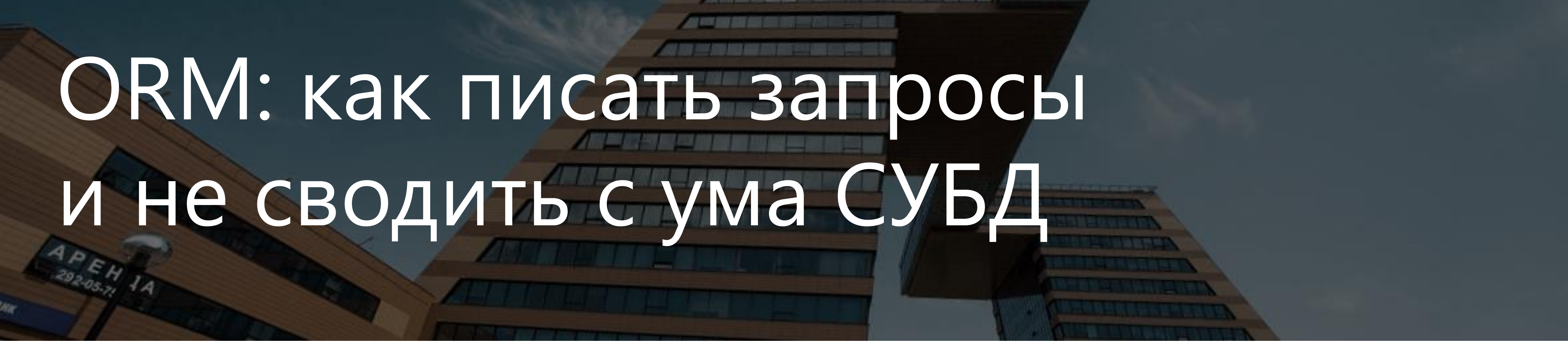


ORM: как писать запросы и не сводить с ума СУБД

Обо мне

Фадеев Алексей Сергеевич

C# разработчик

Компания SibEDGE, г.Томск

Сфера деятельности

Backend-разработка

Работа с СУБД, оптимизация

Используем: PostgreSQL, Microsoft SQL Server

see beyond
THE TOP
KOMTAR PENANG
I
PG
PENANG STATE GALLER

Microsoft ❤ Linux

THE LINUX FOUNDATION

.NET Core

Modular libraries & runtime optimized for server and cloud workloads

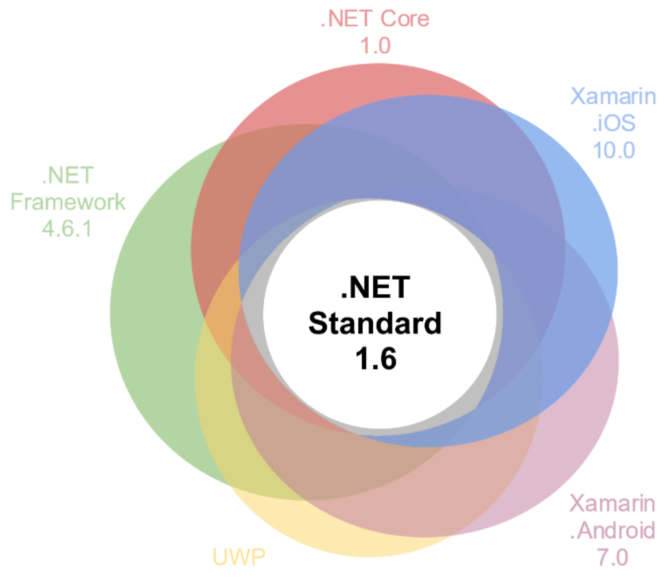
Microsoft

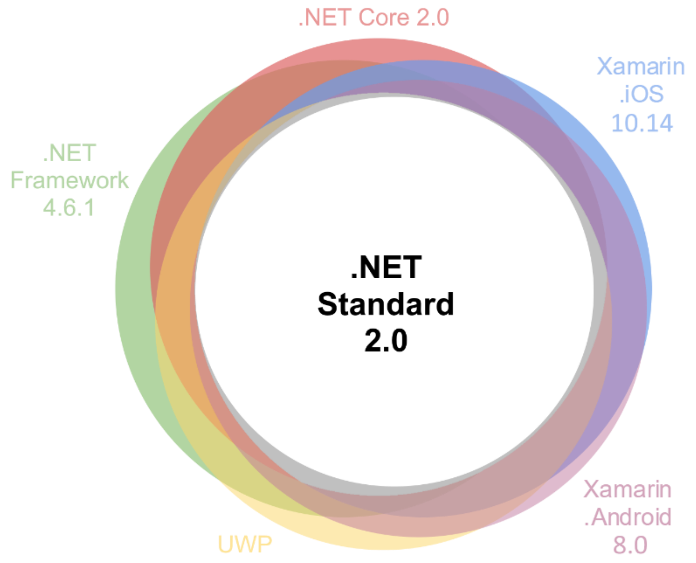
Linux

THE LINUX FOUNDATION

.NET Core

Modular libraries & runtime optimized for server and cloud workloads





```sql
SELECT
    [UnionAll1].[Id] AS [C1],
    [UnionAll1].[Id1] AS [C2],
    [UnionAll1].[Id2] AS [C3],
    [UnionAll1].[Id3] AS [C4],
    [UnionAll1].[Id4] AS [C5],
    [UnionAll1].[Id5] AS [C6],
    [UnionAll1].[Id6] AS [C7],
    [UnionAll1].[Code1] AS [C8],
    [UnionAll1].[VariantId] AS [C9],
    [UnionAll1].[Description1] AS [C10],
    [UnionAll1].[Name] AS [C11],
    [UnionAll1].[BoxQuantity] AS [C12],
    [UnionAll1].[C1] AS [C13],
    [UnionAll1].[Id7] AS [C14],
    [UnionAll1].[ProductId] AS [C15],
    [UnionAll1].[VariantId1] AS [C16],
    [UnionAll1].[CutCodeAnnotation] AS [C17],
    [UnionAll1].[Description2] AS [C18],
    [UnionAll1].[IsActive] AS [C19],
    [UnionAll1].[CreatedDate] AS [C20],
    [UnionAll1].[NutritionServingSize] AS [C21],
    [UnionAll1].[NutritionServingContainer] AS [C22],
    [UnionAll1].[NutritionCalories] AS [C23],
    [UnionAll1].[NutritionTotalFit] AS [C24],
    [UnionAll1].[NutritionSatFat] AS [C25],
    [UnionAll1].[NutritionTransFat] AS [C26],
    [UnionAll1].[NutritionCholesterol] AS [C27],
    [UnionAll1].[NutritionSodium] AS [C28],
    [UnionAll1].[NutritionCarbs] AS [C29],
    [UnionAll1].[NutritionFiber] AS [C30],
    [UnionAll1].[NutritionSugar] AS [C31],
    [UnionAll1].[NutritionAddedSugar] AS [C32],
    [UnionAll1].[NutritionProtein] AS [C33],
    [UnionAll1].[NutritionVitD] AS [C34],
    [UnionAll1].[NutritionCalcium] AS [C35],
    [UnionAll1].[NutritionIron] AS [C36],
    [UnionAll1].[NutritionPotassium] AS [C37],
    [UnionAll1].[NutritionOverride] AS [C38],
    [UnionAll1].[C2] AS [C39],
    [UnionAll1].[C3] AS [C40],
    [UnionAll1].[C4] AS [C41],
    [UnionAll1].[C5] AS [C42],
    [UnionAll1].[C6] AS [C43],
    [UnionAll1].[C7] AS [C44],
    [UnionAll1].[C8] AS [C45],
    [UnionAll1].[C9] AS [C46],
    [UnionAll1].[C10] AS [C47],
    [UnionAll1].[C11] AS [C48],
    [UnionAll1].[C12] AS [C49],
    [UnionAll1].[C13] AS [C50],
    [UnionAll1].[C14] AS [C51],
    [UnionAll1].[C15] AS [C52],
    [UnionAll1].[C16] AS [C53],
    [UnionAll1].[C17] AS [C54],
    [UnionAll1].[C18] AS [C55],
    [UnionAll1].[C19] AS [C56],
    [UnionAll1].[C20] AS [C57],
    [UnionAll1].[C21] AS [C58],
    [UnionAll1].[C22] AS [C59],
    [UnionAll1].[C23] AS [C60],
    [UnionAll1].[C24] AS [C61],
    [UnionAll1].[C25] AS [C62],
    [UnionAll1].[C26] AS [C63],
    [UnionAll1].[C27] AS [C64],
    [UnionAll1].[C28] AS [C65]
    FROM  (SELECT
        CASE WHEN ([Extent4].[Id] IS NULL) THEN
        CAST(NULL AS int) ELSE 1 END AS [C1],
        [Limit1].[SortOrder] AS [SortOrder],
        [Limit1].[Description] AS [Description],
        [Limit1].[Code] AS [Code],
        [Limit1].[Id1] AS [Id],
        [Limit1].[Id1] AS [Id1],
        [Limit1].[Id2] AS [Id2],
        [Limit1].[Id1] AS [Id3],
        [Limit1].[Id2] AS [Id4],
        [Limit1].[Id] AS [Id5],
        [Limit1].[Id] AS [Id6],
        [Limit1].[Code] AS [Code1],
        [Limit1].[VariantId] AS [VariantId],
        [Limit1].[Description] AS [Description1],
        [Limit1].[Name] AS [Name],
        [Limit1].[BoxQuantity] AS [BoxQuantity],
        [Extent4].[Id] AS [Id7],
        [Extent4].[ProductId] AS [ProductId],
        [Extent4].[VariantId] AS [VariantId1],
        [Extent4].[CutCodeAnnotation] AS [CutCodeAnnotation],
        [Extent4].[Description] AS [Description2],
        [Extent4].[IsActive] AS [IsActive],
        [Extent4].[CreatedDate] AS [CreatedDate],
        [Extent4].[NutritionServingSize] AS [NutritionServingSize],
        [Extent4].[NutritionServingContainer]
            AS [NutritionServingContainer],
        [Extent4].[NutritionCalories] AS [NutritionCalories],
        [Extent4].[NutritionTotalFit] AS [NutritionTotalFit],
        [Extent4].[NutritionSatFat] AS [NutritionSatFat],
        [Extent4].[NutritionTransFat] AS [NutritionTransFat],
        [Extent4].[NutritionCholesterol] AS [NutritionCholesterol],
        [Extent4].[NutritionSodium] AS [NutritionSodium],
        [Extent4].[NutritionCarbs] AS [NutritionCarbs],
        [Extent4].[NutritionFiber] AS [NutritionFiber],
        [Extent4].[NutritionSugar] AS [NutritionSugar],
        [Extent4].[NutritionAddedSugar] AS [NutritionAddedSugar],
        [Extent4].[NutritionProtein] AS [NutritionProtein],
        [Extent4].[NutritionVitD] AS [NutritionVitD],
        [Extent4].[NutritionCalcium] AS [NutritionCalcium],
    [Extent4].[NutritionIron] AS [NutritionIron],
        [Extent4].[NutritionPotassium] AS [NutritionPotassium],
        [Extent4].[NutritionOverride] AS [NutritionOverride],
        CAST(NULL AS int) AS [C2],
        CAST(NULL AS int) AS [C3],
        CAST(NULL AS int) AS [C4],
        CAST(NULL AS int) AS [C5],
        CAST(NULL AS int) AS [C6],
        CAST(NULL AS int) AS [C7],
        CAST(NULL AS varchar(1)) AS [C8],
        CAST(NULL AS varchar(1)) AS [C9],
        CAST(NULL AS int) AS [C10],
        CAST(NULL AS int) AS [C11],
        CAST(NULL AS int) AS [C12],
        CAST(NULL AS int) AS [C13],
        CAST(NULL AS varchar(1)) AS [C14],
        CAST(NULL AS int) AS [C15],
        CAST(NULL AS decimal(8,3)) AS [C16],
        CAST(NULL AS int) AS [C17],
        CAST(NULL AS int) AS [C18],
        CAST(NULL AS decimal(5,2)) AS [C19],
        CAST(NULL AS int) AS [C20],
        CAST(NULL AS int) AS [C21],
        CAST(NULL AS decimal(5,2)) AS [C22],
        CAST(NULL AS int) AS [C23],
        CAST(NULL AS int) AS [C24],
        CAST(NULL AS decimal(6,2)) AS [C25],
        CAST(NULL AS decimal(19,4)) AS [C26],
        CAST(NULL AS varchar(1)) AS [C27],
        CAST(NULL AS datetime2) AS [C28]
        FROM   (SELECT [Project1].[Id] AS [Id], [Project1].[VariantId] AS [VariantId],
          [Project1].[BoxQuantity] AS [BoxQuantity], [Project1].[Id1] AS [Id1],
