

Dr. Holger Schwichtenberg

# **Moderne Datenzugriffslösungen mit Entity Framework Core 9.0**

**Datenbankprogrammierung /  
Objekt-Relationales Mapping  
mit C# in .NET 8.0 und .NET 9.0**



---

Buchversion:	13. 0.2 (03.11.2024)
Verlag:	<a href="http://www.IT-Visions.de">www.IT-Visions.de</a> , Fahrenberg 40b, D-45257 Essen
Sprachliche Korrektur:	Dorotha Fleischer, Heike Rickert, Matthias Bloch (M.A.)
ISBN:	978-3-934-27928-5
Formate:	Druck, Kindle, PDF, PDF-Abo
Bezugsquellen:	<a href="http://www.IT-Visions.de/Buch/EFC13">www.IT-Visions.de/Buch/EFC13</a>

*Für Heidi, Felix und Maja*

# 1 Inhaltsverzeichnis (Hauptkapitel)

1	Inhaltsverzeichnis (Hauptkapitel) .....	4
2	Inhaltsverzeichnis (Details) .....	5
3	Vorwort.....	22
4	Über den Autor .....	24
5	Über dieses Buch .....	26
6	Fallbeispiele in diesem Buch .....	31
7	Programmcodebeispiel zum Download .....	41
8	Entity Framework Core-Basisinformationen .....	44
9	Installation von Entity Framework Core .....	72
10	Konzepte von Entity Framework Core .....	75
11	Reverse Engineering bestehender Datenbanken .....	80
12	Forward Engineering für neue Datenbanken .....	121
13	Anpassung des Datenbankschemas.....	147
14	Datenbankschemamigrationen .....	179
15	Daten lesen mit LINQ.....	212
16	Objektbeziehungen und Ladestrategien .....	253
17	Einfügen, Löschen und Ändern .....	288
18	Datenänderungskonflikte (Concurrency) .....	322
19	Protokollierung (Logging) .....	335
20	Asynchrone Programmierung .....	349
21	Dynamische LINQ-Abfragen.....	354
22	Daten lesen und ändern mit SQL, Stored Procedures und Table Valued Functions .....	359
23	Weitere Tipps und Tricks zum Mapping .....	388
24	Weitere Tipps und Tricks zu LINQ und SQL .....	540
25	Leistungsoptimierung (Performance Tuning) .....	569
26	Zusatzwerkzeuge .....	632
27	Zusatzzkomponenten .....	677
28	Softwarearchitektur mit Entity Framework Core .....	705
29	Praxislösungen .....	735
30	Migration von ADO.NET Entity Framework zu Entity Framework Core .....	800
31	Quellen im Internet .....	803
32	Versionsgeschichte dieses Buchs .....	804
33	Stichwortverzeichnis (Index) .....	805
34	Werbung in eigener Sache ☺ .....	821

## 2 Inhaltsverzeichnis (Details)

1	Inhaltsverzeichnis (Hauptkapitel).....	4
2	Inhaltsverzeichnis (Details).....	5
3	Vorwort .....	22
4	Über den Autor.....	24
5	Über dieses Buch.....	26
5.1	Versionsgeschichte dieses Buchs .....	26
5.2	Hinweis zu den Vertriebswegen.....	26
5.3	Bezugsquelle des PDF-E-Books für Amazon-Kunden .....	26
5.4	Bezugsquelle für Aktualisierungen .....	27
5.5	Hinweise zur Breite und Tiefe dieses Buchs – Sie haben Einfluss! .....	27
5.6	Geplante Kapitel .....	27
5.7	Programmiersprache in diesem Buch.....	29
5.8	Notwendige Vorkenntnisse .....	29
5.9	Hinweis auf einen Bug im PDF-Exporter von Word .....	29
5.10	Ihre Belohnung, wenn Sie helfen, dieses Buch zu verbessern! .....	30
6	Fallbeispiele in diesem Buch.....	31
6.1	Entitäten .....	31
6.2	Englische Version des Beispiels .....	34
6.3	Anwendungarten in diesem Buch .....	36
6.4	Hilfsklasse zur Konsolenausgabe (CUI) .....	36
7	Programmcodebeispiel zum Download.....	41
7.1	Webadresse für Downloads.....	41
7.2	Übersicht über die Beispiele .....	41
7.3	Qualitätssicherung der Programmcodebeispiele .....	43
8	Entity Framework Core-Basisinformationen.....	44
8.1	Erscheinungszyklus.....	44
8.2	Namenschaos .....	44
8.3	Was ist ein Objekt-Relationaler Mapper (ORM)?.....	45
8.4	ORM in der .NET-Welt .....	46
8.5	Versionsgeschichte von Entity Framework Core .....	47
8.6	Marktlage .....	48
8.7	Unterstützte Betriebssysteme .....	49
8.8	Unterstützte .NET-Versionen.....	50
8.9	Unterstützte Visual Studio-Versionen.....	50
8.10	Unterstützte Datenbanken .....	50
8.11	NuGet-Pakete.....	52
8.12	Kommandozeilenwerkzeuge .....	54

8.12.1	PowerShell-Commandlets und ef.exe .....	54
8.12.2	.NET CLI-Erweiterung .....	54
8.13	Neuerungen in Entity Framework Core 9.0 .....	55
8.13.1	Plattformen für Entity Framework Core 9.0.....	55
8.13.2	Neue Funktionen in Entity Framework Core 9.0 .....	55
8.14	Breaking Changes in Entity Framework Core 9.0 .....	56
8.15	Verbliebene Schwächen in Entity Framework Core 9.0 .....	56
8.16	Vergleich zwischen Entity Framework 6.5 und Entity Framework Core 9.0 .....	57
8.16.1	Funktionen, die dauerhaft entfallen sind .....	58
8.16.2	Neue Funktionen in Entity Framework Core .....	58
8.16.3	Detailvergleich.....	59
8.16.4	Zusammenfassende Bewertung.....	65
8.17	Kritik an der Entwicklungsgeschwindigkeit von Entity Framework Core .....	66
8.18	Ausblick auf zukünftige Erscheinungstermine .....	66
8.19	Einsatzszenarien für Entity Framework Core .....	66
8.20	Dokumentation für Entity Framework Core .....	67
8.21	Mitarbeit beim Open Source-Projekt.....	68
8.22	Support für Entity Framework Core .....	70
9	Installation von Entity Framework Core .....	72
9.1	NuGet-Pakete .....	72
9.2	Paketinstallation .....	72
9.3	Aktualisierung auf eine neue Version.....	73
10	Konzepte von Entity Framework Core .....	75
10.1	Vorgehensmodelle bei Entity Framework Core.....	75
10.2	Artefakte bei Entity Framework Core .....	78
11	Reverse Engineering bestehender Datenbanken .....	80
11.1	Reverse Engineering-Werkzeuge .....	80
11.2	NuGet-Pakete für das Reverse Engineering .....	80
11.3	Projektstruktur für das Reverse Engineering .....	81
11.4	Beispieldatenbank.....	84
11.5	Codegenerierung starten .....	85
11.6	Generierter Programmcode.....	87
11.7	Namensänderungen .....	96
11.8	Beispiel-Client.....	96
11.9	Lediglich Teile einer Datenbank generieren.....	98
11.10	Trennung von Kontextklasse und Entitätsklassen .....	98
11.11	Konfiguration der Datenbankverbindung .....	99
11.11.1	Verwenden von Konfigurationsdateien in .NET Framework .....	100
11.11.2	Verwenden von Konfigurationsdateien in .NET Core .....	100
11.11.3	Übergabe der Verbindungszeichenfolge .....	100
11.11.4	Verwendung von DbContextOptions .....	101

---

11.11.5	Eigenes Datenbankverbindungsmanagement .....	102
11.11.6	Flexible Konfiguration der Kontextklasse.....	103
11.12	Thread-Sicherheit.....	103
11.13	EF Core-Kommandozeilenwerkzeug (dotnet ef).....	103
11.14	Anpassung der Codegenerierung mit T4-Vorlagen.....	104
11.14.1	Keine Vorlagen in Entity Framework Core 1.0 bis 6.0 .....	104
11.14.2	T4-Vorlagen für Entity Framework Core 7.0 installieren.....	105
11.14.3	Kein eingebauter T4-Editor .....	107
11.14.4	Codegenerierung anstoßen .....	108
11.14.5	Aufbau von T4.....	108
11.14.6	Anpassung der Vorlagen .....	109
11.15	Manuelle Anpassung des generierten Programcodes .....	115
11.15.1	Codegenerierung ohne Vererbungsbeziehungen .....	115
11.15.2	Schönere Objektmodelle mit Vererbung .....	116
11.15.3	Manuelle Nachbearbeitung des generierten Programmcodes .....	118
11.16	Schwächen des Reverse Engineering .....	119
12	Forward Engineering für neue Datenbanken.....	121
12.1	Zwei Klassentypen beim Forward Engineering .....	121
12.2	Beispiele in diesem Kapitel.....	121
12.3	Projektstruktur für das Forward Engineering.....	121
12.4	Regeln für die selbsterstellten Entitätsklassen .....	123
12.4.1	NuGet-Pakete .....	123
12.4.2	Klassenarten .....	123
12.4.3	Konstruktoren.....	124
12.4.4	Properties.....	124
12.4.5	Datentypen.....	124
12.4.6	Null-Werte erlauben (Nullable/Nullability).....	125
12.4.7	Non-Nullable Reference Types (Nullable Context).....	125
12.4.8	Aufzählungstypen (Enumerationen) .....	133
12.4.9	Beziehungen (Master-Detail).....	133
12.4.10	Vererbung.....	136
12.4.11	Primärschlüssel.....	136
12.4.12	Beispiele .....	136
12.5	Regeln für die selbsterstellte Kontextklasse.....	139
12.5.1	NuGet-Pakete .....	139
12.5.2	Basisklasse.....	139
12.5.3	Konstruktor.....	139
12.5.4	Verweise zu den Entitätsklassen.....	140
12.5.5	Provider und Verbindungszeichenfolge.....	140
12.5.6	Beispiel für eine selbsterstellte Kontextklasse.....	140
12.6	Regeln für die Datenbankschemagenerierung.....	141
12.7	Beispiel-Client .....	141
12.8	Anpassung per Fluent-API (OnModelCreating()).....	142

