



# SQL Server Data Tools

Speaker: Olaf Lischke

# Speaker: Olaf Lischke



macht .NET, seit es .NET gibt



versucht, Projekte und Seminare zu kombinieren



singt Tenor in zwei Chören



zockte schon auf dem ZX 81, heute ausschließlich PC



fotografiert, seit er eine Kamera halten kann



fliegt, wenn Wetter und Zeit es zulassen

# Überblick

- Offline-Entwicklung
- Entwicklung via Datenbankverbindung
- Datenbank-Wartung
- Deployment

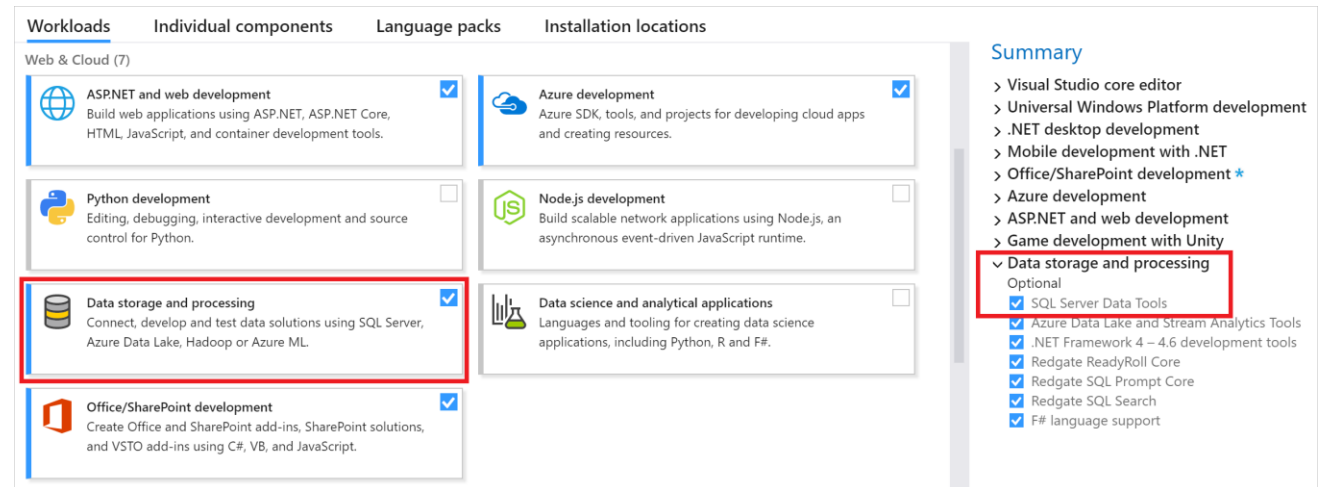
## Microsoft Doku:

englisch - <https://docs.microsoft.com/en-us/sql/ssdt/sql-server-data-tools>

deutsch - <https://docs.microsoft.com/de-de/sql/ssdt/sql-server-data-tools>

# Neu? Nicht wirklich...

- Stand: Version 17.8, Release 15.8 (05.09.2018)
- Eigenes Paket im VS 2017 Installer



- ... aber auch standalone verfügbar





# Offline-Entwicklung

(MSDN: „Project-Oriented Offline Database Development“)

# Offline-Entwicklung

- Ohne direkte Datenbank-Anbindung
- Prima für:
  - Entwurf von Datenbanken
  - Größere Schema-Änderungen/-Erweiterungen
- Visual Studio-Unterstützung
- Funktionale Elemente testbar (mit LocalDb)

# Offline – Project Oriented Development

 T-SQL	 Eigenständig	 Integriert	 Schemaorientiert
Go To Definition	Source Code basiert	Source Code Control	Snapshots
Find All References	F5 Debugging & Testing mit LocalDB	MSBuild	Schemavergleich
Refactoring		Kommandozeilenwerkzeuge	T-SQL Code Analysis