          [Project1].[Code] AS [Code], [Project1].[Description] AS [Description],
          [Project1].[Id2] AS [Id2], [Project1].[Name] AS [Name],
         [Project1].[SortOrder] AS [SortOrder]
          FROM ( SELECT
            [Extent1].[Id] AS [Id],
            [Extent1].[VariantId] AS [VariantId],
            [Extent1].[BoxQuantity] AS [BoxQuantity],
            [Extent2].[Id] AS [Id1],
            [Extent2].[Code] AS [Code],
            [Extent2].[Description] AS [Description],
            [Extent3].[Id] AS [Id2],
            [Extent3].[Name] AS [Name],
            [Extent3].[SortOrder] AS [SortOrder]
            FROM   [dbo].[OrderDetailFab] AS [Extent1]
            INNER JOIN [dbo].[Product] AS [Extent2] ON [Extent1].[ProductId]
              = [Extent2].[Id]
            INNER JOIN [dbo].[PrimalCut] AS [Extent3] ON [Extent2].[PrimalCutId]
              = [Extent3].[Id]
            WHERE ([Extent1].[OrderId] = @p__linq__0) AND ((@p__linq__1 IS NULL)
              OR ([Extent2].[PrimalCutId] = @p__linq__2) OR (1 = 0))
          )  AS [Project1]
          ORDER BY row_number() OVER (ORDER BY [Project1].[SortOrder] ASC,
           [Project1].[Description] ASC, [Project1].[Code] ASC)
          OFFSET 0 ROWS FETCH NEXT 100 ROWS ONLY  ) AS [Limit1]
        LEFT OUTER JOIN [dbo].[ProductVariant] AS [Extent4] ON [Limit1].[Id1]
         = [Extent4].[ProductId]
    UNION ALL
        SELECT
        2 AS [C1],
        [Limit2].[SortOrder] AS [SortOrder],
        [Limit2].[Description] AS [Description],
        [Limit2].[Code] AS [Code],
        [Limit2].[Id1] AS [Id],
        [Limit2].[Id1] AS [Id1],
        [Limit2].[Id2] AS [Id2],
        [Limit2].[Id1] AS [Id3],
        [Limit2].[Id2] AS [Id4],
        [Limit2].[Id] AS [Id5],
        [Limit2].[Id] AS [Id6],
        [Limit2].[Code] AS [Code1],
        [Limit2].[VariantId] AS [VariantId],
        [Limit2].[Description] AS [Description1],
        [Limit2].[Name] AS [Name],
        [Limit2].[BoxQuantity] AS [BoxQuantity],
    CAST(NULL AS int) AS [C2],
        CAST(NULL AS int) AS [C3],
        CAST(NULL AS int) AS [C4],
        CAST(NULL AS int) AS [C5],
        CAST(NULL AS varchar(1)) AS [C6],
        CAST(NULL AS bit) AS [C7],
        CAST(NULL AS datetime2) AS [C8],
        CAST(NULL AS varchar(1)) AS [C9],
        CAST(NULL AS varchar(1)) AS [C10],
        CAST(NULL AS decimal(19,5)) AS [C11],
        CAST(NULL AS decimal(19,5)) AS [C12],
        CAST(NULL AS decimal(4,1)) AS [C13],
        CAST(NULL AS decimal(4,1)) AS [C14],
        CAST(NULL AS decimal(19,5)) AS [C15],
        CAST(NULL AS decimal(19,5)) AS [C16],
        CAST(NULL AS decimal(19,5)) AS [C17],
        CAST(NULL AS decimal(19,5)) AS [C18],
        CAST(NULL AS decimal(19,5)) AS [C19],
        CAST(NULL AS decimal(19,5)) AS [C20],
        CAST(NULL AS decimal(19,5)) AS [C21],
        CAST(NULL AS decimal(19,5)) AS [C22],
        CAST(NULL AS decimal(19,5)) AS [C23],
        CAST(NULL AS decimal(19,5)) AS [C24],
        CAST(NULL AS decimal(19,5)) AS [C25],
        CAST(NULL AS bit) AS [C26],
        [Extent8].[Id] AS [Id7],
        [Extent8].[BusinessUnitId] AS [BusinessUnitId],
        [Extent8].[DetailId] AS [DetailId],
        [Extent8].[OrderId] AS [OrderId],
        [Extent8].[ProductId] AS [ProductId],
     [Extent8].[VariantId] AS [VariantId1],
        [Extent8].[DjProductCode] AS [DjProductCode],
        [Extent8].[CustomerProductCode] AS [CustomerProductCode],
        [Extent8].[AnimalCategoryId] AS [AnimalCategoryId],
        [Extent8].[AnimalGradeId] AS [AnimalGradeId],
        [Extent8].[AnimalSubGradeId] AS [AnimalSubGradeId],
        [Extent8].[EndReceiverId] AS [EndReceiverId],
        [Extent8].[AnimalNumber] AS [AnimalNumber],
        [Extent8].[AnimalSideId] AS [AnimalSideId],
        [Extent8].[ScaleWeight] AS [ScaleWeight],
        [Extent8].[ScaleQuantity] AS [ScaleQuantity],
        [Extent8].[BagSizeId] AS [BagSizeId],
        [Extent8].[BagTareWeight] AS [BagTareWeight],
        [Extent8].[PiecesPerBag] AS [PiecesPerBag],
        [Extent8].[BoxSizeId] AS [BoxSizeId],
        [Extent8].[BoxTareWeight] AS [BoxTareWeight],
        [Extent8].[BagsPerBox] AS [BagsPerBox],
        [Extent8].[BagsPerPartialBox] AS [BagsPerPartialBox],
        [Extent8].[TotalTareWeight] AS [TotalTareWeight],
        [Extent8].[PricePerPound] AS [PricePerPound],
        [Extent8].[SourceMachineName] AS [SourceMachineName],
        [Extent8].[CreatedDate] AS [CreatedDate]
        FROM   (SELECT [Project3].[Id] AS [Id], [Project3].[VariantId] AS [VariantId],
         [Project3].[BoxQuantity] AS [BoxQuantity], [Project3].[Id1] AS [Id1],
         [Project3].[Code] AS [Code], [Project3].[Description] AS [Description],
         [Project3].[Id2] AS [Id2], [Project3].[Name] AS [Name], [Project3].[SortOrder] AS [SortOrder]
          FROM ( SELECT
            [Extent5].[Id] AS [Id],
            [Extent5].[VariantId] AS [VariantId],
            [Extent5].[BoxQuantity] AS [BoxQuantity],
            [Extent6].[Id] AS [Id1],
            [Extent6].[Code] AS [Code],
            [Extent6].[Description] AS [Description],
            [Extent7].[Id] AS [Id2],
            [Extent7].[Name] AS [Name],
            [Extent7].[SortOrder] AS [SortOrder]
            FROM   [dbo].[OrderDetailFab] AS [Extent5]
            INNER JOIN [dbo].[Product] AS [Extent6] ON [Extent5].[ProductId] = [Extent6].[Id]
            INNER JOIN [dbo].[PrimalCut] AS [Extent7] ON [Extent6].[PrimalCutId] = [Extent7].[Id]
            WHERE ([Extent5].[OrderId] = @p__linq__0) AND ((@p__linq__1 IS NULL)
              OR ([Extent6].[PrimalCutId] = @p__linq__2) OR (1 = 0))
          )  AS [Project3]
          ORDER BY row_number() OVER (ORDER BY [Project3].[SortOrder] ASC,
            [Project3].[Description] ASC, [Project3].[Code] ASC)
          OFFSET 0 ROWS FETCH NEXT 100 ROWS ONLY  ) AS [Limit2]
        INNER JOIN [dbo].[ProcessingLog] AS [Extent8] ON (2 = [Extent8].[BusinessUnitId])
          AND ([Limit2].[Id] = [Extent8].[DetailId])) AS [UnionAll1]
      ORDER BY [UnionAll1].[SortOrder] ASC, [UnionAll1].[Description] ASC, [UnionAll1].[Code] ASC,
       [UnionAll1].[Id] ASC, [UnionAll1].[Id1] ASC, [UnionAll1].[Id2] ASC, [UnionAll1].[Id3] ASC,
       [UnionAll1].[Id4] ASC, [UnionAll1].[Id6] ASC, [UnionAll1].[C1] ASC
```

```
SELECT
  [UnionAll1].[Id] AS [C1],
  [UnionAll1].[Id1] AS [C2],
  [UnionAll1].[Id2] AS [C3],
  [UnionAll1].[Id3] AS [C4],
  [UnionAll1].[Id4] AS [C5],
