Tracking and Saving Data with EF Core 6

Julie Lerman

Most Trusted Authority on Entity Framework Core

@julielerman thedatafarm.com

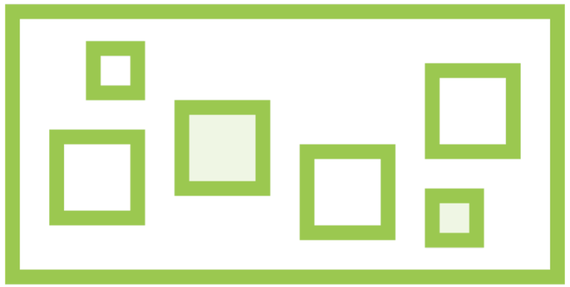
Module Overview

- DbContext and Entity roles in tracking
- Tracking and saving workflow
- Inserting, updating and deleting objects
- Tracking and saving multiple objects
- Bulk operation support
- How persistence differs in disconnected apps

Gaining a Better Understanding of DbContext and Entity

A fundamental understanding of how things work will pay off in productivity.

Tracking Components



DbContext

Represents a session with the database

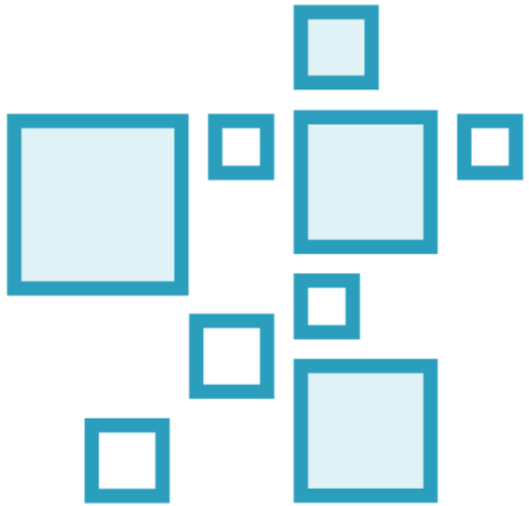
DbContext.ChangeTracker

Manages a collection of EntityEntry objects

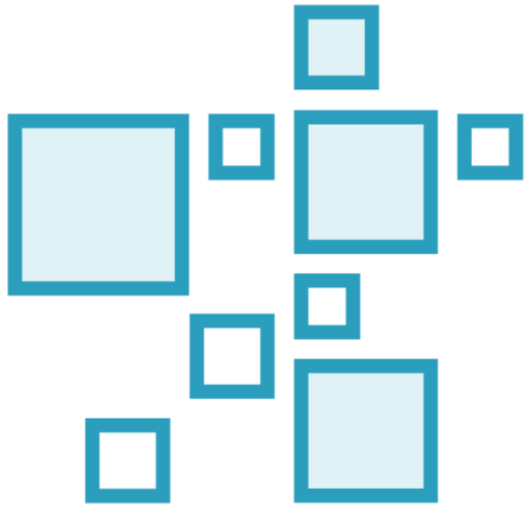
EntityEntry

State info for each entity: CurrentValues, OriginalValues, State enum, Entity & more

