SolarLab>_

Что нового в EF Core 9: Работа с NoSQL и MongoDB Provider

Виктор Дзицкий

Давайте познакомимся

Виктор Дзицкий

SolarLab

TeamLead, .Net Developer

Telegram: *@Dzitskiy*

О чём поговорим

- Обзор новинок EF Core 9
- Поддержка NoSQL баз данных
- Нативная интеграция с MongoDB

Обзор новинок EF Core 9

What's New

- Azure Cosmos DB for NoSQL
- AOT and pre-compiled queries
- LINQ and SQL translation
- Migrations
- Model building
- Tooling

What's New

- **Azure Cosmos DB for NoSQL** → **Breaking changes (Discriminator)**
- AOT and pre-compiled queries
- LINQ and SQL translation
- Migrations
- Model building
- Tooling

Было:

```
{
    "id": "Blog|1099",
    ...
}
```

Стало:

```
{
    "id": 1099,
    "$type": "Blog",
    ...
}
```

https://learn.microsoft.com/en-us/ef/core/what-is-new/ef-core-9.0/breaking-changes#azure-cosmos-db-breaking-changes

What's New

- Azure Cosmos DB for NoSQL
- **AOT and pre-compiled queries** → *Experimental... expecting* **EF 10**
- LINQ and SQL translation
- Migrations
- Model building
- Tooling

Precompiled queries without NativeAOT:

dotnet ef dbcontext optimize --precompile-queries

https://learn.microsoft.com/en-us/ef/core/performance/nativeaot-and-precompiled-queries

What's New

- Azure Cosmos DB for NoSQL
- AOT and pre-compiled queries
- **LINQ and SQL translation** → Complex types support , etc.
- Migrations
- Model building
- Tooling

Поддержка составных типов

```
var groupedAddresses = await context.Stores
    .GroupBy(b => b.StoreAddress)
    .Select(g => new { g.Key, Count = g.Count() })
    .ToListAsync();


var newAddress = new Address("Gressenhall Farm Shop", null, "Beetley",
"Norfolk", "NR20 4DR");


await context.Stores
    .Where(e => e.Region == "Germany")
    .ExecuteUpdateAsync(s => s.SetProperty(b => b.StoreAddress, newAddress));
```

https://github.com/dotnet/EntityFramework.Docs/blob/main/samples/core/Miscellaneous/NewInEFCore9

Параметризированные примитивные коллекции

C#:

```
async Task<List<Post>>
GetPostsForceConstantCollection(int[] ids)
    => await context.Posts
        .Where(
            e => e.Title == ".NET Blog" &&
EF.Constant(ids).Contains(e.Id))
        .ToListAsync();
```

→

SQL:

```
Executed DbCommand (5ms)
[Parameters=[@__ids_0='[1,2,3]' (Size = 4000)],
CommandType='Text', CommandTimeout='30']
SELECT [p].[Id], [p].[Archived], [p].[AuthorId],
[p].[BlogId], [p].[Content], [p].[Discriminator],
[p].[PublishedOn], [p].[Rating], [p].[Title],
[p].[PromoText], [p].[Metadata]
FROM [Posts] AS [p]
WHERE [p].[Title] = N'.NET Blog' AND [p].[Id] IN (
    SELECT [i].[value]
    FROM OPENJSON(@__ids_0) WITH ([value] int '$')
AS [i]
)
```

https://github.com/dotnet/EntityFramework.Docs/blob/main/samples/core/Miscellaneous/NewInEFCore9

Семантическое сравнение с Null

C#:

```
var negatedNullableComparisonFilter =
await context.Entities
    .Where(x => !(x.NullableIntOne >
x.NullableIntTwo))
    .Select(x => new { x.NullableIntOne,
x.NullableIntTwo }).ToListAsync();
```

→

SQL:

```
SELECT [e].[NullableIntOne], [e].[NullableIntTwo]
FROM [Entities] AS [e]
WHERE CASE
    WHEN [e].[NullableIntOne] >
[e].[NullableIntTwo] THEN CAST(0 AS bit)
    ELSE CAST(1 AS bit)
END = CAST(1 AS bit)
```

https://github.com/dotnet/EntityFramework.Docs/blob/main/samples/core/Miscellaneous/NewInEFCore9