# Entwicklung via Datenbankverbindung





(MSDN: „Connected Database Development“)



# Entwicklung via DB-Verbindung

- Direkte Verbindung zu einer Datenbank
- Prima für:
  - Wartung und Pflege
  - Daten-/Schemavergleich
  - Erweiterung bestehender Datenbanken
- Elegant durch Cloning und Publishing
- Visual Studio-Unterstützung
- SQL Server ab 2005, Azure

# Entwicklung via DB-Verbindung

 Bekannte Umgebung	 Schema Tools	 Visual Studio Unterstützung	 Vergleich
SQL Server Object Explorer	Tabellendesigner	T-SQL IntelliSense	Datenvergleich
T-SQL Editor	Codeansicht	T-SQL Debugging	Schemavergleich
	Fehlerliste	Daten bearbeiten	

# Datenbank-Wartung

# Vergleich

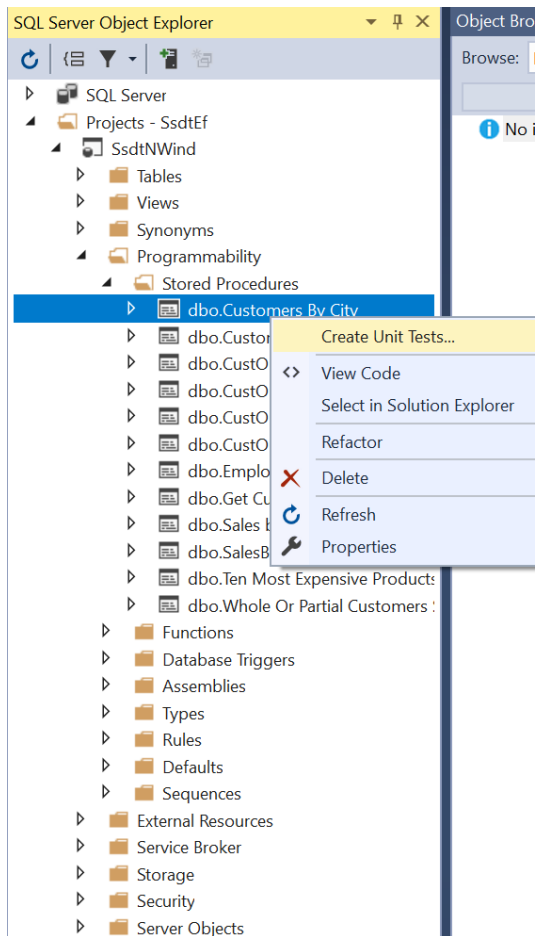
## Schema

- Projekt-Schema vs. Ziel-Datenbank
- Compare-Tool
  - direktes Update
  - Skriptgenerierung
- ...auch per Code -> DACFx (DAC-Framework)  
Microsoft.SqlServer.Dac.\*.dll in  
[ProgramFiles]\SqlServer\140\DAC  
oder SQLPackage.exe

## Daten

- Daten zweier Datenbanken:
  - Deltas
  - Nur in Quelle
  - Nur in Ziel
  - Identische
- Compare-Tool
  - direktes Update
  - Skriptgenerierung

# Unit Testing



- Testbar:
  - Stored Procedures
  - Funktionen
- Testprojekt C#/VB mit:
  - Designer/Code je Unit
  - .resx mit T-SQL
  - SQLDatabaseSetup.cs/.vb

# Deployment

# Schema Deployment



Deklarativ

Modellbasiert

Inkrementelles  
Deployment  
(Schema)



