Erweiterte Funktionen in EF Core

EF Core bietet erweiterte Funktionen zur Verbesserung der Funktionalität und Leistung.

Erläutern Sie die Vorteile und Anwendungsfälle von Query Tags (Hinzufügen von Metadaten zu Abfragen für Debugging und Leistungsanalyse),

Global Filters (Anwendung von Filtern auf alle Abfragen für einen bestimmten Entitätstyp, z. B. Soft Deletes) und

Stored Procedures (Ausführung von vorkompilierter Datenbanklogik für bestimmte Operationen).

Analysieren Sie, wie diese Funktionen die Wartbarkeit des Codes, die Sicherheit und die Leistung verbessern können. Untersuchen Sie mögliche Nachteile, wie die Komplexität der Verwaltung von Stored Procedures und die Auswirkungen globaler Filter auf die Abfrageleistung. Diskutieren Sie Szenarien, in denen diese Funktionen besonders nützlich sind, und zeigen Sie anhand von Beispielen, wie man sie effektiv implementieren kann. Untersuchen Sie, wie jedes Feature seine eigenen Vorteile hat. Erkunden Sie die Verwendung von Roh-SQL. Lesen Sie mehr

Query Tags

- Abfrage-Tags sind eine Funktion in EF Core 5.0 und höher, die es Ihnen ermöglicht, Ihre Abfragen mit Tags zu versehen. Diese können bei der Fehlersuche und beim Debugging nützlich sein, da die Tags in den Protokollen erscheinen und Ihnen helfen können, die Quelle einer Abfrage zu identifizieren.
- Verwendung:

```
var query = context.Posts
    .TagWith("Get recent posts")
    .Where(p => p.CreatedDate >= DateTime.Today.AddDays(-7))
    .ToList();
```

- Vorteile:
  - Hilft bei der Identifizierung und Korrelation von Abfragen mit Code in Protokollen.
  - Nützlich bei der Leistungsoptimierung und Analyse von Datenbankabfragen.

Globale Filter

- Globale Abfragefilter sind eine leistungsstarke Möglichkeit, Filter zu definieren, die automatisch auf Entitätstypen in EF Core angewendet werden. Dies ist besonders nützlich für die Implementierung von Funktionen wie Multi-Tenancy oder Soft Deletes.
- Verwendung:
  - Definieren Sie einen globalen Filter in der OnModelCreating Methode Ihres DbContext:

```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    modelBuilder.Entity().HasQueryFilter(p => !p.IsDeleted);
}
```

- Vorteile:
  - Wendet Filter automatisch global an und gewährleistet so eine einheitliche Datenzugriffslogik in der gesamten Anwendung.
  - Vereinfacht den Code durch Beseitigung sich wiederholender Filterlogik.
- Erwägungen:
  - Overhead durch zusätzliche Abfragelogik, wenn sie nicht sorgfältig eingesetzt wird.
  - Achten Sie auf komplexe Filterlogik, die die Abfrageleistung beeinträchtigt.

Stored Procedures

- EF Core unterstützt die Ausführung von Roh-SQL-Abfragen und -Befehlen, einschließlich gespeicherter Prozeduren. Sie können die Ergebnisse gespeicherter Prozeduren auf Entitäten abbilden oder sie zur Datenmanipulation ausführen.
- Verwendung:
  - Um eine Stored Procedure auszuführen, ist in diesem Fall eine Abfrage:

```
var results = context.Posts
    .FromSqlInterpolated($"EXEC GetRecentPosts @p0={days}")
    .ToList();
```

```
context.Database.ExecuteSqlInterpolated($"EXEC DeleteOldPosts @p0={date}");
```

  - Man kann aber auch SPs verwenden, um Operationen auf Daten anzustoßen (non-query stored procedure):

- Vorteile:
  - Nutzung der vorhandenen, optimierten Datenbanklogik.
  - Potenzielle Leistungsvorteile aufgrund der Wiederverwendung von Ausführungsplänen und datenbankspezifischen Optimierungen.
- Erwägungen:
  - Erhöhte Komplexität bei der Pflege sowohl des Anwendungscodes als auch der Datenbankprozeduren.
  - Herausforderungen beim Debuggen und Testen im Vergleich zu LINQ-Abfragen.