12.9	Das erzeigte Datenmodell .....	144
13	Anpassung des Datenbankschemas .....	147
13.1	Beispiele in diesem Kapitel .....	147
13.2	Konvention versus Konfiguration .....	147
13.3	Persistente versus transiente Klassen .....	148
13.4	Namen im Datenbankschema .....	149
13.4.1	Änderung der Tabellen- und Spaltennamen .....	149
13.4.2	Breaking Change in EF Core 3.0 .....	150
13.5	Sortierreihenfolge der Werte .....	150
13.6	Datentypen und Spaltentypen .....	150
13.7	Mapping von Arrays und Listen elementarer Typen seit Entity Framework Core 8.0 .....	156
13.8	Anpassungen des Datentypmappings .....	158
13.9	Typkonvertierungen .....	159
13.10	Pflichtfelder und optionale Felder .....	159
13.11	Feldlängen .....	159
13.12	Primärschlüssel .....	159
13.13	Beziehungen und Fremdschlüssel .....	160
13.14	Optionale Beziehungen und Pflichtbeziehungen .....	161
13.15	Uni- und Bidirektionale Beziehungen .....	162
13.16	1:1-Beziehungen .....	163
13.17	N:M-Beziehungen .....	163
13.17.1	N:M Skip Navigation (Direkte N:M-Beziehungen) .....	165
13.17.2	Direkter Zugriff auf die Zwischentabelle .....	169
13.17.3	Namen der Zwischentabelle ändern .....	170
13.17.4	Explizite Klasse für die Zwischentabelle .....	171
13.18	Indexe festlegen .....	173
13.18.1	Verwendung von HasIndex() .....	173
13.18.2	Annotation [Index] .....	174
13.19	Syntaxoptionen für das Fluent-API .....	175
13.19.1	Sequentielle Konfiguration .....	175
13.19.2	Strukturierung durch Statement Lambdas .....	176
13.19.3	Strukturierung durch Unterroutinen .....	176
13.19.4	Strukturierung durch Konfigurationsklassen .....	177
13.20	Ausblick auf weitere Mapping-Einstellungen .....	178
14	Datenbankschemamigrationen .....	179
14.1	Informationen über die Datenbank .....	179
14.2	Prüfung der Datenbankexistenz .....	179
14.3	Anlegen der Datenbank zur Laufzeit .....	180
14.4	Schemamigrationen zur Entwicklungszeit .....	181
14.5	Befehle für die Schemamigrationen .....	181
14.6	Add-Migration .....	182

---

14.7	Get-Migration .....	185
14.8	Statusabfrage für ausstehende Schemamigrationen .....	185
14.9	Update-Database .....	186
14.10	Remove-Migration .....	189
14.11	Script-Migration .....	190
14.12	Get-DbContext .....	190
14.13	Commandlet-Optionen .....	190
14.14	Schemamigrationswerkzeuge in Visual Studio .....	191
14.15	Wechsel von Reverse Engineering auf Forward Engineering bei bestehenden Datenbanken 193	
14.16	Mehrere Kontextklassen in einem Projekt .....	196
14.17	Schemamigrationen zur Laufzeit der Anwendung (Installation oder Anwendungsstart).....	196
14.17.1	Verwendung von Migrate() .....	197
14.17.2	IMigrator-Service .....	198
14.17.3	Informationen zum Migrationsstand.....	198
14.17.4	Praxiseinsatz: Ein Kommandozeilenwerkzeug für die Schemamigration .....	199
14.17.5	Migration Bundles .....	201
14.18	Schemamigrationsszenarien .....	202
14.18.1	Neue Tabellen und Spalten .....	202
14.18.2	Tabellen oder Spalten löschen .....	203
14.18.3	Tabellen oder Spalten umbenennen .....	203
14.18.4	Spaltendatentyp ändern .....	204
14.18.5	NULL-Werte verbieten .....	204
14.18.6	Kardinalitäten ändern .....	205
14.18.7	Andere Datenbankartefakte anlegen .....	206
14.18.8	SQL-Skriptdateien ausführen .....	208
14.18.9	Eigenständige Entitäten bilden .....	208
14.19	Probleme bei der Schemamigration in Verbindung mit TFS .....	210
14.20	Startverhalten von Entity Framework Core .....	211
15	Daten lesen mit LINQ .....	212
15.1	Kontextklasse .....	212
15.2	LINQ-Abfragen .....	212
15.3	Schrittweise Zusammensetzung von LINQ-Abfragen .....	214
15.4	Übersichtlichere SQL-Generierung ohne überflüssige Klammern .....	215
15.5	Einsatz von var .....	215
15.6	Repository-Pattern .....	215
15.7	LINQ-Abfragen mit Paging .....	218
15.8	Projektionen .....	219
15.8.1	Projektion auf einen Entitätstypen .....	219
15.8.2	Projektionen auf einen anonymen Typen .....	221
15.8.3	Projektionen auf einen beliebigen Typen .....	222
15.9	Abfrage nach Einzelobjekten .....	223

15.10	Laden anhand des Primärschlüssels mit Find() .....	224
15.11	Gruppierungen mit GroupBy() .....	224
15.11.1	Gruppierungen seit Entity Framework Core 3.0 .....	225
15.11.2	Gruppieren nach Entitätstyp (seit Version 7.0) .....	226
15.11.3	Gruppieren zum Abschluss einer LINQ-Abfrage (seit Version 7.0) .....	227
15.12	Übersetzung von Contains() .....	228
15.13	LINQ im RAM statt in der Datenbank (Client-Evaluation).....	229
15.14	Falsche Befehlsreihenfolge.....	231
15.15	Eigene Funktionen in LINQ .....	232
15.15.1	Eigene Funktionen in LINQ in Entity Framework Core 1.x und 2.x .....	232
15.15.2	Eigene Funktionen in LINQ seit Entity Framework Core 3.0 .....	233
15.16	Kurzübersicht über die LINQ-Syntax.....	234
15.16.1	<a href="https://learn.microsoft.com/de-ch/samples/dotnet/try-samples/101-linq-samples/">https://learn.microsoft.com/de-ch/samples/dotnet/try-samples/101-linq-samples/</a> .....	235
15.16.2	Einfache SELECT-Befehle (Alle Datensätze) .....	235
15.16.3	Bedingungen (where) .....	236
15.16.4	Bedingungen mit Mengen (in) .....	236
15.16.5	Sortierungen (orderby) .....	236
15.16.6	Paging (Skip() und Take()) .....	236
15.16.7	Projektion .....	237
15.16.8	Aggregatfunktionen (Count(), Min(), Max(), Average(), Sum()).....	237
15.16.9	Gruppierungen (GroupBy).....	238
15.16.10	Einzelobjekte (SingleOrDefault(), FirstOrDefault()) .....	239
15.16.11	Verbundene Objekte (Include()) .....	239
15.16.12	Inner Join (Join) .....	240
15.16.13	Cross Join (Kartesisches Produkt).....	240
15.16.14	Join mit Gruppierung .....	241
15.17	Unter-Abfragen (Sub-Select) .....	241
15.17.1	15.17 Lokaler Objektzwischenspeicher in der Kontextklasse (First-Level-Object-Cache) .....	243
15.17.2	Wirkung des Zwischenspeichers .....	244
15.17.3	Neuladen veralteter Objekte (Reload).....	247
15.17.4	Neuladen gelöschter Objekte .....	247
15.17.5	Den Zwischenspeicher verwalten.....	250
15.18	Ein typischer Fehler bei Unit Tests .....	248
15.19	Kontextreset.....	251
15.19	Datenbindung .....	252
16	16 Objektbeziehungen und Ladestrategien .....	253
16.1	Überblick über die Ladestrategien.....	253
16.2	Standardverhalten .....	254
16.3	Lazy Loading.....	255
16.3.1	Aktivierung des Lazy Loading .....	255
16.3.2	Gefahren von Lazy Loading.....	256
16.3.3	Lazy Loading ohne Proxyklassen .....	258
16.3.4	Lazy Loading bei No-Tracking-Abfragen.....	260

16.3.5	Abschalten des Lazy Loading für einzelne Beziehungen .....	260
16.4	Explizites Nachladen (Explicit Loading) .....	260
16.4.1	Synchrones explizites Nachladen .....	260
16.4.2	Asynchrones explizites Nachladen .....	262
16.5	Eager Loading (Server Join) .....	262
16.5.1	Include() und ThenInclude().....	262
16.5.2	Anzahl der SQL-Befehle (Split Queries).....	264
16.6	Eager Loading mit Bedingungen (Filtered Includes) .....	269
16.6.1	Bedingungen bei Filtered Includes .....	270
16.6.2	Unerwartete Seiteneffekte .....	272
16.6.3	Sortierung und Paging bei Filtered Includes.....	272
16.7	Eager Loading per Auto-Include.....	274
16.8	Preloading (Client-Join).....	277
16.8.1	Beispiel für Fall 1 .....	277
16.8.2	Beispiel für Fall 2 .....	278
16.8.3	Beispiel für Fall 3 .....	279
16.9	Preloading mit Relationship Fixup (Client-Join) .....	280
16.10	Objektbeziehungen und lokaler Zwischenspeicher .....	284
16.10.1	Inhalt des Zwischenspeichers .....	286
16.10.2	Abfrage der Objekte im lokalen Zwischenspeicher.....	286
16.10.3	Säubern des Zwischenspeichers .....	287
17	Einfügen, Löschen und Ändern .....	288
17.1	Speichern mit SaveChanges() .....	288
17.1.1	Verhalten von SaveChanges() .....	288
17.1.2	Beispiel.....	289
17.1.3	Objektzustandsänderungen .....	290
17.2	Fehlerfälle beim Speichern .....	290
17.3	Änderungsverfolgung auch für Unterobjekte.....	291
17.4	Zusammenfassen von Befehlen (Batching).....	292
17.5	Das Foreach-Problem.....	293
17.6	Objekte hinzufügen mit Add() .....	295
17.7	Objekte verbinden .....	296
17.8	Verbundene Objekte ändern / Relationship Fixup .....	299
17.9	Widersprüchliche Beziehungen .....	301
17.10	Objekte löschen (Löschoperationen).....	304
17.10.1	Objekte löschen mit Remove() .....	304
17.10.2	Löschen mit einem Attrappen-Objekt.....	306
17.10.3	Massenlöschen.....	307
17.11	Datenbanktransaktionen .....	308
17.11.1	Transaktion in einer Kontextinstanz .....	308
17.11.2	Transaktion über mehrere Kontextinstanzen ohne TransactionScope .....	308
17.11.3	Transaktion über mehrere Kontextinstanzen mit TransactionScope.....	310