  [UnionAll1].[Id5] AS [C6],
  [UnionAll1].[Id6] AS [C7],
  [UnionAll1].[Code1] AS [C8],
  [UnionAll1].[VariantId] AS [C9],
  [UnionAll1].[Description1] AS [C10],
  [UnionAll1].[Name] AS [C11],
  [UnionAll1].[BoxQuantity] AS [C12],
  [UnionAll1].[C1] AS [C13],
  [UnionAll1].[Id7] AS [C14],
  [UnionAll1].[ProductId] AS [C15],
  [UnionAll1].[VariantId1] AS [C16],
  [UnionAll1].[CutCodeAnnotation] AS [C17],
  [UnionAll1].[Description2] AS [C18],
  [UnionAll1].[IsActive] AS [C19],
  [UnionAll1].[CreatedDate] AS [C20],
  [UnionAll1].[NutritionServingSize] AS [C21],
  [UnionAll1].[NutritionServingContainer] AS [C22],
  [UnionAll1].[NutritionCalories] AS [C23],
  [UnionAll1].[NutritionTotalFit] AS [C24],
  [UnionAll1].[NutritionSatFat] AS [C25],
  [UnionAll1].[NutritionTransFat] AS [C26],
  [UnionAll1].[NutritionCholesterol] AS [C27],
  [UnionAll1].[NutritionSodium] AS [C28],
  [UnionAll1].[NutritionCarbs] AS [C29],
  [UnionAll1].[NutritionFiber] AS [C30],
  [UnionAll1].[NutritionSugar] AS [C31],
  [UnionAll1].[NutritionAddedSugar] AS [C32],
  [UnionAll1].[NutritionProtein] AS [C33],
  [UnionAll1].[NutritionVitD] AS [C34],
  [UnionAll1].[NutritionCalcium] AS [C35],
  [UnionAll1].[NutritionIron] AS [C36],
  [UnionAll1].[NutritionPotassium] AS [C37],
  [UnionAll1].[NutritionOverride] AS [C38],
  [UnionAll1].[C2] AS [C39],
  [UnionAll1].[C3] AS [C40],
  [UnionAll1].[C4] AS [C41],
  [UnionAll1].[C5] AS [C42],
  [UnionAll1].[C6] AS [C43],
  [UnionAll1].[C7] AS [C44],
  [UnionAll1].[C8] AS [C45],
  [UnionAll1].[C9] AS [C46],
  [UnionAll1].[C10] AS [C47],
  [UnionAll1].[C11] AS [C48],
  [UnionAll1].[C12] AS [C49],
  [UnionAll1].[C13] AS [C50],
  [UnionAll1].[C14] AS [C51],
  [UnionAll1].[C15] AS [C52],
  [UnionAll1].[C16] AS [C53],
  [UnionAll1].[C17] AS [C54],
  [UnionAll1].[C18] AS [C55],
  [UnionAll1].[C19] AS [C56],
  [UnionAll1].[C20] AS [C57],
  [UnionAll1].[C21] AS [C58],
  [UnionAll1].[C22] AS [C59],
  [UnionAll1].[C23] AS [C60],
  [UnionAll1].[C24] AS [C61],
  [UnionAll1].[C25] AS [C62],
  [UnionAll1].[C26] AS [C63],
  [UnionAll1].[C27] AS [C64],
  [UnionAll1].[C28] AS [C65]
  FROM  (SELECT
    CASE WHEN ([Extent4].[Id] IS NULL) THEN
    CAST(NULL AS int) ELSE 1 END AS [C1],
    [Limit1].[SortOrder] AS [SortOrder],
    [Limit1].[Description] AS [Description],
    [Limit1].[Code] AS [Code],
    [Limit1].[Id1] AS [Id],
    [Limit1].[Id1] AS [Id1],
    [Limit1].[Id2] AS [Id2],
    [Limit1].[Id1] AS [Id3],
    [Limit1].[Id2] AS [Id4],
    [Limit1].[Id] AS [Id5],
    [Limit1].[Id] AS [Id6],
    [Limit1].[Code] AS [Code1],
    [Limit1].[VariantId] AS [VariantId],
    [Limit1].[Description] AS [Description1],
    [Limit1].[Name] AS [Name],
    [Limit1].[BoxQuantity] AS [BoxQuantity],
    [Extent4].[Id] AS [Id7],
    [Extent4].[ProductId] AS [ProductId],
    [Extent4].[VariantId] AS [VariantId1],
    [Extent4].[CutCodeAnnotation] AS [CutCodeAnnotation],
    [Extent4].[Description] AS [Description2],
    [Extent4].[IsActive] AS [IsActive],
    [Extent4].[CreatedDate] AS [CreatedDate],
    [Extent4].[NutritionServingSize] AS [NutritionServingSize],
    [Extent4].[NutritionServingContainer]
       AS [NutritionServingContainer],
    [Extent4].[NutritionCalories] AS [NutritionCalories],
    [Extent4].[NutritionTotalFit] AS [NutritionTotalFit],
    [Extent4].[NutritionSatFat] AS [NutritionSatFat],
    [Extent4].[NutritionTransFat] AS [NutritionTransFat],
    [Extent4].[NutritionCholesterol] AS [NutritionCholesterol],
    [Extent4].[NutritionSodium] AS [NutritionSodium],
    [Extent4].[NutritionCarbs] AS [NutritionCarbs],
    [Extent4].[NutritionFiber] AS [NutritionFiber],
    [Extent4].[NutritionSugar] AS [NutritionSugar],
    [Extent4].[NutritionAddedSugar] AS [NutritionAddedSugar],
    [Extent4].[NutritionProtein] AS [NutritionProtein],
    [Extent4].[NutritionVitD] AS [NutritionVitD],
    [Extent4].[NutritionCalcium] AS [NutritionCalcium],
[Extent4].[NutritionIron] AS [NutritionIron],
    [Extent4].[NutritionPotassium] AS [NutritionPotassium],
    [Extent4].[NutritionOverride] AS [NutritionOverride],
    CAST(NULL AS int) AS [C2],
    CAST(NULL AS int) AS [C3],
    CAST(NULL AS int) AS [C4],
    CAST(NULL AS int) AS [C5],
    CAST(NULL AS int) AS [C6],
    CAST(NULL AS int) AS [C7],
    CAST(NULL AS varchar(1)) AS [C8],
    CAST(NULL AS varchar(1)) AS [C9],
    CAST(NULL AS int) AS [C10],
    CAST(NULL AS int) AS [C11],
    CAST(NULL AS int) AS [C12],
    CAST(NULL AS int) AS [C13],
    CAST(NULL AS varchar(1)) AS [C14],
    CAST(NULL AS int) AS [C15],
    CAST(NULL AS decimal(8,3)) AS [C16],
    CAST(NULL AS int) AS [C17],
    CAST(NULL AS int) AS [C18],
    CAST(NULL AS decimal(5,2)) AS [C19],
    CAST(NULL AS int) AS [C20],
    CAST(NULL AS int) AS [C21],
    CAST(NULL AS decimal(5,2)) AS [C22],
    CAST(NULL AS int) AS [C23],
    CAST(NULL AS int) AS [C24],
    CAST(NULL AS decimal(6,2)) AS [C25],
    CAST(NULL AS decimal(19,4)) AS [C26],
    CAST(NULL AS varchar(1)) AS [C27],
    CAST(NULL AS datetime2) AS [C28]
    FROM   (SELECT [Project1].[Id] AS [Id], [Project1].[VariantId] AS [VariantId],
      [Project1].[BoxQuantity] AS [BoxQuantity], [Project1].[Id1] AS [Id1],
      [Project1].[Code] AS [Code], [Project1].[Description] AS [Description],
      [Project1].[Id2] AS [Id2], [Project1].[Name] AS [Name],
     [Project1].[SortOrder] AS [SortOrder]
      FROM ( SELECT
        [Extent1].[Id] AS [Id],
        [Extent1].[VariantId] AS [VariantId],
        [Extent1].[BoxQuantity] AS [BoxQuantity],
        [Extent2].[Id] AS [Id1],
        [Extent2].[Code] AS [Code],
        [Extent2].[Description] AS [Description],
        [Extent3].[Id] AS [Id2],
        [Extent3].[Name] AS [Name],
        [Extent3].[SortOrder] AS [SortOrder]
        FROM   [dbo].[OrderDetailFab] AS [Extent1]
        INNER JOIN [dbo].[Product] AS [Extent2] ON [Extent1].[ProductId]
          = [Extent2].[Id]
        INNER JOIN [dbo].[PrimalCut] AS [Extent3] ON [Extent2].[PrimalCutId]
          = [Extent3].[Id]
        WHERE ([Extent1].[OrderId] = @p__linq__0) AND ((@p__linq__1 IS NULL)
         OR ([Extent2].[PrimalCutId] = @p__linq__2) OR (1 = 0))
      )  AS [Project1]
      ORDER BY row_number() OVER (ORDER BY [Project1].[SortOrder] ASC,
       [Project1].[Description] ASC, [Project1].[Code] ASC)
      OFFSET 0 ROWS FETCH NEXT 100 ROWS ONLY  ) AS [Limit1]
    LEFT OUTER JOIN [dbo].[ProductVariant] AS [Extent4] ON [Limit1].[Id1]
     = [Extent4].[ProductId]