Entity and the DbContext

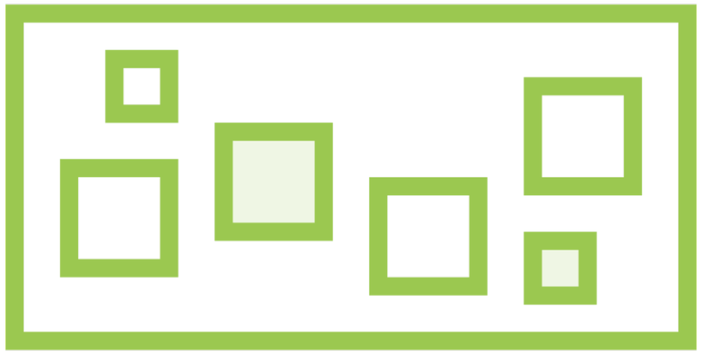


In-Memory Objects



Entities

In-memory objects with key (identity) properties that the DbContext is aware of

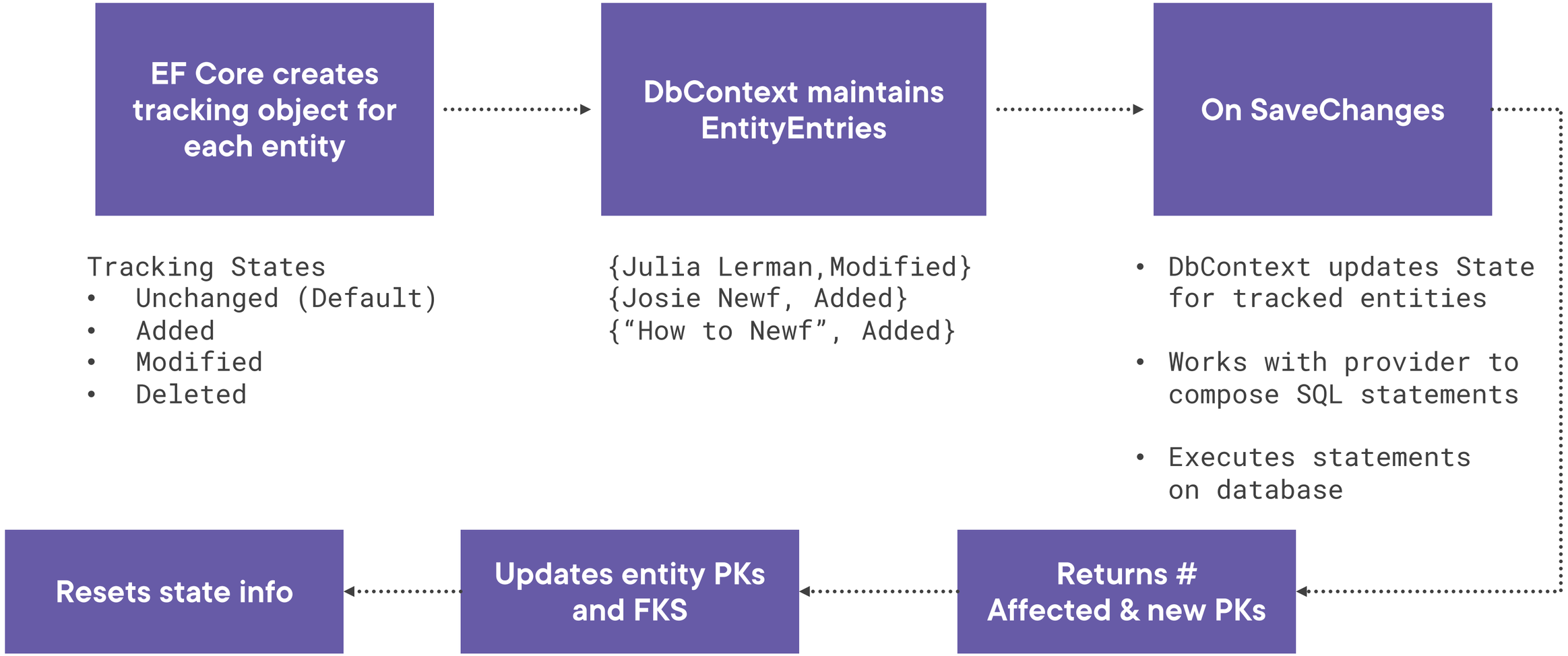


DbContext

Contains EntityEntry objects with reference pointers to in-memory objects

Understanding Tracking and Saving Workflow

Tracking and Saving Workflow



Inserting Simple Objects

Add from DbSet or DbContext

```
context.Authors.Add(…)
```

Track via DbSet

DbSet knows the type
DbContext knows that it's *Added*

```
context.Add(…)
```

Track via DbContext

DbContext will discover type
Knows it's *Added*

Updating Simple Objects

How DbContext Discovers About Changes

DbContext.ChangeTracker.DetectChanges()
Reads each object being tracked and updates state details in EntityEntry object

DbContext.SaveChanges()
Always calls DetectChanges for you

You can call DetectChanges() in your code if needed

The entities are just plain objects and don’t communicate their changes to the DbContext.

Various Paths for Tracking to Detect Edits

If DbContext is aware of object, it will detect changes on SaveChanges

DbSet and DbContext also have an Update() method

More advanced: modify state directly

Updating Untracked Objects

Begin Tracking and Set State to Modified

```
context.Authors.Update(…)
```

Track via DbSet

DbSet knows the type
Knows right away that it's Modified

```
context.Update(…)
```