17.12	Change Tracker abfragen.....	312
17.12.1	POCOs .....	312
17.12.2	Zustand eines Objekts .....	312
17.12.3	Liste aller geänderten Objekte.....	314
17.13	Ereignisse bei Zustandsänderungen und beim Speichern .....	316
17.14	Fallbeispiel: CRUD-Operationen mit komplexem Objektbaum .....	317
18	Datenänderungskonflikte (Concurrency) .....	322
18.1	Keine Konflikterkennung .....	322
18.2	Optimistisches Sperren .....	323
18.3	Konflikterkennung durch Wertvergleich .....	324
18.4	Konflikterkennung für alle Spalten.....	325
18.5	Konflikteinstellung per Konvention .....	325
18.6	Fallweise Konflikteinstellung.....	326
18.7	Zeitstempel (Timestamp).....	327
18.8	Konflikte auflösen .....	329
18.9	Pessimistisches Sperren .....	332
19	Protokollierung (Logging) .....	335
19.1	Protokollierung mit UseLoggerFactory() .....	335
19.2	Filtern nach Level und Protokollkategorien .....	337
19.3	Übergabe der LoggerFactory .....	338
19.4	Ereignis-IDs.....	339
19.5	Erweiterungsmethode Log() .....	340
19.6	Implementierung der Log()-Erweiterungsmethode .....	342
19.7	Protokollierung mit LogTo() .....	345
19.8	EnableSensitiveDataLogging und EnableDetailedErrors .....	346
19.9	SQL-Ausgabe mit ToQueryString() .....	346
19.10	Debugger-Ansichten (DebugView) .....	347
20	Asynchrone Programmierung .....	349
20.1	Asynchrone Erweiterungsmethoden .....	349
20.2	ToListAsync().....	349
20.2.1	ToHashSetAsync() .....	350
20.3	SaveChangesAsync().....	350
20.4	ForeachAsync() .....	351
20.5	Asynchrone Streams mit AsAsyncEnumerable() .....	352
21	Dynamische LINQ-Abfragen.....	354
21.1	Schrittweises zusammensetzen von LINQ-Abfragen .....	354
21.2	Expression Trees .....	355
21.3	Dynamic LINQ .....	357
22	Daten lesen und ändern mit SQL, Stored Procedures und Table Valued Functions .....	359
22.1	Abfragen mit FromSql(), FromSqlInterpolated() und FromSqlRaw() .....	359

---

22.1.1	FromSqlInterpolated() .....	359
22.1.2	FromSqlRaw() .....	360
22.1.3	SQL-Injektionsangriffe.....	361
22.2	Projektionen mit SQL auf Entitätsklassen.....	362
22.3	Mapping von SQL-Abfragen auf beliebige Klassen .....	364
22.4	SQL-Abfragen, die primitive Typen liefern.....	366
22.5	Beliebige Resultsets mit SQL .....	367
22.5.1	Systeminformationen des Microsoft SQL Server .....	367
22.5.2	Anzahl der Datensätze pro Tabelle .....	368
22.5.3	Datenbankschema ausgeben .....	368
22.5.4	SqlQueryRaw() und SqlQuery() direkt im Database-Objekt (Ab Entity Framework Core 8.0) .....	371
22.6	Zusammensetbarkeit von LINQ und SQL.....	375
22.7	Erweiterungsmethode ExecuteSqlQuery() .....	376
22.8	Mapping von SQL-Ergebnissen auf primitive Typen .....	377
22.9	SQL-DML-Befehle ohne Resultset.....	377
22.10	Stored Procedures und Table Valued Functions .....	378
22.10.1	Stored Procedures mit FromSql() und FromSqlRaw() .....	378
22.10.2	Table Value Functions mit FromSqlRaw().....	379
22.10.3	Wrapper-Methoden für Table Valued Functions .....	379
22.10.4	Codegenerierung für Stored Procedures und Table Valued Functions .....	380
22.11	Stored Procedures bei SaveChanges().....	382
22.11.1	CUD-Mapping auf Stored Procedures im klassischen Entity Framework .....	382
22.11.2	CUD-Mapping auf Stored Procedures in Entity Framework Core .....	383
22.11.3	Parameter für die Stored Procedures festlegen .....	384
22.11.4	Beispiel: Einfügen, Ändern und Löschen in der Airline-Tabelle.....	384
22.11.5	Stored Procedures für CUD.....	385
22.11.6	Festlegung des Mappings auf Stored Procedures .....	386
23	Weitere Tipps und Tricks zum Mapping .....	388
23.1	Vererbung .....	388
23.1.1	Praxisszenario.....	388
23.1.2	Impedance Mismatch.....	390
23.1.3	Vergleich der Vererbungsabbildungsstrategien .....	392
23.1.4	Umsetzung mit Entity Framework Core 7.0 .....	393
23.1.5	TPH ist weiterhin der Standard.....	394
23.1.6	Konfiguration der Discriminator-Spalte .....	395
23.1.7	Im Ausnahmefall gibt es eine Mischung aus TPC und TPH .....	395
23.1.8	TPH erzwingen.....	397
23.1.9	Sparse Columns.....	397
23.1.10	TPT erzwingen .....	398
23.1.11	TPC erzwingen .....	398
23.1.12	Einschränkungen der neuen Methoden seit Version 7.0.....	398
23.1.13	Keine zusätzlichen Schlüssel.....	399

23.2 Tabellenaufteilung (Table Splitting).....	400
23.2.1 Table Splitting mit Entity Types .....	402
23.2.2 Table Splitting mit Owned Types (Aggregate) .....	405
23.2.3 Table Splitting mit Complex Types als Alternative zu Owned Types .....	415
23.3 JSON-Mapping.....	421
23.3.1 Beispielszenario .....	421
23.3.2 Owned Types ohne JSON-Mapping .....	424
23.3.3 Owned Types mit JSON-Mapping.....	425
23.3.4 LINQ-Abfragen beim JSON-Mapping.....	426
23.3.5 Weitere Einschränkungen .....	428
23.3.6 Weitere Optionen .....	428
23.3.7 Geschwindigkeitsmessungen .....	430
23.4 Entity Splitting.....	431
23.5 Temporale Tabellen.....	432
23.5.1 Temporale Tabellen im SQL Server aktivieren.....	433
23.5.2 Abfrage temporaler Tabellen .....	435
23.5.3 Temporale Tabellen mit Entity Framework Core .....	436
23.5.4 Direkte SQL-Abfragen auf temporale Tabellen .....	439
23.5.5 Schemaänderungen .....	439
23.5.6 Aufräumarbeiten .....	439
23.5.7 Festplattenverbrauch.....	440
23.6 Mapping auf Properties oder Fields.....	441
23.6.1 Beispielszenario .....	441
23.6.2 Mapping von Fields und privaten Properties.....	444
23.6.3 Fields bevorzugt beim Materialisieren .....	445
23.6.4 Abweichungen von den Konventionen .....	447
23.6.5 Zugriff auf Property erzwingen.....	447
23.6.6 Verwendung von privaten Mitgliedern in LINQ-Abfragen.....	448
23.7 Standardwerte (Default Values) .....	448
23.7.1 Standardwerte beim Forward Engineering festlegen.....	449
23.7.2 Standardwerte verwenden .....	449
23.7.3 Festlegen des Wächter-Wertes mit HasSentinel() .....	451
23.7.4 Praxistipp: Standardwerte schon beim Anlegen des Objekts vergeben.....	451
23.7.5 Standardwerte beim Reverse Engineering .....	452
23.8 Berechnete Spalten (Computed Columns).....	452
23.8.1 Automatisches SELECT .....	452
23.8.2 Praxistipp: Spalten mit einer Berechnungsformel anlegen.....	453
23.8.3 Spalten mit einer Berechnungsformel nutzen.....	454
23.8.4 Spalten mit einer Berechnungsformel beim Reverse Engineering .....	455
23.9 Wertkonvertierungen (Value Converter) .....	456
23.9.1 Einschränkungen.....	457
23.9.2 Beispiel 1: Konvertierung zwischen String und Boolean.....	457
23.9.3 Beispiel 2: Konvertierung zwischen Aufzählungstyp und String.....	460
23.9.4 Beispiel 3: Konvertierung zwischen einem Record-Typ und einer Zeichenkette .....	463

---

23.10 Spaltenreihenfolge .....	463
23.10.1 Spaltenreihenfolge mit Entity Framework Core .....	464
23.10.2 Spaltenreihenfolge setzen mit Entity Framework Core .....	466
23.10.3 Spaltenreihenfolge ändern in SQL Server Management Studio .....	467
23.10.4 Spaltenreihenfolge verändern bei Entity Framework Core.....	468
23.10.5 Andere Datenbankmanagementsysteme .....	469
23.11 Sequenzobjekte (Sequences) .....	470
23.11.1 Was sind Sequenzen? .....	470
23.11.2 Erstellen von Sequenzen mit T-SQL .....	470
23.11.3 Erstellen von Sequenzen beim Forward Engineering .....	472
23.11.4 Sequenzen im Einsatz.....	473
23.12 Alternative Schlüssel.....	476
23.12.1 Alternative Schlüssel definieren .....	476
23.12.2 Alternative Schlüssel im Einsatz .....	478
23.13 Shadow Properties .....	481
23.13.1 Automatische Shadow Properties .....	482
23.13.2 Festlegung eines Shadow Property .....	482
23.13.3 Ausgabe aller Shadow Properties einer Entitätsklasse .....	482
23.13.4 Lesen und Ändern eines Shadow Property .....	483
23.13.5 LINQ-Abfragen mit Shadow Properties .....	484
23.13.6 Praxisbeispiel: Automatisches Setzen bei jedem Speichern .....	484
23.13.7 Praxisbeispiel: Erweitern der Tabellen zur Betriebszeit der Anwendung.....	485
23.14 Indexer Properties .....	487
23.14.1 Indexer Properties zur Erweiterung von Entitätsklassen .....	487
23.14.2 Indexer Properties für dynamische Entitätsklassen .....	488
23.14.3 Beispiel für den Einsatz von Indexer Properties .....	489
23.15 Kaskadierendes Löschen (Cascading Delete) .....	492
23.15.1 Löschoptionen in Entity Framework Core.....	492
23.15.2 Beispiel.....	494
23.16 Abbildung von Datenbanksichten (Views) .....	498
23.16.1 Datenbanksicht anlegen .....	498
23.16.2 Entitätsklasse für die Datenbanksicht anlegen.....	499
23.16.3 Einbinden der Entitätsklasse in die Kontextklasse .....	499
23.16.4 Datenbanksichten bei der Schemamigration.....	500
23.16.5 Verwendung der Datenbanksicht.....	500
23.16.6 Änderungen persistieren in Datenbanksichten.....	501
23.17 Sichten auf Kontextebene (Defining Queries) .....	501
23.17.1 Entitätsklassen für Defining Queries .....	502
23.17.2 Registrierung der Entitätsklassen in der Kontextklasse .....	502
23.17.3 Abfragedefinition .....	502
23.17.4 Verwendung der Defining Queries .....	503
23.17.5 Alternativen zu Defining Queries .....	504
23.18 Hierarchische Daten im SQL Server.....	505
23.19 Selbstdokumentation im Datenbankschema.....	509