  UNION ALL
    SELECT
    2 AS [C1],
    [Limit2].[SortOrder] AS [SortOrder],
    [Limit2].[Description] AS [Description],
    [Limit2].[Code] AS [Code],
    [Limit2].[Id1] AS [Id],
    [Limit2].[Id1] AS [Id1],
    [Limit2].[Id2] AS [Id2],
    [Limit2].[Id1] AS [Id3],
    [Limit2].[Id2] AS [Id4],
    [Limit2].[Id] AS [Id5],
    [Limit2].[Id] AS [Id6],
    [Limit2].[Code] AS [Code1],
    [Limit2].[VariantId] AS [VariantId],
    [Limit2].[Description] AS [Description1],
    [Limit2].[Name] AS [Name],
    [Limit2].[BoxQuantity] AS [BoxQuantity],
CAST(NULL AS int) AS [C2],
    CAST(NULL AS int) AS [C3],
    CAST(NULL AS int) AS [C4],
    CAST(NULL AS int) AS [C5],
    CAST(NULL AS varchar(1)) AS [C6],
    CAST(NULL AS bit) AS [C7],
    CAST(NULL AS datetime2) AS [C8],
    CAST(NULL AS varchar(1)) AS [C9],
    CAST(NULL AS varchar(1)) AS [C10],
    CAST(NULL AS decimal(19,5)) AS [C11],
    CAST(NULL AS decimal(19,5)) AS [C12],
    CAST(NULL AS decimal(4,1)) AS [C13],
    CAST(NULL AS decimal(4,1)) AS [C14],
    CAST(NULL AS decimal(19,5)) AS [C15],
    CAST(NULL AS decimal(19,5)) AS [C16],
    CAST(NULL AS decimal(19,5)) AS [C17],
    CAST(NULL AS decimal(19,5)) AS [C18],
    CAST(NULL AS decimal(19,5)) AS [C19],
    CAST(NULL AS decimal(19,5)) AS [C20],
    CAST(NULL AS decimal(19,5)) AS [C21],
    CAST(NULL AS decimal(19,5)) AS [C22],
    CAST(NULL AS decimal(19,5)) AS [C23],
    CAST(NULL AS decimal(19,5)) AS [C24],
    CAST(NULL AS decimal(19,5)) AS [C25],
    CAST(NULL AS bit) AS [C26],
    [Extent8].[Id] AS [Id7],
    [Extent8].[BusinessUnitId] AS [BusinessUnitId],
    [Extent8].[DetailId] AS [DetailId],
    [Extent8].[OrderId] AS [OrderId],
    [Extent8].[ProductId] AS [ProductId],
 [Extent8].[VariantId] AS [VariantId1],
    [Extent8].[DjProductCode] AS [DjProductCode],
    [Extent8].[CustomerProductCode] AS [CustomerProductCode],
    [Extent8].[AnimalCategoryId] AS [AnimalCategoryId],
    [Extent8].[AnimalGradeId] AS [AnimalGradeId],
    [Extent8].[AnimalSubGradeId] AS [AnimalSubGradeId],
    [Extent8].[EndReceiverId] AS [EndReceiverId],
    [Extent8].[AnimalNumber] AS [AnimalNumber],
    [Extent8].[AnimalSideId] AS [AnimalSideId],
    [Extent8].[ScaleWeight] AS [ScaleWeight],
    [Extent8].[ScaleQuantity] AS [ScaleQuantity],
    [Extent8].[BagSizeId] AS [BagSizeId],
    [Extent8].[BagTareWeight] AS [BagTareWeight],
    [Extent8].[PiecesPerBag] AS [PiecesPerBag],
    [Extent8].[BoxSizeId] AS [BoxSizeId],
    [Extent8].[BoxTareWeight] AS [BoxTareWeight],
    [Extent8].[BagsPerBox] AS [BagsPerBox],
    [Extent8].[BagsPerPartialBox] AS [BagsPerPartialBox],
    [Extent8].[TotalTareWeight] AS [TotalTareWeight],
    [Extent8].[PricePerPound] AS [PricePerPound],
    [Extent8].[SourceMachineName] AS [SourceMachineName],
    [Extent8].[CreatedDate] AS [CreatedDate]
    FROM   (SELECT [Project3].[Id] AS [Id], [Project3].[VariantId] AS [VariantId],
      [Project3].[BoxQuantity] AS [BoxQuantity], [Project3].[Id1] AS [Id1],
      [Project3].[Code] AS [Code], [Project3].[Description] AS [Description],
      [Project3].[Id2] AS [Id2], [Project3].[Name] AS [Name], [Project3].[SortOrder] AS [SortOrder]
      FROM ( SELECT
        [Extent5].[Id] AS [Id],
        [Extent5].[VariantId] AS [VariantId],
        [Extent5].[BoxQuantity] AS [BoxQuantity],
        [Extent6].[Id] AS [Id1],
        [Extent6].[Code] AS [Code],
        [Extent6].[Description] AS [Description],
        [Extent7].[Id] AS [Id2],
        [Extent7].[Name] AS [Name],
        [Extent7].[SortOrder] AS [SortOrder]
        FROM   [dbo].[OrderDetailFab] AS [Extent5]
        INNER JOIN [dbo].[Product] AS [Extent6] ON [Extent5].[ProductId] = [Extent6].[Id]
        INNER JOIN [dbo].[PrimalCut] AS [Extent7] ON [Extent6].[PrimalCutId] = [Extent7].[Id]
        WHERE ([Extent5].[OrderId] = @p__linq__0) AND ((@p__linq__1 IS NULL)
          OR ([Extent6].[PrimalCutId] = @p__linq__2) OR (1 = 0))
      )  AS [Project3]
      ORDER BY row_number() OVER (ORDER BY [Project3].[SortOrder] ASC,
        [Project3].[Description] ASC, [Project3].[Code] ASC)
      OFFSET 0 ROWS FETCH NEXT 100 ROWS ONLY  ) AS [Limit2]
    INNER JOIN [dbo].[ProcessingLog] AS [Extent8] ON (2 = [Extent8].[BusinessUnitId])
      AND ([Limit2].[Id] = [Extent8].[DetailId])) AS [UnionAll1]
  ORDER BY [UnionAll1].[SortOrder] ASC, [UnionAll1].[Description] ASC, [UnionAll1].[Code] ASC,
   [UnionAll1].[Id] ASC, [UnionAll1].[Id1] ASC, [UnionAll1].[Id2] ASC, [UnionAll1].[Id3] ASC,
   [UnionAll1].[Id4] ASC, [UnionAll1].[Id6] ASC, [UnionAll1].[C1] ASC
```

```
SELECT
  [Project1].[Id] AS [Id],
  [Project1].[OrderId] AS [OrderId],
  [Project1].[Code] AS [Code],
  [Project1].[Description] AS [Description],
  [Project1].[Name] AS [Name],
  [Project1].[SortOrder] AS [SortOrder],
  [Project1].[C1] AS [C1],
  [Project1].[C2] AS [C2],
  [Project1].[BoxQuantity] AS [BoxQuantity]
  FROM ( SELECT
    [Extent1].[Id] AS [Id],
    [Extent1].[OrderId] AS [OrderId],
    [Extent1].[BoxQuantity] AS [BoxQuantity],
    [Extent4].[Code] AS [Code],
    [Extent4].[Description] AS [Description],
    [Extent5].[Name] AS [Name],
    [Extent5].[SortOrder] AS [SortOrder],
    CASE WHEN ([Extent3].[Id] IS NULL) THEN 0 ELSE [Extent3].[CutCodeAnnotation] END AS [C1],
    CASE WHEN ([GroupBy1].[K1] IS NULL) THEN 0 ELSE [GroupBy1].[A1] END AS [C2]
    FROM     [dbo].[OrderDetailFab] AS [Extent1]
    LEFT OUTER JOIN  (SELECT
      [Extent2].[DetailId] AS [K1],
      SUM([Extent2].[ScaleQuantity]) AS [A1]
      FROM [dbo].[ProcessingLog] AS [Extent2]
      WHERE ([Extent2].[OrderId] = @p__linq__0) AND (2 = [Extent2].[BusinessUnitId])
      GROUP BY [Extent2].[DetailId] ) AS [GroupBy1] ON [Extent1].[Id] = [GroupBy1].[K1]
    LEFT OUTER JOIN [dbo].[ProductVariant] AS [Extent3] ON ([Extent1].[ProductId]
      = [Extent3].[ProductId]) AND (([Extent1].[VariantId] = [Extent3].[VariantId])
      OR (([Extent1].[VariantId] IS NULL) AND ([Extent3].[VariantId] IS NULL)))
    INNER JOIN [dbo].[Product] AS [Extent4] ON [Extent1].[ProductId] = [Extent4].[Id]
    INNER JOIN [dbo].[PrimalCut] AS [Extent5] ON [Extent4].[PrimalCutId] = [Extent5].[Id]
    WHERE ([Extent1].[OrderId] = @p__linq__1) AND ((@p__linq__2 IS NULL) OR ([Extent4].[PrimalCutId] = @p__linq__3) OR (1 = 0))
  )  AS [Project1]
  ORDER BY row_number() OVER (ORDER BY [Project1].[SortOrder] ASC, [Project1].[Description] ASC, [Project1].[Code] ASC)
  OFFSET 0 ROWS FETCH NEXT 100 ROWS ONLY
```