What's New

- Azure Cosmos DB for NoSQL
- AOT and pre-compiled queries
- LINQ and SQL translation
- **Migrations** → Protection against concurrent migrations!
- Model building
- Tooling

https://learn.microsoft.com/en-us/ef/core/what-is-new/ef-core-9.0/whatsnew#migrations

What's New

- Azure Cosmos DB for NoSQL
- AOT and pre-compiled queries
- LINQ and SQL translation
- Migrations
- **Model building** → Auto-compiled models, etc.
- Tooling

Автоматические компилируемые модели

```
dotnet ef dbcontext optimize
```

```
public virtual
Microsoft.EntityFrameworkCore.DbContext
OptionsBuilder<TContext> UseModel
(Microsoft.EntityFrameworkCore.Metadata
.IModel model);
```

→

```
dotnet add package
Microsoft.EntityFrameworkCore.Tasks
--version 9.0.0
```

```
<PropertyGroup>
    <EFOptimizeContext>true</EFOpti
mizeContext>
    <EFScaffoldModelStage>build</EF
ScaffoldModelStage>
</PropertyGroup>
```

https://github.com/dotnet/EntityFramework.Docs/tree/main/samples/core/Miscellaneous/NewInEFCore9.CompiledModels/

What's New

- Azure Cosmos DB for NoSQL
- AOT and pre-compiled queries
- LINQ and SQL translation
- Migrations
- Model building
- **Tooling** → no-build option

*dotnet ef migrations bundle **--no-build** --project …*

https://github.com/dotnet/efcore/issues/31750

Поддержка NoSQL баз данных

DB-Engines Ranking

423 systems in ranking, January 2025

Rank			DBMS	Database Model	Score		
Jan 2025	Dec 2024	Jan 2024			Jan 2025	Dec 2024	Jan 2024
1.	1.	1.	Oracle	Relational, Multi-model	1258.76	-5.03	+11.27
2.	2.	2.	MySQL	Relational, Multi-model	998.15	-5.61	-125.31
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	798.55	-7.14	-78.05
4.	4.	4.	PostgreSQL	Relational, Multi-model	663.41	-2.97	+14.45
5.	5.	5.	MongoDB	Document, Multi-model	402.50	+2.12	-14.98
6.	↑ 7.	↑ 9.	Snowflake	Relational	153.90	+6.54	+27.98
7.	↓ 6.	↓ 6.	Redis	Key-value, Multi-model	153.36	+3.08	-6.03
8.	8.	↓ 7.	Elasticsearch	Multi-model	134.92	+2.60	-1.15
9.	9.	↓ 8.	IBM Db2	Relational, Multi-model	122.97	+0.19	-9.43
10.	10.	↑ 11.	SQLite	Relational	106.69	+4.97	-8.51