---

23.19.1	Kommentare im Microsoft SQL Server .....	509
23.19.2	Kommentare direkt eingeben .....	511
23.19.3	Verwendung der Kommentare in SQL Server Management Studio .....	511
23.19.4	Kommentare verwalten per T-SQL .....	512
23.19.5	Kommentare bei Forward Engineering .....	514
23.19.6	Kommentare bei Reverse Engineering .....	515
23.19.7	Kommentare in anderen Datenbankmanagementsystem.....	516
23.19.8	Kommentare in SQLite .....	516
23.20	Datenbefüllung bei der Schemamigration (Data Seeding).....	518
23.20.1	Verbindung von Objekten beim Data Seeding .....	520
23.21	Eigene Konventionen (Custom Conventions) und eigene Annotationen.....	521
23.21.1	Eigene Konventionen per Massenkonfiguration mit dem Fluent-API .....	522
23.21.2	Eigene Konventionen per IConvention .....	527
23.21.3	Eigene Konventionen konfigurieren mit ConfigureConventions() .....	536
23.22	Eingriffe in OR-Mapping-Abläufe mit Interceptoren .....	536
23.23	Erneute Ausführung von OnModelCreating() .....	538
24	Weitere Tipps und Tricks zu LINQ und SQL .....	540
24.1	Neue Übersetzung von LINQ in SQL in Entity Framework Core 9.0 .....	540
24.1.1	TimeOnly.FromDateTime() und TimeOnly.FromTimeSpan() .....	540
24.1.2	Math.Min() und Math.Max().....	540
24.1.3	TrimStart() und TrimEnd() .....	541
24.2	Globale Abfragefilter (seit Version 2.0) .....	541
24.2.1	Filter definieren.....	541
24.2.2	Filter nutzen .....	542
24.2.3	Praxistipp: Filter ignorieren .....	542
24.2.4	Globale Abfragefilter bei SQL-Abfragen (seit Version 2.0) .....	542
24.2.5	Globale Abfragefilter bei Stored Procedures und Table Valued Functions .....	543
24.3	Skalare Datenbankfunktionen.....	544
24.4	Zukünftige Abfragen (Future Queries) .....	546
24.4.1	Konzept der Future Queries .....	546
24.4.2	Methode Future() .....	546
24.4.3	Methode FutureValue().....	548
24.4.4	Bug in Verbindung mit EF Profiler .....	549
24.5	Befehlsverfolgung mit Query Tags (seit Version 2.2) .....	549
24.5.1	WithTag().....	550
24.5.2	Einsatz von TagWith() .....	550
24.5.3	Einschränkungen.....	555
24.6	Benachrichtigungen bei Datenänderungen (Query Notifications) .....	555
24.6.1	SqlDependency für Microsoft SQL Server .....	555
24.6.2	Aufbau des SQL-Befehls .....	556
24.6.3	Query Notification in einer Konsolenanwendungen .....	557
24.6.4	Diagnosemöglichkeiten.....	559
24.6.5	Query Notification in einer Desktop-Anwendungen.....	559

---

24.7	Automatische Aktualisierung von Datensätzen.....	563
24.8	Datenbank-Trigger mit Entity Framework Core .....	564
24.8.1	Trigger anlegen im Entity Framework Core-Client.....	565
24.8.2	Trigger anlegen in Schemamigrationen.....	565
24.8.3	Entity Framework Core muss wissen, dass es Trigger gibt .....	566
24.8.4	Community-Projekt für Trigger in C#-Syntax.....	567
24.8.5	Getriggerte Spalten automatisch aktualisieren.....	567
25	Leistungsoptimierung (Performance Tuning).....	569
25.1	Vorgehensmodell zur Leistungsoptimierung bei Entity Framework Core .....	569
25.2	Best Practices für Ihre eigenen Leistungstests .....	569
25.3	Leistungsvergleich verschiedener Datenzugriffstechniken in .NET .....	570
25.4	Allgemeine Optimierungsprinzipien für den Datenbankzugriff.....	572
25.5	Projektionen.....	573
25.6	Kontextinstanziierung optimieren.....	574
25.7	Leistungsoptimierung durch kompilierte Modelle .....	574
25.7.1	Manuelle Erstellung eines kompilierten Modells .....	575
25.7.2	Kompiliertes Modell aktivieren.....	576
25.7.3	Auto-Compiled Models seit Entity Framework Core 9.0 .....	577
25.7.4	Geschwindigkeitsmessungen.....	578
25.7.5	Nachteile der kompilierten Modelle .....	580
25.8	Optimierung durch Notification Entities .....	580
25.9	Objektzuweisungen optimieren.....	583
25.10	Massenoperationen .....	585
25.10.1	Einzellöschen.....	586
25.10.2	Optimierung durch Batching .....	586
25.10.3	Löschen ohne Laden mit Pseudo-Objekten .....	587
25.10.4	Einsatz von klassischem SQL anstelle des Entity Framework Core-APIs .....	588
25.10.5	Massenoperationen mit ExecuteUpdate() und ExecuteDelete() .....	590
25.10.6	Massenlöschen mit EFPlus.....	591
25.10.7	Masseneinfügen (Bulk Insert).....	591
25.11	Einfluss auf die Parametrisierung .....	591
25.12	Leistungsoptimierung durch No-Tracking .....	592
25.12.1	No-Tracking aktivieren.....	593
25.12.2	No-Tracking fast immer möglich .....	594
25.12.3	No-Tracking im änderbaren Datagrid.....	596
25.12.4	QueryTrackingBehavior und AsTracking() .....	604
25.12.5	Konsequenzen des No-Tracking-Modus .....	605
25.12.6	Identitätsfeststellung.....	606
25.12.7	Best Practices .....	608
25.13	Leistung SQLite vs. Microsoft SQL Server .....	609
25.14	Leistungsoptimierung durch Compiled Queries.....	609
25.14.1	Konzept einer Compiled Query .....	609

25.14.2	Compiled Queries in Entity Framework Core.....	610
25.14.3	Leistungstest .....	611
25.14.4	Einschränkungen.....	614
25.15	Auswahl der besten Ladestrategie .....	614
25.15.1	Keine pauschale Aussage möglich.....	614
25.15.2	Vorsicht bei automatischem Lazy Loading.....	614
25.15.3	Eager Loading versus Client-Join .....	615
25.16	Zwischenspeicherung (Caching).....	620
25.16.1	MemoryCache.....	621
25.16.2	CacheManager .....	622
25.17	Second-Level-Caching mit EFPlus.....	628
25.17.1	Einrichten des Second-Level-Cache .....	628
25.17.2	Verwenden des Second-Level-Cache.....	629
25.18	Entity Framework Core und der Ahead-of-Timer-Compiler (NativeAOT) .....	630
26	Zusatzwerkzeuge .....	632
26.1	Dotnet-Counters (Leistungsindikatoren) .....	632
26.2	Entity Framework Core Power Tools (EFPT) .....	636
26.2.1	Funktionsüberblick.....	637
26.2.2	Reverse Engineering mit Entity Framework Core Power Tools .....	638
26.2.3	Reverse Engineering von Stored Procedures .....	644
26.2.4	Schemamigrationen mit Entity Framework Core Power Tools.....	646
26.2.5	Diagramme mit Entity Framework Core Power Tools.....	647
26.3	LINQPad .....	649
26.3.1	Aufbau von LINQPad .....	650
26.3.2	Datenquellen einbinden.....	650
26.3.3	LINQ-Befehle ausführen.....	652
26.3.4	Abspeichern .....	654
26.3.5	Weitere LINQPad-Treiber .....	654
26.3.6	Interaktive Programmcodeeingabe.....	654
26.3.7	Fazit zu LINQPad .....	655
26.4	Entity Developer.....	655
26.4.1	Auswahl der ORM-Technik .....	656
26.4.2	Reverse Engineering mit Entity Developer.....	657
26.4.3	Forward Engineering mit Entity Developer .....	664
26.5	Entity Framework Profiler.....	668
26.5.1	Einbinden des Entity Framework Profilers .....	668
26.5.2	Befehle überwachen mit Entity Framework Profiler.....	669
26.5.3	Warnungen vor potenziellen Problemen .....	672
26.5.4	Analysefunktionen .....	673
26.5.5	Kommandozeilenunterstützung und API .....	673
26.5.6	Fazit zu Entity Framework Profiler.....	674
26.6	SQLite & SQL Server Compact Toolbox .....	674
27	Zusatzkomponenten .....	677

---

27.1	Oracle-Treiber von DevArt (dotConnect for Oracle).....	677
27.1.1	Unterstützte Oracle-Versionen .....	677
27.1.2	Installation .....	677
27.1.3	Visual Studio-Integration .....	678
27.1.4	Datenbanktreibername.....	680
27.1.5	Entity Framework Core-Werkzeuge.....	680
27.1.6	Kontextklasse .....	680
27.1.7	Entitätsklassen .....	681
27.1.8	Datentypen.....	681
27.2	Entity Framework Plus (EFPlus).....	682
27.3	Second-Level-Caching mit EFSecondLevelCache.Core.....	683
27.4	Objekt-Objekt-Mapping mit AutoMapper .....	684
27.4.1	Objekt-Objekt-Mapping per Reflection.....	685
27.4.2	AutoMapper.....	687
27.4.3	Beispieldaten.....	687
27.4.4	Abbildungen konfigurieren.....	688
27.4.5	Abbildung ausführen mit Map() .....	690
27.4.6	Abbildungskonventionen.....	690
27.4.7	Abbildungskonventionen ändern .....	691
27.4.8	Profilklassen .....	692
27.4.9	Verbundene Objekte .....	692
27.4.10	Manuelle Abbildungen .....	693
27.4.11	Typkonvertierungen .....	695
27.4.12	Objektmengen .....	696
27.4.13	Vererbung.....	697
27.4.14	Generische Klassen .....	699
27.4.15	Zusatzaktionen vor und nach dem Mapping.....	701
27.4.16	Geschwindigkeit.....	702
27.4.17	Fazit zu AutoMapper .....	703
27.5	Andere Erweiterungen .....	703
28	Softwarearchitektur mit Entity Framework Core .....	705
28.1	Rolle von Entity Framework Core in der Softwarearchitektur.....	705
28.1.1	Monolithisches Modell.....	705
28.1.2	Entity Framework Core als Datenzugriffsschicht.....	705
28.1.3	Reine Geschäftslogik.....	707
28.1.4	Geschäftsobjekt- und ViewModel-Klassen .....	708
28.1.5	Verteilte Systeme.....	708
28.1.6	Fazit.....	711
28.2	Entity Framework Core in verteilten Systemen (N-Tier-Szenarien / Detached Objects).....	711
28.2.1	Techniken für Verteilte Systeme in .NET .....	712
28.2.2	Detached Objects (Disconnected Objects) .....	712
28.2.3	Herausforderung.....	713
28.2.4	Detached Objects in 2-Tier-Anwendungen .....	714
28.2.5	Laden von Objekten .....	714