Multi-Targeting

SQL Server ab 2005  
Azure

Retargeting Support



Standards

DACPAC

Connected

SQL Skript



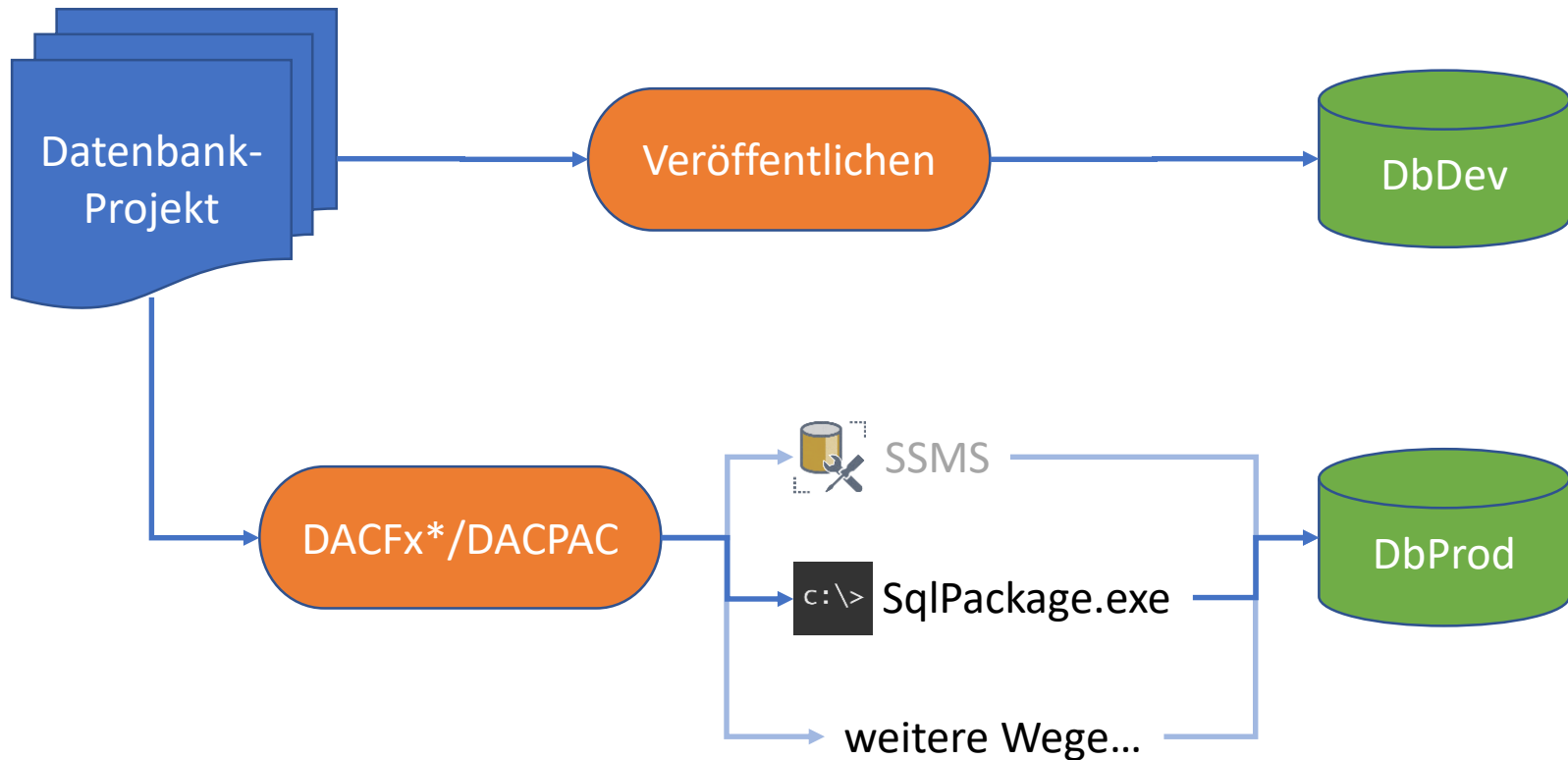
DACFramework

Format

Engine

API & REDIST

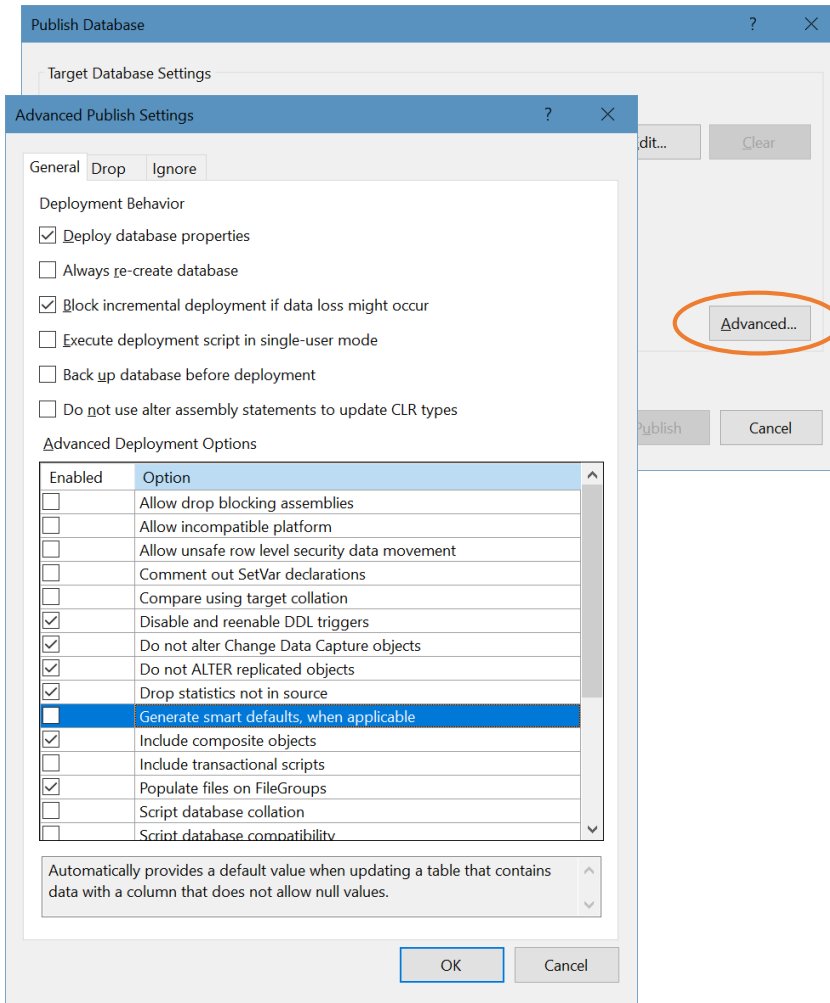
# Deployment-Pfade



\* Eigenständiger Download: [DACFramework](#) – auch für Linux und macOS verfügbar (preview)!  
Maschinen mit SQL Server: [ProgramFiles]\Microsoft SQL Server\140\DAC\bin



# Veröffentlichen



- Veröffentlichung aus Visual Studio heraus
- Konfiguration lässt kaum Wünsche offen
  - Verschiedene Profile
  - Generierung von Daten für NOT NULL-Felder u.ä. (Produktiv-Werte später per PostDeployment-Skript nachtragen!)
  - Auto-Backup vorher, optional
  - ...