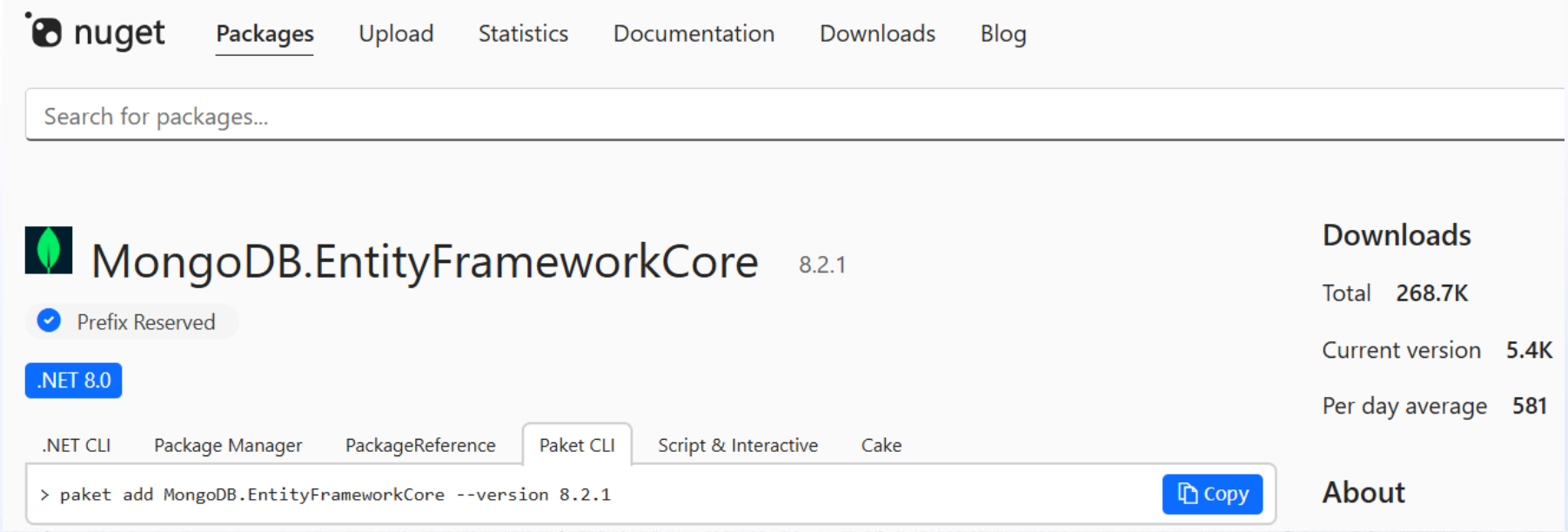
https://db-engines.com/en/ranking

Преимущества MongoDB

- Гибкость (бессхемная модель).
- Масштабируемость (горизонтальное масштабирование).
- Высокая производительность для больших объемов данных.

Нативная интеграция с MongoDB

Entity Framework Core & MongoDB



MongoDB Driver

Прямое взаимодействие через MongoDB Driver.

```
var client = new MongoClient("mongodb://localhost:27017");
var database = client.GetDatabase("MyDatabase");
var collection = database.GetCollection<User>("Users");
var users = await collection.Find(u => u.Age > 25).ToListAsync();
```

Плюсы:

- Полный контроль над запросами.
- Высокая производительность.

MongoDB.EntityFrameworkCore

Установка пакета:

```
dotnet add package MongoDB.EntityFrameworkCore
```

Настройка DbContext:

```
public class MyDbContext : DbContext
{
    public DbSet<User> Users { get; set; }

    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
    {
        optionsBuilder.UseMongoDB("mongodb://localhost:27017", "MyDatabase");
    }
}
```

Выполнение запросов

LINQ-запросы:

```
var users = await _context.Users.Where(u => u.Age > 25).ToListAsync();
```

Вставка данных:

```
var user = new User { Name = "John", Age = 30 };
await _context.Users.AddAsync(user);
await _context.SaveChangesAsync();
```

Стоит ли переходить на MongoDB.EntityFrameworkCore?

Преимущества:

- Единый API и знакомые паттерны EF Core
- Автоматическое отслеживание изменений (Change Tracking)
- Интеграция с экосистемой .NET
- Транзакции и оптимистическая блокировка
- Гибкость схемы и Code-First

Ограничения и рекомендации

Ограничения:

- Не все функции EF Core поддерживаются.
- Ограниченная поддержка сложных запросов.

Рекомендации:

- Используйте MongoDB .NET Driver для сложных сценариев.
- Для простых CRUD-операций используйте EF Core Provider.

Ссылки на источники

- What's New in EF Core 9
- Breaking changes in EF Core 9
- Entity Framework Core 9 Preview 5 - Release Notes
- NewInEFCore9 | GitHub
- Entity Framework Core - Database Providers
- MongoDB with C#
- MongoDB C# Driver
- MongoDB Entity Framework Core Provider
- mongo-efcore-provider | GitHub
- MongoDB.EntityFrameworkCore 8.2.1

Мои контакты

Спасибо за внимание!