---

28.2.6	Objekte ändern .....	718
28.2.7	Punktuelle Aktualisierung .....	718
28.2.8	Nachrichtenbasierte Methodensignaturen .....	720
28.2.9	Vollständige Objektaktualisierung .....	721
28.2.10	Objektaktualisierung mit Datentransferobjekten .....	722
28.2.11	Änderungsverfolgung auf dem Client .....	723
28.2.12	Objekte löschen .....	725
28.2.13	Objektbäume persistieren .....	728
28.2.14	Objekte mit Autowert-Primärschlüssel .....	729
28.2.15	Objekte ohne Autowert-Primärschlüssel .....	731
28.2.16	TrackGraph() .....	732
28.2.17	State in jedem Entitätsobjekt .....	734
28.2.18	Open Data Protocol (OData) als Alternative .....	734
29	Praxislösungen .....	735
29.1	Entity Framework Core in einem ASP.NET Core-Backend .....	735
29.1.1	Das Fallbeispiel "MiracleList" .....	735
29.1.2	Architektur .....	738
29.1.3	Entitätsklassen .....	741
29.1.4	Entity Framework Core-Kontextklasse .....	743
29.1.5	Lebensdauer der Kontextklasse in ASP.NET Core-Anwendungen .....	745
29.1.6	Geschäftslogik .....	745
29.1.7	WebAPI .....	752
29.1.8	Verwendung von Entity Framework Core per Dependency Injection .....	760
29.1.9	Praxistipp: Kontextinstanzpooling (DbContext Pooling) .....	762
29.2	DevOps mit Entity Framework (Continous Integration und Continous Delivery) .....	763
29.2.1	Unit Tests und Integrationstests mit Entity Framework Core .....	763
29.2.2	In-Memory-Treiber .....	764
29.2.3	SQLite-In-Memory-Treiber .....	766
29.2.4	Entity Framework Core beim serverseitigen Build (Continous Integration) .....	767
29.2.5	Entity Framework Core beim automatischen Release (Continous Delivery) .....	770
29.3	Fallbeispiel Cross-Platform-App "MiracleList Light" .....	771
29.3.1	GUI-Frameworks .....	771
29.3.2	Funktionen der App .....	772
29.3.3	Datenbankunterstützung .....	775
29.3.4	Softwarearchitektur von MiracleList Light .....	775
29.3.5	Entitätsklassen im Projekt BO .....	777
29.3.6	Entity Framework Core-Kontextklasse im Projekt DAL .....	778
29.3.7	Konfigurationsdatenübergabe via Dependency Injection (IEnv) .....	779
29.3.8	Geschäftslogik im Projekt BL .....	781
29.3.9	Startcode der Anwendung .....	783
29.3.10	Implementierung von IEnv .....	784
29.3.11	Benutzeroberfläche .....	784
29.3.12	Benutzerschnittstellensteuerung .....	787
29.3.13	Einblicke in die erzeigte SQLite-Datenbank .....	792

---

29.4	N:M-Beziehungen zu sich selbst.....	794
30	Migration von ADO.NET Entity Framework zu Entity Framework Core .....	800
30.1	Pro und Contra Migration .....	800
30.2	Koexistenz (Hybride Strategie).....	800
30.3	Migration von Entity Framework Code First zu Entity Framework Core.....	801
30.4	Migration von EDMX zu Entity Framework Core.....	802
31	Quellen im Internet.....	803
32	Versionsgeschichte dieses Buchs .....	804
33	Stichwortverzeichnis (Index).....	805
34	Werbung in eigener Sache ☺ .....	821
34.1	Dienstleistungen.....	821
34.2	Aktion "Buch für Buchrezension".....	822
34.3	Angebot "PDF-Buch-Abo" .....	823

### 3 Vorwort

Liebe Leserinnen und Leser,

ich nutze Entity Framework in echten Softwareentwicklungsprojekten seit der allerersten Version, also seit der Version 1.0 von ADO.NET Entity Framework aus dem Jahr 2008. Zuvor hatte ich einen selbstentwickelten Objekt-Relationalen Mapper in meinen .NET-Projekten verwendet. Entity Framework Core ist das Nachfolgeprodukt, das es seit 2016 gibt. Ich setzte seitdem auch (aber nicht ausschließlich) Entity Framework Core in der Praxis ein. Einige meiner Projekte laufen noch mit dem klassischen Entity Framework.

Microsoft entwickelt Entity Framework Core inkrementell, d.h., die ersten Versionen waren noch sehr unvollständig. Mit jeder neuen Version schließt Microsoft weitere Lücken. Inzwischen hat Entity Framework Core eine gute Reife erreicht – wenngleich es immer noch einzelne Schwächen (wie in jeder Software) gibt, und diese thematisiert dieses Buch natürlich ebenfalls.

Genau so wie sich Entity Framework Core weiterentwickelt hat, so hat sich auch dieses Fachbuch seit der ersten Version im September 2016 kontinuierlich weiterentwickelt. Die vor Ihnen liegende Ausgabe beschreibt alle Kernaspekte und viele Tipps und Tricks sowie Praxisszenarien zu Entity Framework Core einschließlich der **Version 9.0**. Ich plane, in Zukunft weitere Versionen dieses Buchs zu veröffentlichen, die die kommenden Versionen von Entity Framework Core behandeln werden und auch weitere Tipps & Tricks sowie Praxisszenarien ergänzen.

Das Buch setzt aber voraus, dass Sie .NET, C# und Visual Studio schon kennen. Ich biete dazu aber auch weitere Bücher an, siehe Kapitel "Notwendige Vorkenntnisse".

Dieses Fachbuch wird vertrieben auf folgenden Wegen (ich nenne neben dem Verkaufspreis auch, wie viel – bzw. wenig – ich als Autor von den jeweiligen Händlern erhalte. Der Rest ist Gewinn der Händler):

- Gedruckt bei Amazon.de für 69,99 Euro (der Autor erhält 25,30 Euro):  
[www.amazon.de/exec/obidos/ASIN/3934279295/itvisions-21](http://www.amazon.de/exec/obidos/ASIN/3934279295/itvisions-21)
- Kindle-E-Book bei Amazon.de für 59,99 Euro (der Autor erhält 19,62 Euro):  
[www.amazon.de/exec/obidos/ASIN/B0CMBZW766/itvisions-21](http://www.amazon.de/exec/obidos/ASIN/B0CMBZW766/itvisions-21)
- PDF-E-Book bei Leanpub.com ab 49,99 Dollar (der Autor erhält 39,99 Dollar):  
[www.leanpub.com/EntityFrameworkCore9](http://www.leanpub.com/EntityFrameworkCore9)
- Als Teil des E-Book-Buch-Abos zusammen mit anderen aktuellen Fachbüchern für 99,00 Euro im Jahr **inkl. aller Updates** (der Autor erhält den kompletten Preis):  
[www.IT-Visions.de/BuchAbo](http://www.IT-Visions.de/BuchAbo)

**Tipp:** Käufer bei Leanpub.com können jederzeit Aktualisierungen des PDF-Buchs (gleiche Hauptversion) kostenfrei dort beziehen. Käufer bei Amazon erhalten die PDF-Ausgabe einmalig kostenfrei (siehe Kapitel "Über dieses Fachbuch").

Da diese Preise in Anbetracht der vielen Stunden Arbeit an diesem Buch leider nicht nennenswert dazu beitragen können, den Lebensunterhalt meiner Familie zu bestreiten, ist dieses Projekt ein Hobby. Dementsprechend kann ich nicht garantieren, wann es Updates zu diesem Buch geben wird. Ich werde dann an diesem Buch arbeiten, wenn ich neben meinem Beruf als Softwarearchitekt, Berater und Dozent und meinen sportlichen Betätigungen noch etwas Zeit für das Fachbuchautorenhobby übrig habe.

Falls mir in diesem Buch oder den zugehörigen Downloads menschliche Fehler passiert sind, möchte ich mich dafür schon jetzt in aller Form bei Ihnen entschuldigen. Bitte geben Sie mir einen freundlichen, genau beschriebenen Hinweis auf meine Fehler. Ich freue mich immer über konstruktives Feedback und Verbesserungsvorschläge. Bitte verwenden Sie dazu das Kontaktformular: [www.dotnet-doktor.de/Leserfeedback](http://www.dotnet-doktor.de/Leserfeedback)

**Tipp:** Ich belohne Sie mit E-Books für gemeldete Fehler, siehe Kapitel "Über dieses Buch/ Ihre Belohnung, wenn Sie helfen, dieses Buch zu verbessern".

Ich helfe Ihnen gerne dabei, Ihren eigenen Programmcode zu schreiben, aber ich hoffe, Sie verstehen, dass ich dies nicht ehrenamtlich tun kann. Wenn Sie **technische Hilfe** zu Entity Framework und Entity Framework Core oder anderen Themen rund um die Entwicklung und den Betrieb von Anwendungen (Desktop, Web und Mobile) sowie Server und Cloud benötigen, stehe ich Ihnen im Rahmen meiner

beruflichen Tätigkeit für die Firma [www.IT-Visions.de](http://www.IT-Visions.de) (Beratung, Schulung, Support, Softwareentwicklung) gerne zur Verfügung. Bitte wenden Sie sich für ein Angebot an das jeweilige Kundenteam. Bitte kontaktieren Sie die Firmen aber nicht für Feedback und Verbesserungsvorschläge zu diesem Buch, da dieses Buch reine Privatsache ist.