Метод AsNoTracking

Если не требуется изменение данных

Для больших объёмов данных

```
var users = context.Users.AsNoTracking();

var moneybags = context.Employees
    .Where(x => x.Salary > 1000000)
    .AsNoTracking();
```

.Net Framework

Tracking

```
00:00:14.119
00:00:14.302
```

NoTracking

```
00:00:07.302
00:00:07.372
```

x1.94

.Net Core

Tracking

```
00:00:08.392
00:00:08.619
```

NoTracking

…

.Net Framework

Tracking

```
00:00:14.119
00:00:14.302
```

NoTracking

```
00:00:07.302
00:00:07.372
```

x1.94

.Net Core

Tracking

```
00:00:08.392
00:00:08.619
```

NoTracking

```
00:00:02.510
00:00:02.804
```

x3.34

.Net Framework

Tracking

```
00:00:14.119
00:00:14.302
```

NoTracking

```
00:00:07.302
00:00:07.372
```

.Net Core

Tracking

```
00:00:08.392
00:00:08.619
```

x1.68

NoTracking

```
00:00:02.510
00:00:02.804
```

x2.6

View без PK

```
var users = context.UsersExtended.ToList();
```

Все объекты одинаковые!

```
var users = context.UsersExtended
    .AsNoTracking().ToList();
```

OK

Когда AsNoTracking не нужен

Результат запроса – объекты не Entity-класса

```
var employees = context.Employees.Select(x =>
                new
                {
                    x.IDEmployee,
                    x.Name,
                    x.Surname
                })
                .ToList();
```

С view без PK – тоже всё ОК

Entity Framework

Многофункциональная технология

Dapper .Net

Простая высокопроизводительная технология

.Net Framework

EF (AsNoTracking)

```
00:00:07.302
00:00:07.372
```

Dapper

```
00:00:04.283
00:00:04.335
```

x1.7

.Net Core

EF (AsNoTracking)

```
00:00:02.510
00:00:02.804
```

Dapper

```
00:00:02.150
00:00:02.179
```

x1.22

Dapper не генерирует SQL

```
var user = db.Query<User>("SELECT * FROM users WHERE Id = @id",
    new { id })
    .FirstOrDefault();
```

Написание запросов остаётся за разработчиком

Решение

Сгенерировать SQL-запрос в Entity Framework

Выполнить через Dapper

```
Install-Package Dapper.EntityFramework.Extensions
```

.Net Framework

```
var moneybags = context.Employees
    .Where(x => x.Salary > 1000000)
    .ToDapper();
```

Решение

Сгенерировать SQL-запрос в Entity Framework

Выполнить через Dapper

```
var sql = context.Employees
    .Where(x => x.Salary > 1000000)
    .ToString();
```

.Net Framework

Получить SQL в .Net Core

?

```csharp
using System.Linq;
using System.Reflection;

using Microsoft.EntityFrameworkCore.Query;
using Microsoft.EntityFrameworkCore.Query.Internal;
using Microsoft.EntityFrameworkCore.Storage;

public static class SqlHelper
{
    private static readonly TypeInfo QueryCompilerTypeInfo = typeof(QueryCompiler).GetTypeInfo();

    private static readonly FieldInfo QueryCompilerField = typeof(EntityQueryProvider)
        .GetTypeInfo().DeclaredFields.First(x => x.Name == "_queryCompiler");

    private static readonly FieldInfo QueryModelGeneratorField = typeof(QueryCompiler)
        .GetTypeInfo().DeclaredFields.First(x => x.Name == "_queryModelGenerator");

    private static readonly FieldInfo DataBaseField = QueryCompilerTypeInfo.DeclaredFields
        .Single(x => x.Name == "_database");

    private static readonly PropertyInfo DatabaseDependenciesField = typeof(Database)
        .GetTypeInfo().DeclaredProperties.Single(x => x.Name == "Dependencies");

    public static string ToSql<TEntity>(this IQueryable<TEntity> query)
    {
        var queryCompiler = (QueryCompiler)QueryCompilerField.GetValue(query.Provider);
        var queryModelGenerator = (QueryModelGenerator)QueryModelGeneratorField.GetValue(queryCompiler);
        var queryModel = queryModelGenerator.ParseQuery(query.Expression);
        var database = DataBaseField.GetValue(queryCompiler);
        var databaseDependencies = (DatabaseDependencies)DatabaseDependenciesField.GetValue(database);
        var queryCompilationContext = databaseDependencies.QueryCompilationContextFactory.Create(false);
        var modelVisitor = (RelationalQueryModelVisitor)queryCompilationContext.CreateQueryModelVisitor();
        modelVisitor.CreateQueryExecutor<TEntity>(queryModel);
        var sql = modelVisitor.Queries.First().ToString();

        return sql;
    }
}
```

Библиотека EFSqlTranslator

Библиотека генерирует SQL-запросы вместо Entity Framework

Запрос выполняется через Dapper

```
Install-Package EFSqlTranslator.Translation
Install-Package EFSqlTranslator.EFModels
```

.Net Standard

```
var query = context.Employees
    .Where(x => x.Salary > 1000000);

var moneybags = var result = db.Query(
                        query,
                        new EFModelInfoProvider(db),
                        new PostgresQlObjectFactory(),
                        out var sql);
```

.Net Framework

EF (AsNoTracking)

```
00:00:07.302
00:00:07.372
```

Dapper

```
00:00:04.283
00:00:04.335
```

x1.7

.Net Core

EF (AsNoTracking)

```
00:00:02.510
00:00:02.804
```

Dapper

```
00:00:02.150
00:00:02.179
```

x1.22

Mock? Не вопрос!

```
var moneybags = context.Employees
    .Where(x => x.Salary > 1000000)
    .ToDapper();
```

Entity Framework 6

Технология от Microsoft

Dapper .Net

Micro-ORM

LINQ to DB

Быстрый функциональный ORM-провайдер

.Net Framework

EF (AsNoTracking)

```
00:00:07.302
00:00:07.372
```

Dapper

```
00:00:04.283
00:00:04.335
```

LINQ to DB

```
00:00:02.785
00:00:02.812
```

x1.54 vs Dapper

x2.62 vs EF

Небольшие итоги

Один и тот же запрос может выполняться с разной скоростью

Можно отключить трекинг изменений

Можно использовать разные ORM вместе

Lazy loading

```
var man = context.Employees.First();

var sectName = man.Section.Name;
```

```
SELECT "Alias1"."id", "Alias1"."id_section", "Alias1"."name",
     "Alias1"."salary", "Alias1"."second_name", "Alias1"."surname"
FROM "public"."employee" AS "Alias1" LIMIT 1


SELECT "Extent1"."id", "Extent1"."id_department", "Extent1"."name"
FROM "public"."section" AS "Extent1"
WHERE "Extent1"."id_section" = @EntityKeyValue1
```