Track via DbContext

DbContext will discover the type
Knows right away that it's *Modified*

More Pointers for Disconnected Data

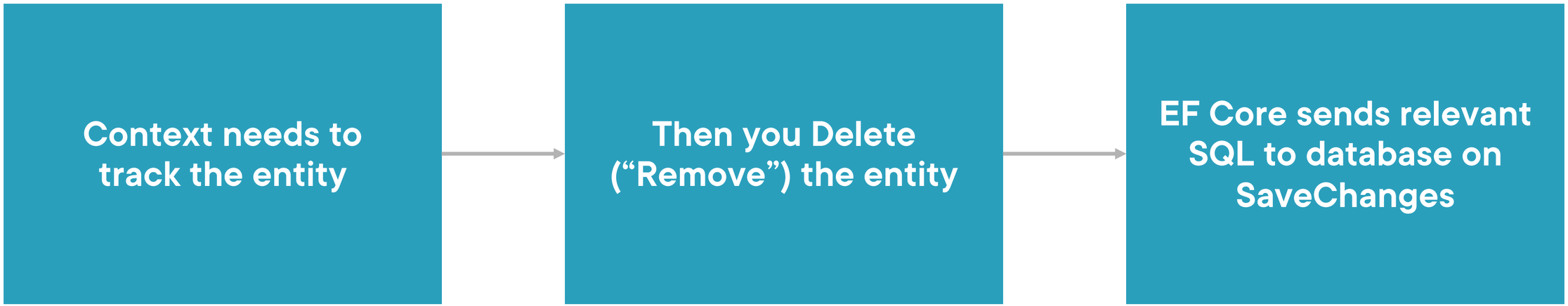
Update is not typically needed when object is tracked

Update and Add and Remove are important for untracked objects

Your app logic will need to know when to call Update, Add or Remove

Deleting Simple Objects

Deleting May Seem a Little Weird



There is no
Delete(id)
similar to Find(id)

Deleting without Querying for a Tracked Entity

Fake object with key property filled
Watch out for possible side effects

Stored procedure via EF Core raw SQL
Further on in this course

Soft delete via Global Query Filters
Link in resources

EF Core 7

A “bulk update” feature in EF Core 7 will help

Tracking Multiple Objects and Bulk Support

AddRange Method

```
_context.Authors.AddRange(author1,author2,author3,author4, etc.)
```

Tracking Multiple Entities

```
context.Authors.AddRange(…)
context.Authors.UpdateRange(…)
context.Authors.RemoveRange(…)
```

Track via DbSet

DbSet indicates type

```
context.AddRange(…)
context.UpdateRange(…)
context.RemoveRange(…)
```

Track via DbContext

Context will discover type(s)

Combined EF Core + SQL Server Effort

Individual Commands <=3
Faster to send individual commands for 1-3 entities

Batched Commands 4+
Faster to send batched commands for 4+ entities

Batching can combine
types and operations

Default Batch Constraints of SQL Server Provider

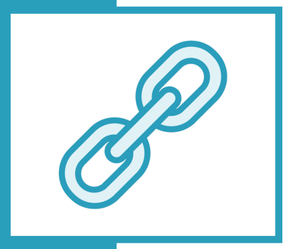
```
public class SqlServerModificationCommandBatch : AffectedCountModificationCommandBatch
{
    private const int DefaultNetworkPacketSizeBytes = 4096;
    private const int MaxScriptLength = 65536 * DefaultNetworkPacketSizeBytes / 2;
    private const int MaxParameterCount = 2100;
    private const int MaxRowCount = 1000;
    private int _parameterCount = 1; // Implicit parameter for the command text
    private readonly int _maxBatchSize;
    private readonly List<IReadOnlyModificationCommand> _bulkInsertCommands = new();
    private int _commandsLeftToLengthCheck = 50;
```

Review

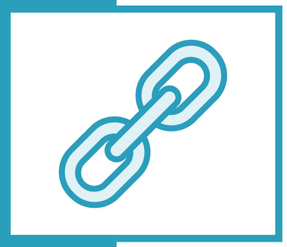
- How tracking and saving works
- Roles of DbContext and entities when tracking and saving
- Inserting, updating and deleting data
- Explicit updates for untracked objects
- Tracking and saving multiple objects with the Range methods
- Batching support

Up Next:
Controlling Database Creation and Schema Changes with Migrations

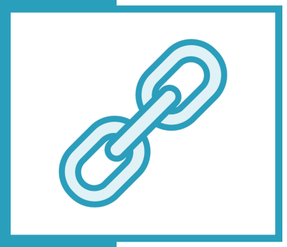
Resources



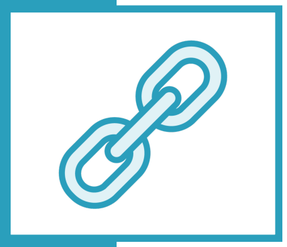
Entity Framework Core on GitHub: github.com/dotnet/efcore



EF Core Documentation: docs.microsoft.com/ef

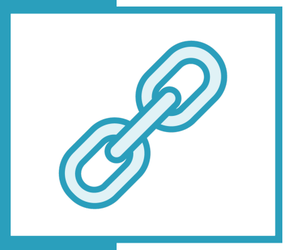


Article about SQL Server Merge Joins: brentozar.com/archive/2017/05/case-entity-framework-cores-odd-sql/



EF Core 6: Fulfilling the Bucket List: codemag.com/Article/2111072

Resources Cont.



Soft Delete in EF Core Docs:
docs.microsoft.com/en-us/ef/core/querying/filters