Die Beispiele zu diesem Buch können Sie herunterladen auf der von mir ehrenamtlich betriebenen **Leser-Website** unter [www.dotnet-doktor.de/Leser](http://www.dotnet-doktor.de/Leser). Dort müssen Sie sich registrieren. Bei der Registrierung wird ein Losungswort abgefragt. Bitte geben Sie dort **Prodigy** ein (siehe auch Kapitel "Programmcodebeispiele zum Download").

Herzliche Grüße aus Essen, dem Herzen der Metropole Ruhrgebiet

Holger Schwichtenberg

## 4 Über den Autor

- Studienabschluss Diplom-Wirtschaftsinformatik an der Universität Essen
- Promotion an der Universität Essen im Fachgebiet komponentenbasierter Softwareentwicklung
- Seit 1996 in der IT tätig als Softwareentwickler, Softwarearchitekt, Berater, Dozent und Fachjournalist
- Fachlicher Leiter des Expertenteams bei [www.IT-Visions.de](http://www.IT-Visions.de) in Essen
- Über 95 Fachbücher bei verschiedenen Verlagen, u.a. Carl Hanser Verlag, O'Reilly, APress, Microsoft Press und Addison Wesley sowie im Selbstverlag
- Mehr als 1500 Beiträge in Fachzeitschriften und Online-Portalen
- Gutachter in den Wettbewerbsverfahren der EU vs. Microsoft (2006-2009)
- Ständiger Mitarbeiter der Zeitschriften iX (seit 1999), dotnetpro (seit 2000) und Windows Developer (seit 2010) sowie beim Online-Portal heise.de (seit 2008)
- Regelmäßiger Sprecher auf nationalen und internationalen Fachkonferenzen (z.B. BASTA!, Developer Week, .NET Developer Conference, MD DevDays, Microsoft TechEd, Microsoft Summit, Microsoft IT Forum, OOP, .NET Architecture Camp, IT Tage, enterJS, Advanced Developers Conference, DOTNET Cologne, iterate=>ruhr, Community in Motion, DOTNET-Konferenz, VS One, NRW.Conf, Windows Forum, Container Conf)
- Auszeichnungen und Zertifikate von Microsoft:
  - Microsoft Most Valuable Professional (MVP), ununterbrochen ausgezeichnet seit 2004
  - Microsoft Certified Solution Developer (MCSD)
- Thematische Schwerpunkte:
  - Softwarearchitektur, mehrschichtige Softwareentwicklung, Softwarekomponenten
  - Visual Studio, Continous Integration (CI) und Continous Delivery (CD) mit Azure DevOps
  - Microsoft .NET (.NET Framework, .NET Core, modernes .NET), C#, Visual Basic
  - .NET-Architektur, Auswahl von .NET-Techniken
  - Einführung von .NET, Migration auf .NET
  - Webanwendungsentwicklung und Cross-Plattform-Anwendungen mit HTML/CSS, JavaScript/TypeScript und C# sowie Webframeworks wie Angular, Vue.js, Svelte, ASP.NET (Core) und Blazor
  - Verteilte Systeme/Webservices mit .NET, insbesondere WebAPI, gRPC und WCF/CoreWCF
  - Relationale Datenbanken, XML, Datenzugriffsstrategien
  - Objekt-Relationales Mapping (ORM), insbesondere ADO.NET Entity Framework und Entity Framework Core
  - PowerShell
  - Architektur- und Code-Reviews
  - Performance-Analysen und -Optimierung
  - Entwicklungsrichtlinien



- Ehrenamtliche Community-Tätigkeiten:
  - Vortragender für die International .NET Association (INETA) und .NET Foundation
  - Betrieb diverser Community-Websites:  
[www.dotnet-lexikon.de](http://www.dotnet-lexikon.de)  
[www.dotnetframework.de](http://www.dotnetframework.de)  
[www.windows-scripting.de](http://www.windows-scripting.de)  
[www.aspnetdev.de](http://www.aspnetdev.de)  
u.a.
- Firmenwebsite: [www.IT-Visions.de](http://www.IT-Visions.de)
- Weblog: [www.dotnet-doktor.de](http://www.dotnet-doktor.de)
- Kontakt für Anfragen zu Schulung und Beratung sowie Softwareentwicklungsprojekten:  
E-Mail [kundenteam@IT-Visions.de](mailto:kundenteam@IT-Visions.de)  
Telefon [0201 / 64 95 90 – 50](tel:0201/649590-50)

Kontakt für Feedback zu diesem Buch:

[www.dotnet-doktor.de/Leserfeedback](http://www.dotnet-doktor.de/Leserfeedback)

# 5 Über dieses Buch

## 5.1 Versionsgeschichte dieses Buchs

Die Versionsgeschichte dieses Buch finden Sie in einem eigenen Kapitel am Ende des Buchs.

**Hinweis:** Die Versionsgeschichte ist eine wichtige Referenz für die Leser, die sich aktuelle Versionen des Buchs beschaffen (z.B. über Leanpub.com) und wissen wollen, was sich geändert hat. Wenn Sie das Buch erstmalig lesen, müssen Sie die Versionsgeschichte nicht lesen.

## 5.2 Hinweis zu den Vertriebswegen

Dieses Fachbuch wird vertrieben auf folgenden Wegen (Ich nenne neben dem Verkaufspreis auch, wie viel – bzw. wenig – ich als Autor von den jeweiligen Händlern erhalte. Der Rest ist Gewinn der Händler!):

- Gedruckt bei Amazon.de für 69,99 Euro (der Autor erhält 25,30 Euro):  
[www.amazon.de/exec/obidos/ASIN/3934279295/itvisions-21](http://www.amazon.de/exec/obidos/ASIN/3934279295/itvisions-21)
- Kindle-E-Book bei Amazon.de für 59,99 Euro (der Autor erhält 19,62 Euro):  
[www.amazon.de/exec/obidos/ASIN/B0CMBZW766/itvisions-21](http://www.amazon.de/exec/obidos/ASIN/B0CMBZW766/itvisions-21)
- PDF-E-Book bei Leanpub.com ab 49,99 Dollar (der Autor erhält 39,99 Dollar):  
[www.leanpub.com/EntityFrameworkCore9](http://www leanpub com/EntityFrameworkCore9)
- Als Teil des E-Book-Buch-Abos zusammen mit anderen aktuellen Fachbüchern für 99,00 Euro im Jahr **inkl. aller Updates** (der Autor erhält den kompletten Preis):  
[www.IT-Visions.de/BuchAbo](http://www.IT-Visions.de/BuchAbo)

**Hinweise:** Ich habe mich für den Vertriebsweg des gedruckten Buchs über Amazon entschieden, weil ich dort ständig Updates zu dem Buch einreichen kann. Per Print-on-Demand erhalten Leser dann immer das topaktuelle Buch. Oft liefert Amazon dennoch am Tag nach der Bestellung das Buch schon aus. Der Vertrieb dieses Buch über klassische IT-Verlage, die leider heutzutage immer noch größere Auflagen vorproduzieren, ist für ein sehr agiles Softwareprodukt wie Entity Framework Core keine Alternative mehr.

Ich nenne dabei auch den Erlös für den Autor, weil ich sehr häufig Leser treffe, die fälschlicherweise denken, der wesentliche Teil des Buchpreises komme dem Autor zu Gute. Das ist leider nicht so, außer bei Leanpub.com oder eigenen Vertriebswegen wie meinem Buchabo. Daher denke ich, dass es sinnvoll ist, dies transparent zu machen.

**Tipp:** Käufer bei Leanpub.com können jederzeit Aktualisierungen des PDF-Buchs (gleiche Hauptversion) kostenfrei dort beziehen. Käufer bei Amazon erhalten die PDF-Ausgabe einmalig kostenfrei (siehe Kapitel "Bezugsquelle des PDF-E-Books"). E-Book-Abonnenten haben jederzeit Zugriff auf alle aktuellsten Ausgaben der Fachbücher von Dr. Holger Schwichtenberg.

## 5.3 Bezugsquelle des PDF-E-Books für Amazon-Kunden

Wenn Sie dieses Buch in gedruckter Form oder als Kindle-Ausgabe bei Amazon erworben haben, können Sie zusätzlich eine PDF-Version des Buchs **kostenfrei** erhalten.

Leiten Sie dazu Ihren Kaufbeleg von Amazon an folgende E-Mail-Adresse weiter:

[PDFBuchZugabe@dotnet-doktor.de](mailto:PDFBuchZugabe@dotnet-doktor.de)

Geben Sie dabei bitte Vorname, Name, Firma und E-Mail-Adresse an.

Sie erhalten dann binnen 1-2 Wochen das auf Sie personalisierte PDF-Dokument. Dieses Angebot gilt innerhalb von 6 Monaten nach dem Kauf des Buchs bei Amazon.

## 5.4 Bezugsquelle für Aktualisierungen

Sie können jederzeit Aktualisierungen des PDF-Buchs (gleiche Hauptversion!) kostenfrei bei Leanpub.com beziehen.

Käufer der Kindle- oder Druck-Version können die aktuelle PDF-Version zum Preis von 15,00 Dollar (zzgl. 7% Mehrwertsteuer) unter folgender Webadresse beziehen:

<http://www.leanpub.com/EntityFrameworkCore9/Prodigy>

**Hinweise:** Leider erlauben Amazon und andere Buchhändler aufgrund der Buchpreisbindungsgesetze in Deutschland den Autoren grundsätzlich nicht, dass Leser eine Aktualisierung im Kindle-Format oder in gedruckter Form vergünstigt erhalten.

Bitte beachten Sie auch, dass die ISBN-Regularien erfordern, dass man bei einer Titeländerung bei neuer Produktversion eine neue ISBN vergeben und damit auch ein neues Buchprojekt bei Amazon und Leanpub erstellt werden muss.

## 5.5 Hinweise zur Breite und Tiefe dieses Buchs – Sie haben Einfluss!

Ein Fachbuch, das ein riesengroßes Themengebiet wie Entity Framework Core behandelt, kann nicht jedes Teilgebiet und jeden Aspekt der Programmiersprache behandeln, zumindest nicht in gleicher Tiefe. Dann würde solch ein Fachbuch über eintausend Seiten, in einigen Fällen sogar mehrere tausend Seiten umfassen.

Ich denke, dass ich nach aktuellem Stand der Technik und meinem Wissenstand etwa 1.500 Seiten zu Entity Framework Core schreiben könnte. Würden Sie so ein dickes (und entsprechend teures) Buch kaufen und lesen wollen?