Lazy loading в цикле

```
var crew = context.Employees.ToList();

foreach (var man in crew)
{
    Console.WriteLine(man.Section.Name);
}
```

```
SELECT "Extent1"."id", "Extent1"."id_section", "Extent1"."name",
     "Extent1"."salary", "Extent1"."second_name", "Extent1"."surname",
FROM "public"."employee" AS "Extent1"


SELECT "Extent1"."id", "Extent1"."id_department", "Extent1"."name" FROM
"public"."section" AS "Extent1" WHERE "Extent1"."id_section" = @EntityKeyValue1

SELECT "Extent1"."id", "Extent1"."id_department", "Extent1"."name" FROM
"public"."section" AS "Extent1" WHERE "Extent1"."id_section" = @EntityKeyValue1

          ......
```

~~Lazy loading~~ в цикле

```
var crew = context.Employees
    .Include(x => x.Section)
    .ToList();

foreach (var man in crew)
{
    Console.WriteLine(man.Section.Name);
}
```

```
SELECT "Extent1"."id", "Extent1"."id_section", "Extent1"."name",
    "Extent1"."salary", "Extent1"."second_name", "Extent1"."surname",
    "Extent2"."id" AS "id1", "Extent2"."id_department", "Extent2"."name" AS "name1"
FROM "public"."employee" AS "Extent1"
LEFT OUTER JOIN "public"."section" AS "Extent2"
    ON "Extent1"."id_section"= "Extent2"."id_section"
```

Include

```
var crew = context.Employees
    .Include(x => x.Section)
    .Include(x => x.Section.Department)
    .ToList();
```

.Net Core:

```
var crew = context.Employees
    .Include(x => x.Section)
        .ThenInclude(s => s.Department)
    .ToList();
```

Генерируется JOIN

```
var crew = context.Employees
    .Include(x => x.Section)
    .Where(x => x.Section.Department.Name == "Филиал")
    .ToList();
```

```
SELECT "x"."id", "x"."id_section", "x"."name", "x"."salary", "x"."second_name",
    "x"."surname", "x.Section"."id", "x.Section"."id_department", "x.Section"."name"
FROM "public"."employee" AS "x"
LEFT JOIN "public"."section" AS "x.Section" ON "x"."id_section" =
    "x.Section"."id_section"
LEFT JOIN "public"."department" AS "x.Section.Department" ON
    "x.Section"."id_department" = "x.Section.Department"."id_department"
WHERE "x.Section.Department"."name" = 'Филиал'
```

Генерируется JOIN

```
var crew = context.Employees.Select(x => new
           {
               FullName = x.Surname + " " + x.Name,
               SectionName = x.Section.Name,
               DepartmentName = x.Section.Department.Name
           })
           .ToList();
```

```
SELECT ((("x"."surname" || ' ')) || "x"."name") AS "FullName",
    "x.Section"."name" AS "SectionName", "x.Section.Department"."name"
    AS "DepartmentName"
FROM "public"."employee" AS "x"
LEFT JOIN "public"."section" AS "x.Section" ON "x"."id_section" =
    "x.Section"."id_section"
LEFT JOIN "public"."department" AS "x.Section.Department" ON
    "x.Section"."id_department" = "x.Section.Department"."id_department"
```

Явный JOIN

```
var info = (from e in context.Employees
            join d in context.Employees on e.Section.Department.IdEmployee
                                        equals d.IdEmployee
               select new
               {
                   Employee = e.Surname,
                   Director = d.Surname
               });
```

```
SELECT "e"."surname" AS "Employee", "d"."surname" AS "Director"
FROM "public"."employee" AS "e"
LEFT JOIN "public"."section" AS "e.Section" ON "e"."id_section" =
    e.Section"."id_section"
LEFT JOIN "public"."department" AS "e.Section.Department" ON
    e.Section"."id_department" = "e.Section.Department"."id_department“
INNER JOIN "public"."employee" AS "d" ON "e.Section.Department"."id_employee" =
    "d"."id_employee"
```

Когда нужны не все поля

```
public class EmployeeModel
{
    public int Id { get; set; }

    public string FullName { get; set; }
}

var crew = context.Employees.Select(x => new EmployeeModel
           {
               Id = x.Id,
               FullName = x.Surname + " " + x.Name,
           })
           .ToList();
```

```
SELECT "x"."id" AS "Id",
       ((("x"."surname" || ' ')) || "x"."name") AS "FullName"
FROM "public"."employee" AS "x"
```

AutoMapper и проекции

```
public class EmployeeModel
{
    public int Id { get; set; }

    public string FullName { get; set; }
}

expression.CreateMap<Employee, EmployeeModel>()
   .ForMember(dst => dst.FullName, opt =>
       opt.MapFrom(src => $"{src.Surname} {src.Name}"));


var crew = context.Employees.ProjectTo<EmployeeModel>().ToList();
```

```
SELECT "dtoEmployee"."surname", "dtoEmployee"."name", "dtoEmployee"."id"
    FROM "public"."employee" AS "dtoEmployee"
```

Поддержка нескалярных типов

Массивы, hstore

```
public class Model
{
    // text[]
    public string[] KeyWords { get; set; }

    // integer[]
    public int[] Ids { get; set; }

    // timestamp[]
    public DateTime[] TimeStamps { get; set; }

    // hstore
    public Dictionary<string, string> ExtraFields { get; set; }
}
```

Поддержка нескалярных типов

JSONB

```
public class Account
{
    public string Login { get; set; }

    public string Password { get; set; }

    // jsonb
    public AccountDataModel ExtraData { get; set; }
}

public class AccountDataModel
{
    public string Name { get; set; }

    public string Surname { get; set; }

    public DateTime Birth { get; set; }
}
```

Поддержка нескалярных типов

JSONB

```
public class Account
{
    public string Login { get; set; }

    public string Password { get; set; }

    // jsonb
    public AccountDataModel ExtraData { get; set; }
}
```

```
modelBuilder.Entity<Account>()
    .Property(b => b.ExtraData)
    .HasConversion(
        v => JsonConvert.SerializeObject(v),
        v => JsonConvert.DeserializeObject<AccountDataModel>(v));
```

Поддержка нескалярных типов

Работает маппинг в обе стороны

Условия поиска по элементу не поддерживается на стороне ORM

Для поиска по элементам:

- Пишем запрос на SQL
- Маппинг – средствами ORM

Другие проблемные запросы

ORM не поддерживает рекурсивные запросы

Некоторые сложные конструкции

Другие проблемные запросы

```
SELECT [CustomerId], COUNT(*) AS [DealCount]
   FROM [Deal]
   GROUP BY [CustomerId]


var items = context.Deals
          .GroupBy(x => x.CustomerId)
          .Select(g => new
          {
              CustomerId = g.Key,
              DealCount = g.Count()
          })
          .ToList();
```

Другие проблемные запросы

```sql
 SELECT [CustomerId],
        COUNT(*) AS [DealCount],
        COUNT(DISTINCT CompanyId) AS [CompanyDealCount]
   FROM [Deal]
   GROUP BY [CustomerId]
```

```csharp
var items = context.Deals
      .GroupBy(x => x.CustomerId)
      .Select(g => new
      {
          CustomerId = g.Key,
          DealCount = g.Count(),
          // CompanyDealCount = g.DistinctCount(x => x.CompanyId)
          CompanyDealCount = g.Select(x => x.CompanyId).Distinct().Count()
      })
      .ToList();
```

Проблемные запросы

Нескалярные типы: поиск по элементу

Рекурсивные запросы

Некоторые сложные конструкции, например `COUNT(DISTINCT …)`

Open Source проекты готовы к нашим вкладам!

Миф: ORM – это просто

Разработчик всё равно должен знать SQL

Нужен специальный навык написания хороших запросов

Много тонкостей и дополнительных инструментов

SQL на выходе нужно проверять

Плюсы ORM

- Проверка синтаксиса
  - Особенно при изменении структуры
- Защита от SQL-инъекций
- Экранирование спецсимволов
- Тестирование на моках

Тесты и моки

Позитивный опыт: интеграционные тесты

Без реальной БД не обойтись

Код можно сгенерировать!

В .Net Core: InMemoryDatabase

Учимся
читать
и
писать
✓ SELECT

Batch UPDATE, DELETE

Как выполнить операции UPDATE и DELETE
без загрузки записи в приложение?

```
UPDATE employee SET salary = salary + 1000
  WHERE id_section = 2
```

Entity Framework Plus

http://entityframework-plus.net

```
context.Employees.Where(x => x.IdSection == 2)
    .Update(x => new Employee { Salary = x.Salary + 1000 });
```

```
UPDATE employee SET salary = salary + 1000
  WHERE id_section = 2
```