Wie jeder Fachautor lese auch ich immer wieder Kritik, dass ein Leser ein bestimmtes Thema nicht oder nicht in ausreichender Tiefe behandelt sei in dem Buch. Das ist aus der Sicht des einzelnen Lesers sicherlich gerechtfertigt, aber wie jeder Fachautor muss ich eben zwingend eine Auswahl der Themen treffen. Gerne dokumentiere ich hier, wie ich persönlich diese Auswahl für meine Bücher treffe:

- Ich behandle im Buch die Themen, die wir in unserer Firma selbst in der Praxis brauchen.
- Ich behandle zusätzlich die Themen, die unsere Kunden in Beratungsgesprächen behandelt haben möchten.
- Ich behandle Themen, die ich spannend finde.

Folglich sind die Themen, die ich im Buch nicht oder nur kurz behandle für uns und unsere Kunden nicht relevant bzw. so selbsterklärend, dass es keine Fragen dazu gibt.

Natürlich kann das für Sie anders sein. Sie können mir immer gerne schreiben, wenn Sie ein Thema im Buch behandelt haben möchten. Ich sammele diese Anregungen und wenn es mehrere Zuschriften zu einem Thema gibt, dann kommt das Thema weit oben auf die Prioritätenliste. Ich denke, das ist ein faires Verfahren.

## 5.6 Geplante Kapitel

Die Reihenfolge der für die folgenden Versionen geplanten Kapitel ist hier zunächst alphabetisch angeordnet und entspricht nicht der Reihenfolge, in der die Kapitel erscheinen werden.

- AddDbContextFactory() und AddPooledDbContextFactory() (seit Entity Framework Core 5.0)
- Anpassung der Code-Generierung beim Reverse Engineering mit Handlebars (Paket EntityFrameworkCore.Scaffolding.Handlebars)
- Auditing mit Entity Framework Plus
- Azure Cosmos DB (seit Entity Framework Core 3.0)

- Bulk Insert mit Entity Framework Extensions [<https://entityframework-extensions.net/>]
- Change Tracking mit Proxies (UseChangeTrackingProxies())
- Command Interceptors (seit Entity Framework Core 3.0)
- Connection Resiliency / EnableRetryOnFailure() (seit Entity Framework Core 1.1)
- Diagnostik mit Diagnostic Source / Diagnostic Listener (seit Entity Framework Core 3.0)
- Einfügen expliziten Primärschlüsselwerten trotz aktivierten Autowerten
- Event Counter
- Filtered Include für abgeleitete Typen (seit Entity Framework Core 2.1)
- ExcludeFromMigrations()
- Geo-Datentypen mit NetTopologySuite (seit Entity Framework Core 2.2)
- Group Join
- IDesignTimeDbContextFactory (seit Entity Framework Core 5.0)
- IModelCacheKeyFactory  
[<https://docs.microsoft.com/de-de/ef/core/modeling/dynamic-model>]
- MonoDB (Treiber verfügbar seit Mai 2024)
- Parameter in Konstruktoren von Entitätsklassen (seit Entity Framework Core 2.1)
- Performance: ToArray() vs.ToList() vs. ToHashTable() vs. ToDictionary()
- Performance: Asynchrone Methoden vs. Synchrone Methoden
- Pre-Convention Model Configuration (seit Entity Framework Core 6.0)
- Projektstrukturen anlegen mit der .NET Core CLI
- Record-Typen (record class und record struct) mit Entity Framework Core
- Reverse Engineering von Oracle-Datenbanken mit Entity Framework Core Power Tools (seit August 2020 möglich)
- Savepoints in Transaktionen (seit Entity Framework Core 5.0)
- SaveChangesInterceptor (seit Entity Framework Core 5.0)
- Timestamps mit SQLite
- SQL Server Memory-Optimized Tables (seit Entity Framework Core 1.1)
- SQLite in Blazor WebAssembly-Anwendungen nutzen
- Trackable Entities für EF Core [<https://github.com/TrackableEntities/TrackableEntities.Core>]
- Transaction-IDs (seit Entity Framework Core 5.0)
- User Defined Functions (UDF)
- Value Conversion mit komplexen Typen (Mapping mehrerer Properties auf eine Spalte)
- Visual Studio Analyzer für SQL-Abfragen (seit Entity Framework Core 2.1)
- Zusätzliche Erweiterungen aus der Community wie z.B. EntityFrameworkCore.Rx, EFDetached.EntityFramework, EntityFrameworkCore.Triggers, EntityFrameworkCore.PrimaryKey und EntityFrameworkCore.TypedOriginalValues
- Zustandsänderungereignisse im Change Tracker (seit Entity Framework Core 2.1)

**Hinweis:** Mit dieser Liste kommender Themen möchte ich gleichzeitig klarstellen, welche Themen Sie derzeit nicht im Buch finden, damit Sie als Leser keine falschen Erwartungen haben.

## 5.7 Programmiersprache in diesem Buch

Als Programmiersprache kommt in diesem Buch C# zum Einsatz, weil dies die bei weitem häufigste verwendete Programmiersprache in .NET ist. Der Autor dieses Buchs programmiert in einigen Kundenprojekten .NET-Anwendungen zwar auch in Visual Basic .NET, leider bietet dieses Buch jedoch nicht den Raum, alle Listings in beiden Sprachen wiederzugeben.

Eine Sprachkonvertierung zwischen C# und Visual Basic .NET ist im WWW kostenfrei verfügbar z.B. auf der Website <http://converter.telerik.com>

Falls Sie auf C# umsteigen möchten oder noch eine ältere Version von C# verwenden, möchte ich Ihnen mein Fachbuch "C# Crashkurs" empfehlen: [www.IT-Visions.de/CSharpBuch](http://www.IT-Visions.de/CSharpBuch)



## 5.8 Notwendige Vorkenntnisse

Datenbankprogrammierung mit OR-Mappern ist ein großes, komplexes Gebiet. In diesem Buch konzentriere ich mich auf Entity Framework Core im engeren Sinne.

Als Vorkenntnisse sollten Sie zumindest ein gutes Grundlagenwissen in folgenden Gebieten mitbringen:

- Relationale Datenbanken und Datenbankdesign
- Structured Query Language (SQL)
- .NET
- C#
- Visual Studio

## 5.9 Hinweis auf einen Bug im PDF-Exporter von Word

Nutzer der PDF-Version dieses Buch werden feststellen, dass die Hyperlinks an einigen Stellen im Buch nicht zu funktionieren scheinen. Ich möchte Ihnen aber versichern, dass

- ich die Hyperlinks per Copy&Paste aus dem Browser ohne Verlust in mein Manuskript in Microsoft Word übernommen habe und
- die Links innerhalb des Word-Dokuments funktionieren.

Allerdings hat der PDF-Exporter von Word einen seit langem bekannten und von Microsoft immer noch nicht behobenen Bug: Der PDF-Exporter nimmt manchmal einen Buchstaben zu viel in den Link, z.B. aus diesem Hyperlink in Word

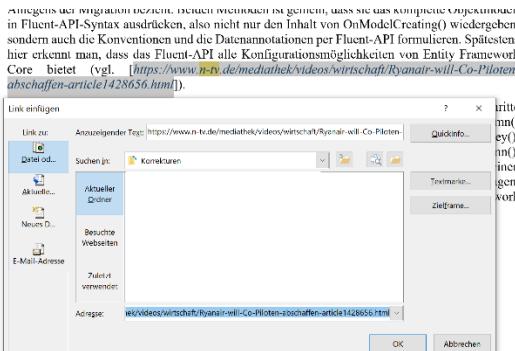


Abbildung: Korrekter Link im Word-Dokument

wird dann im PDF ein Link, bei dem die schließende Klammer fehlerhafter Weise mit aufgenommen wird.

sondern auch die Konventionen und die Datenannotationen per Fluent-API formulieren. Spätestens hier erkennt man, dass das Fluent-API alle Konfigurationsmöglichkeiten von Entity Framework Core bietet (vgl. [<http://www.n-tv.de/mediathek/videos/wirtschaft/Ryanair-will-Co-Piloten-abschaffen-article1428656.html>]).

*Abbildung: Fehlerhafter Link mit einem Buchstaben zu viel im PDF-Dokument, das Word erzeugt*

Ich bedauere das Problem, bitte Sie allerdings, sich in dieser Sache an Microsoft zu wenden, da ich das nicht ändern kann. Ich kann nur einen Tipp zur Problemumgehung geben: Hyperlink per Copy & Paste rauskopieren in den Browser.

## 5.10 Ihre Belohnung, wenn Sie helfen, dieses Buch zu verbessern!

Wenn Sie Fehler in diesem Buch finden, bin ich Ihnen nicht nur wirklich sehr dankbar, sondern Sie bekommen auch eine Belohnung in Form von aktualisierten oder weiteren E-Books.

Fehlerart	E-Book-Guthaben
Inhaltlicher Fehler	Pro Fehler 5 Euro
Sprachlicher Fehler	Pro Fehler 2 Euro

Ein Beispiel: Wenn Sie zwei inhaltliche Fehler und zehn Rechtschreibfehler in diesem Buch finden, dann haben Sie bei mir 30 Euro gut. Dafür können Sie dann eins meiner selbstverlegten Bücher als E-Book bekommen.

Die selbstverlegten Bücher finden Sie unter [www.IT-Visions.de/Verlag](http://www.IT-Visions.de/Verlag)

Melden Sie die Fehler unter [www.dotnet-doktor.de/Leserfeedback](http://www.dotnet-doktor.de/Leserfeedback)

Schreiben Sie dabei, welches E-Book Sie wünschen. Das Buch schicke ich Ihnen dann per E-Mail zu.

**Tipp:** Auch Fehler auf meiner persönlichen Website [www.dotnet-doktor.de](http://www.dotnet-doktor.de) und der Firmenwebsite [www.IT-Visions.de](http://www.IT-Visions.de) zählen mit!

Ich freue mich auf Ihre Fehlermeldung!

Holger Schwichtenberg

P.S. Die Fehlermeldung zählt nur, wenn nicht ein anderer Leser dies bereits gemeldet hat und es daher in der aktuellen Auflage schon korrigiert ist.

## 6 Fallbeispiele in diesem Buch

Die meisten Beispielprogrammcodes in diesem Buch drehen sich um das Fallbeispiel der fiktiven Fluggesellschaft "WorldWideWings", abgekürzt "WWWings" oder als dreibuchstabiger Airline Code einfach "WWW". Es gibt auch eine Website zu der Fluggesellschaft ([www.world-wide-wings.de](http://www.world-wide-wings.de)) – dort einen Flug zu buchen, möchte der Autor dieses Buchs Ihnen aber nicht empfehlen ☺



Abbildung: Logo der fiktiven Fluggesellschaft "WorldWideWings"

**Hinweis:** In einzelnen Unterkapitel werden andere Fallbeispiele verwendet (z.B. die Aufgabenverwaltung "MiracleList"). Diese Fallbeispiele werden dann in den jeweiligen Kapiteln erläutert.

### 6.1 Entitäten

Im Anwendungsfall "WorldWideWings" geht es um folgende Entitäten:

- **Flüge** zwischen zwei Orten, bei denen die Orte bewusst nicht als eigene Entität modelliert wurden, sondern Zeichenketten sind (dies vereinfacht das Verständnis vieler Beispiele).
- **Passagiere**, die auf Flügen fliegen.
- **Mitarbeiter** der Fluggesellschaft, die wiederum Vorgesetzte haben, die auch Mitarbeiter sind.
- **Piloten** als eine Spezialisierung von Mitarbeitern. Ein Flug hat im einfacheren Modell nur einen Piloten. Es gibt keinen Copiloten bei WorldWideWings. Den Copiloten abzuschaffen und im Notfall das Flugzeug von der Stewardess landen zu lassen (wie im Film "Turbulence" von 1997) war übrigens im Jahr 2010 ein echter Vorschlag von Michael O'Leary, dem Chef der irischen Fluggesellschaft Ryanair (siehe [vgl. <http://www.n-tv.de/mediathek/videos/wirtschaft/Ryanair-will-Co-Piloten-abschaffen-article1428656.html>] ] und [<http://www.dailymail.co.uk/news/article-1308852/Let-stewardesses-land-plane-crisis-says-Ryanair-boss-Airline-wants-ditch-pilots.html>]).
- **Personen** als Sammlung der gemeinsamen Eigenschaften für alle Menschen in diesem Beispiel. Personen gibt es aber nicht eigenständig, sondern nur in den Ausprägungen/Spezialisierungen Passagier, Mitarbeiter und Pilot. Im objektorientierten Sinne ist Person also eine abstrakte Basisklasse, die keine Instanzen besitzen kann, sondern nur der Vererbung dient.

Es gibt zwei Datenmodelle:

- Das etwas einfachere Modell #1 (alias Modell Version 1, siehe Abbildungen 1 und 2) ist das Ergebnis klassischen relationalen Datenbankdesigns mit Normalisierung. Das Objektmodell daraus entsteht per Reverse Engineering.
- Modell #2 (alias Modell Version 2, siehe Abbildungen 3 und 4) ist das Ergebnis des Forward Engineering mit Entity Framework Core aus einem Objektmodell. Zusätzlich gibt es hier weitere Entitäten (Personendetail, Flugzeugtyp und Flugzeugtypdetail), um weitere Modellierungsaspekte aufzuzeigen zu können. In diesem Fall gibt es auch für jeden Flug einen optionalen Copiloten.

In Modell #1 gibt es eine jeweils eigene Tabelle für Personen (auch wenn es keine eigenständigen Personen gibt), Mitarbeiter, Piloten und Passagiere. Diese Aufteilung entspricht den Klassen im Objektmodell.

**Hinweise:** Das Objektmodell zum Datenbankschema WorldWideWings Version 1 (Abbildung 2) ist das automatisch von Entity Framework Core aus der Datenbank generierte Objektmodell (Reverse Engineering); es ist bewusst nicht verändert worden, auch wenn einige der generierten Namen unschön sind.

Die Möglichkeit der N:M-Abstraktion, die es in Entity Framework Core seit Version 5.0 gibt, ist in dem Objektmodell noch nicht umgesetzt, es beim Reverse Engineering in Version 5.0 noch keine N:M-Abstraktion gibt; dies soll erst in Version 6.0 kommen.

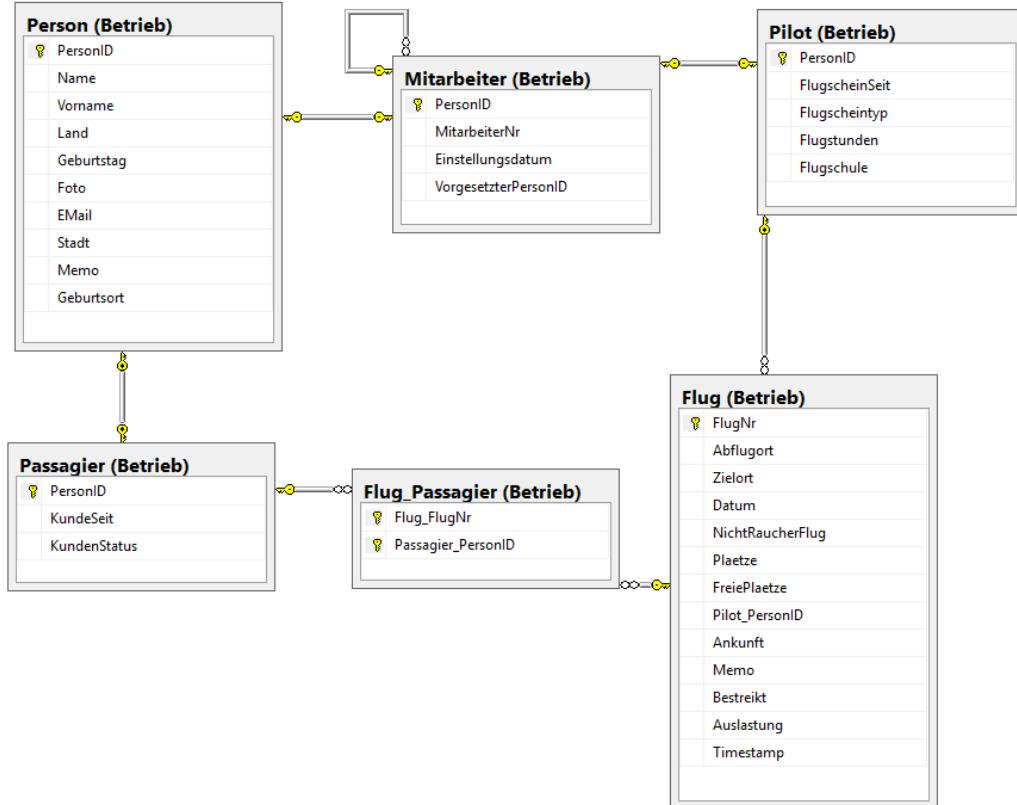


Abbildung 1: WorldWideWings-Datenmodell in der einfacheren Version 1

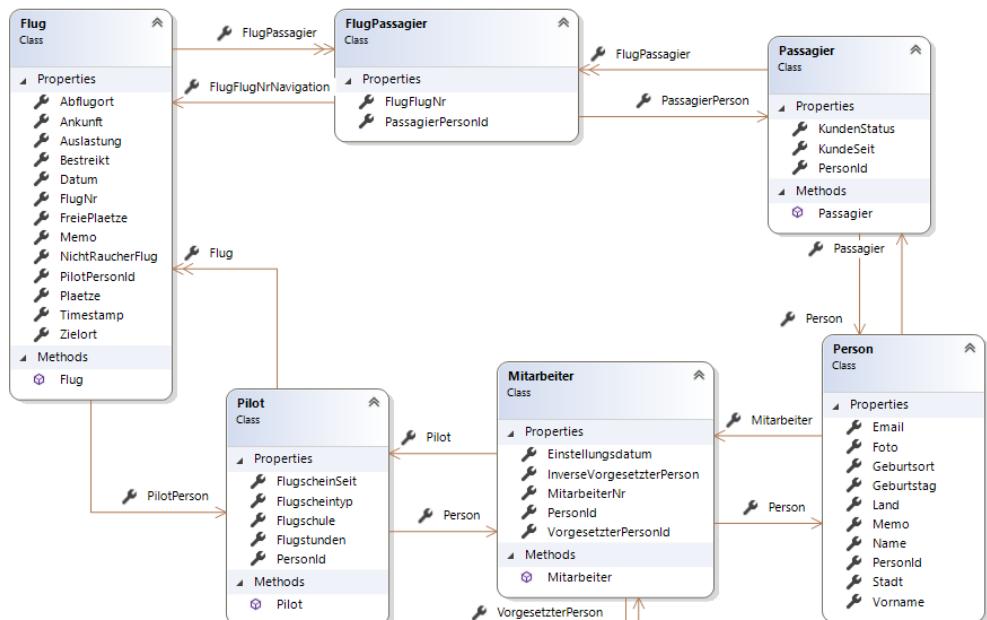


Abbildung 2: Objektmodell zum WorldWideWings-Datenmodell in der einfacheren Version 1

In Modell #2 gibt es lediglich die Tabellen Passagiere und Mitarbeiter für diese vier Entitäten. Entity Framework Core ist derzeit etwas eingeschränkt und unterstützt das "Table-per-Type"-Mapping (also eine eigenständige Tabelle für jede Klasse) nicht. Daher umfasst die Tabelle Passagiere auch alle Eigenschaften von Person. Die Tabelle Mitarbeiter umfasst neben den Personeneigenschaften die Eigenschaften der Entitäten Mitarbeiter und Pilot. In der Tabelle wird per Diskriminatorspalte unterschieden zwischen Datensätzen, die ein Mitarbeiter sind, und solchen, die ein Pilot sind. Entity Framework Core mischt hier die Konzepte Table-per-Concrete-Type (TPC) und Table-per-Hierarchy (TPH). Einen dezidierten Einfluss auf diese Abbildung hat man in Entity Framework Core 1.x/2.0 noch nicht. Das klassische Entity Framework bietet hier mehr Optionen.

Die Abhängigkeitsarten in Modell #2 sind:

- Ein **Flug** muss einen Piloten besitzen. Es gibt einen Copiloten, aber er ist optional.
- Ein Flug kann optional einen **Flugzeugtyp** zugeordnet haben. Ein Flugzeugtyp hat eine Beziehung zu **Flugzeugtypdetail**.
- Jede Person und damit auch jeder Pilot und Passagier muss ein **Personendetail**-Objekt besitzen.

In diesem Buch kommen beide Datenmodelle vor, teilweise auch in modifizierter Form, um bestimmte Szenarien (z.B. Datenbankschemamigrationen) aufzuzeigen.