* Бывает полезно и для одной записи

Entity Framework Plus

Удаление

```
context.Employees.Where(x => x.Salary > 100000).Delete();
```

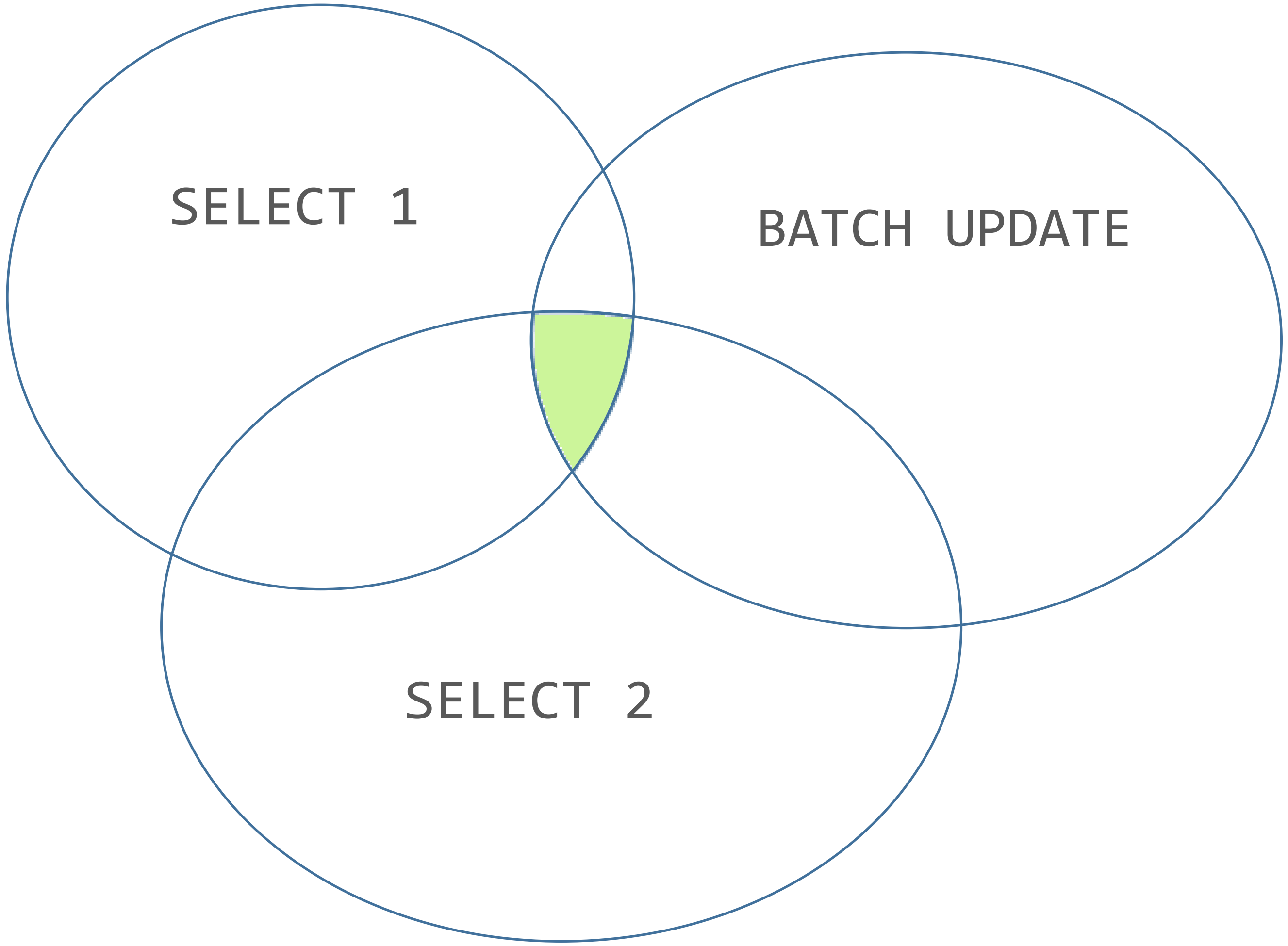
```
DELETE FROM employee WHERE salary > 100000
```

Batch UPDATE и кэш

```
var dude = context.Employees
           .Single(x => x.Id == 10);

Console.WriteLine(x => x.Salary);


context.Employees.Where(x => x.IdSection == 2)
    .Update(x => new Employee { Salary = x.Salary + 1000 });

dude = context.Employees
           .Single(x => x.Id == 10);

Console.WriteLine(x => x.Salary);
```

```
50000.0
50000.0
```

Batch UPDATE и кэш



Batch UPDATE и кэш

Сделан BatchUpdate (изменения уже в БД)
Но в кэше EF старая версия

Сбросить кэш:

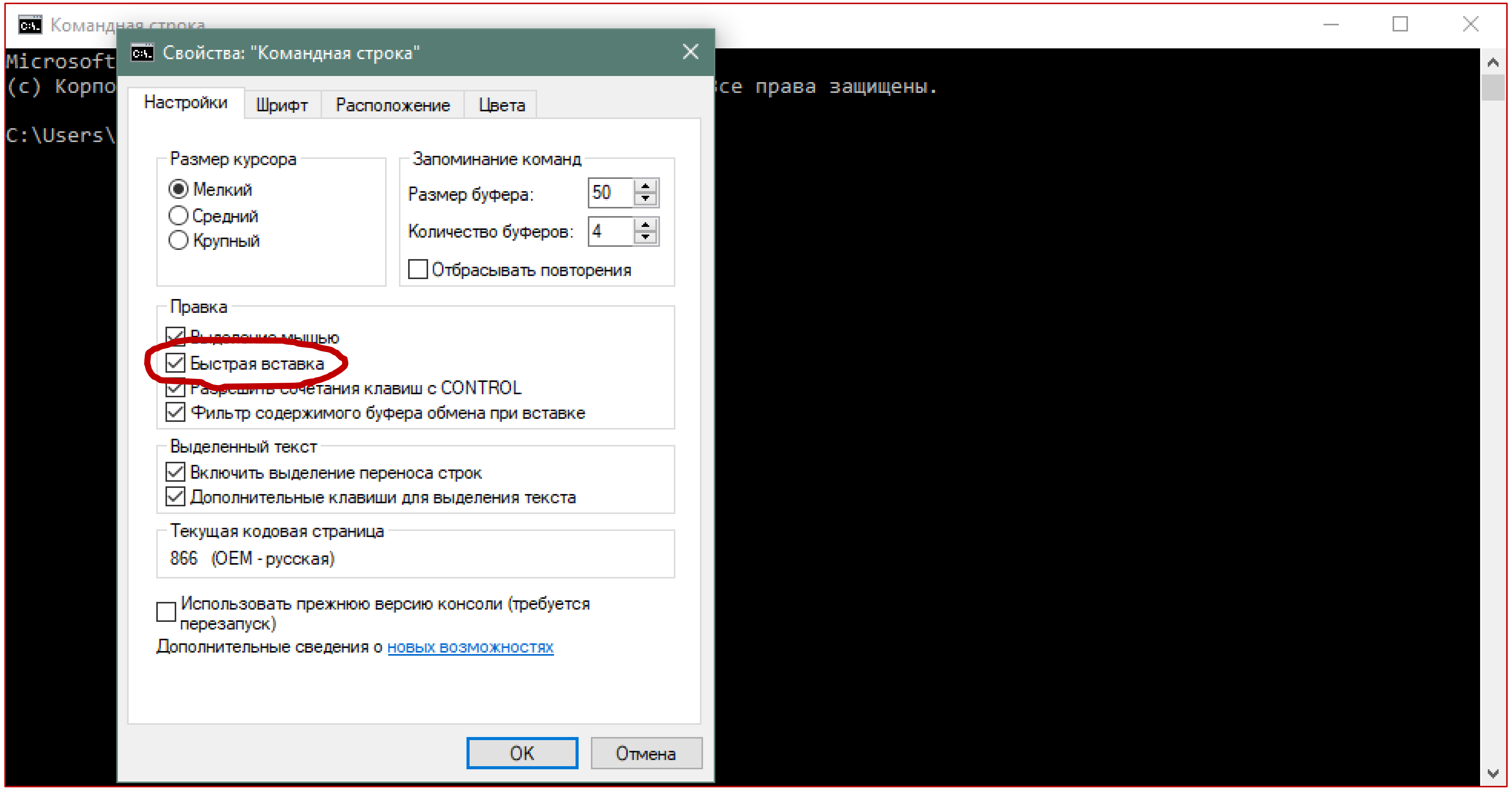
```
foreach (var entity in context.Employees.Local)
{
    context.Entry(entity).State =
        EntityState.Detached;
}
```

Mock для batch UPDATE

```
context.Employees.Where(x => x.IDSection == 2)
    .Update(x => new Employee { Salary = x.Salary + 1000 });
```

InMemoryDatabase в .Net Core – не поддерживает

```csharp
public void BatchUpdate<TEntity>(Expression<Func<TEntity, TEntity>> updateExpression,
        Expression<Func<TEntity, bool>> filerExpression,
        IQueryable<TEntity> items) where TEntity: class
{
    var entities = items.Where(filerExpression).ToList();

    var memberInitExpression = updateExpression.Body as MemberInitExpression;

    foreach (MemberBinding binding in memberInitExpression.Bindings)
    {
        string propertyName = binding.Member.Name;

        var memberAssignment = binding as MemberAssignment;

        var memberExpression = memberAssignment.Expression;

        object value;

        if (memberExpression.NodeType == ExpressionType.Constant)
        {
            var constantExpression = (ConstantExpression)memberExpression;

            value = constantExpression.Value;
        }
        else
        {
            LambdaExpression lambda = Expression.Lambda(memberExpression, null);
            value = lambda.Compile().DynamicInvoke();
        }

        var prop = typeof(TEntity).GetProperties()
            .First(x => x.Name == propertyName);

        foreach (var entity in entities)
        {
            prop.SetValue(entity, value);
        }
    }
}
```


Командная строка

```
Microsoft
(c) Корпо                                          се права защищены.

C:\Users\
```

Свойства: "Командная строка"

Настройки | Шрифт | Расположение | Цвета

Размер курсора
- [✓ Мелкий]
- Средний
- Крупный

Запоминание команд
- Размер буфера: 50
- Количество буферов: 4
- [ ] Отбрасывать повторения

Правка
- [✓] Выделение мышью
- [✓] Быстрая вставка
- [✓] Разрешить сочетания клавиш с CONTROL
- [✓] Фильтр содержимого буфера обмена при вставке

Выделенный текст
- [✓] Включить выделение переноса строк
- [✓] Дополнительные клавиши для выделения текста

Текущая кодовая страница

866 (OEM - русская)

- [ ] Использовать прежнюю версию консоли (требуется перезапуск)

Дополнительные сведения о новых возможностях

OK | Отмена

Вставка 100000 записей

Entity Framework (AddRange)

LINQ to DB (BulkCopy)

Вставка. EF vs LinqToDb

Entity Framework (AddRange)

```
00:00:22.506
00:00:22.740
```

LINQ to DB (BulkCopy)

```
00:00:20.083
00:00:19.900
```

x1.13

Вставка. EF 6 vs EF Core

Entity Framework (AddRange)

```
00:00:35.452
00:00:36.406
```

2777/с

EF Core (AddRange)

```
00:00:16.928
00:00:17.121
```

5882/с

x2.11

2006 г. – разработка системы FTP-поиска

В качестве сервера обычный ПК под Windows 2003

Одна и та же СУБД для записи и чтения

Несколько миллионов записей

MySQL: отказ на 100000 записей

PostgreSQL: стабильная работа, производительность

Multi-insert

```
INSERT INTO mytable (id, name)
    VALUES (1, 'Дмитрий'), (2, 'Анастасия') …
```

Колоссальный выигрыш в производительности!

Советы экспертов

Использовать можно!

Не стоит вставлять миллион записей одним запросом

Сложность диагностики ошибки

* Специфика PostgreSQL

Entity Framework (AddRange)

```
00:00:35.452
00:00:36.406
```

2777/c

EF Core (AddRange)

```
00:00:16.928
00:00:17.121
```

5882/c

Multi-insert (1 млн)

```
00:00:18.583
00:00:19.558
```

52632/c

x18.9 vs EF 6

x8.9 vs EF Core

Генерация SQL для Multi-insert

С помощью Reflection

Снижение производительности

Хардкод

Требуется для каждого класса Entity

```
public override string InsertSql
{
    get
    {
        System.Text.StringBuilder sb = new System.Text.StringBuilder();
        sb = sb.Append("(");
        if ((this.ID > 0))
        {
            sb = sb.Append(this.ID.ToString());
        }
        else
        {
            sb = sb.Append("nextval(\'test1_id_seq\')");
        }
        sb = sb.Append(", ");
        if ((this.Name != null))
        {
            sb = sb.Append("\'");
            sb = sb.Append(this.Name.Replace("\'", "\'\'"));
            sb = sb.Append("\'");
        }
        else
        {
            sb = sb.Append("null");
        }
        sb = sb.Append(", ");
        sb = sb.Append(this.Number.ToString());
        sb = sb.Append(")");
        return sb.ToString();
    }
}
```

Этот код сгенерирован автоматически

Генерация SQL для Multi-insert

С помощью Reflection

Снижение производительности

```
entity.GetType().GetProperties()

prop.GetValue(entity, null);
```

.Net Framework

Хардкод

```
00:00:00.553
00:00:00.596
```

Reflection

```
00:00:00.983
00:00:01.043
```

.Net Core

Хардкод

```
00:00:00.284
00:00:00.286
```

Reflection

```
00:00:00.58
00:00:00.582
```

Thursday, 12 January 2012

Playing with your member

(and: introducing FastMember)

Toying with members. We all do it. Some do it slow, some do it fast.

I am of course talking about the type of flexible member access that you need regularly in data-binding, materialization, and serialization code – and various other utility code.

Background

Here's standard member access:

```
Foo obj = GetStaticTypedFoo();
obj.Bar = "abc";
```

Not very exciting, is it? Traditional static-typed C# is very efficient here when everything is known at compile-time. With C# 4.0, we also get nice support for when the target is not known at compile time:

```
dynamic obj = GetDynamicFoo();
obj.Bar = "abc";
```

Looks much the same, eh? But what about when the *member* is not known? What we can't do is:

```
dynamic obj = GetStaticTypedFoo();
string propName = "Bar";
obj.propName = "abc"; // does not do what we intended!
```

So, we find ourselves in the realm of reflection. And as everyone knows, reflection is *slooooooooow*. Or at least, it is normally; if you don't object to talking with Cthulhu you can get into the exciting realms of meta-programming with tools like `Expression` or `ILGenerator` – but most people like keeping hold of their sanity, so… what to do?

Middle-ground

A few years ago, I threw together `HyperDescriptor`; this is a custom implementation of the `System.ComponentModel` representation of properties, but using some IL instead of reflection –

Stack Overflow

Marc Gravell ♦
756,850
●188 ●2,086 ●2,510

@marcgravell

Subscribe

Posts

Comments

Blog Archive

- ► 2018 (10)
- ► 2017 (7)
- ► 2016 (5)
- ► 2015 (4)
- ► 2014 (16)
- ► 2013 (4)
- ▼ 2012 (9)
  - ► October (1)
  - ► September (1)
  - ► August (1)
  - ► July (3)
  - ► June (1)
  - ► February (1)
  - ▼ January (1)
    - Playing with your member

.Net Framework

Хардкод

00:00:00.553
00:00:00.596

Reflection

00:00:00.983
00:00:01.043

FastMember

00:00:00.664
00:00:00.701

.Net Core

Хардкод

00:00:00.284
00:00:00.286

Reflection

00:00:00.58
00:00:00.582

FastMember

00:00:00.398
00:00:00.405

Генерация SQL для Multi-insert

С помощью Reflection → Вполне годится

Особенно с FastMember

Хардкод

```
context.ExecuteMultiInsert(list);
```

PostgreSQL 9.5+ “UPSERT”

```
INSERT … ON CONFLICT (…) DO UPDATE …
```

Можно использовать с Multi-insert

```
INSERT INTO mytable (id, name)
    VALUES (1, 'Дмитрий'), (2, 'Анастасия’) …
    ON CONFLICT (id) DO UPDATE
        SET name=excluded.name;
```

Multi-upsert!

Вставка, Multi-insert (1 млн)

```
00:00:13.983
00:00:14.256
```

70822/с

Обновление, Multi-upsert (1 млн)

```
00:00:10.858
00:00:11.286
```

90293/с

Результаты зависят от конфигурации:
Где-то вставка работает чуть быстрее, чем обновление

Код (Multi-upsert)

Сейчас:

```
context.ExecuteMultiInsert(list,
    "ON CONFLICT (id) DO UPDATE SET number=excluded.number;");
```

Планы:

```
context.ExecuteMultiUpsert(list,
    x => new Item { Number = x.Number });
```

Оператор COPY

Аналог в MS SQL – BULK INSERT, негативный опыт

Выполнять команду COPY с файлом разрешено только суперпользователям

Оператор COPY

Выполнять COPY с STDIN разрешено всем пользователям

Вызывает все триггеры и обрабатывает все ограничения-проверки

Работает с транзакциями

Поддерживается провайдером Npgsql

Оператор COPY

```
using (var writer = conn
  .BeginBinaryImport("COPY items (number, name) FROM STDIN (FORMAT BINARY)"))
{
    foreach (var item in list)
    {
        writer.StartRow();

        writer.Write(item.Number, NpgsqlDbType.Integer);
        writer.Write(item.Name);
    }

    writer.Complete();
}
```

Entity Framework Core (AddRange)

```
00:01:33.83
00:01:34.29
```

10631/c

Вставка, Multi-insert

```
00:00:09.95
00:00:09.82
```

101215/c

Вставка, COPY

```
00:00:02.49
00:00:02.40
```

408164/c

x4.03 vs MultiInsert

x38.39 vs EF Core

Оператор COPY + UPDATE

Для COPY нет конструкции `ON CONFLICT (…) DO UPDATE …`

Решение:

COPY во временную таблицу

UPSERT из временной таблицы в целевую

Оператор COPY + UPDATE

```
context.Database.ExecuteSqlCommand(
    "CREATE TEMP TABLE tmp_items (LIKE items) ON COMMIT DROP;");

using (var writer = conn
  .BeginBinaryImport("COPY tmp_items (number, name) FROM STDIN (FORMAT BINARY)"))
{
    foreach (var item in list)
    {
        writer.StartRow();

        writer.Write(item.Number, NpgsqlDbType.Integer);
        writer.Write(item.Name);
    }

    writer.Complete();
}

context.Database.ExecuteSqlCommand(@"INSERT INTO items SELECT * FROM tmp_items
   ON CONFLICT (id) DO UPDATE SET number=excluded.number;");
```

Вставка, COPY (1 млн)

```
00:00:02.49
00:00:02.40
```

Обновление, COPY + UPSERT

```
00:00:06.23
00:00:05.77
```

x1.25 vs MultiUpsert

Обновление, MultiUpsert

```
00:00:07.61
00:00:07.47
```

Итоги

Пишем в LINQ, мыслим в SQL!

Проверяйте сгенерированный SQL

Используйте AsNoTracking

EF + Dapper/LinqToDb

Библиотека Entity Framework Plus для batch UPDATE/DELETE

Вставка и обновление больших данных: генерируем SQL

MultiInsert/Upsert

COPY

Контакты

FadeevAS@sibedge.com

https://vk.com/fadeev

https://www.facebook.com/alexey.fadeev.3745

Ссылки

Entity Framework Core tips and tricks, Артур Дробинский

https://www.youtube.com/watch?v=aTE1DTHzE2o

Генерация кода, мои наработки

https://github.com/alexeyfadeev/dblinq2007/tree/ef

ВСЁ!