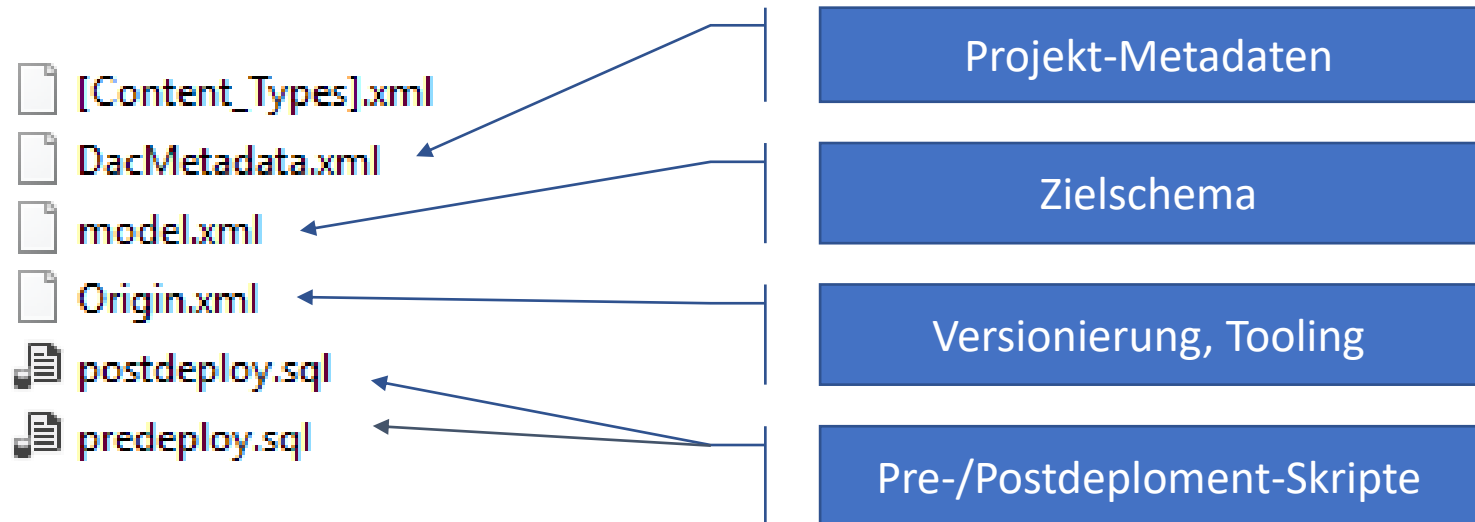
# SqlPackage.exe - Operations

- **Extract:** Erstellt eine Momentaufnahme (DACPAC-Datei) von einer Datenbank.
- **Publish:** Aktualisiert ein Datenbankschema inkrementell, sodass dieses dem Schema einer DACPAC-Quelldatei entspricht.
- **Export:** Exportiert eine Livedatenbank – einschließlich des Datenbankschemas und der Benutzerdaten – aus SQL Server oder Azure in ein BACPAC-Paket (BACPAC-Datei).
- **Import:** Importiert das Schema und die Tabellendaten aus einem BACPAC-Paket in eine neue Benutzerdatenbank einer Instanz von SQL Server oder Azure.

# SqlPackage.exe – Reports/Script

- **DeployReport:** Erstellt einen XML-Bericht der Änderungen, die durch eine Veröffentlichung vorgenommen würden.
- **DriftReport:** Erstellt einen XML-Bericht der Änderungen, die seit der letzten Registrierung an einer registrierten Datenbank vorgenommen wurden.
- **Script:** Erstellt ein inkrementelles Transact-SQL-Updateskript, durch das das Schema eines Ziels aktualisiert wird, sodass es dem Schema einer Quelle entspricht.

# Inside DACPAC



# Fazit

# Fazit

## Pro

- Integration in Visual Studio (Intellisense, Syntaxkorrektur, Debugging, Projektstruktur...)
- Schema-/Datenvergleich
- Build-Prozess-Integration
- Inkrementelles Deployment
- Unit Tests für Prozeduren und Funktionen
- Quellcodeverwaltung (TFS/Git/Ankh...)
- DACFramework

## Contra

- Keine grafischen Designer
- Nur jew. **ein** Pre-/Post-Deployment Skript
- Andere Datenbanksysteme nur mit großen Einschränkungen (OLE DB) – **aber:**
  - Oracle Developer Tools for Visual Studio
- Keine direkte Verbindung Entity Framework zu Database Project

# Vielen Dank!

Für spätere Fragen:  
[olaf.lischke@lischke-edv.de](mailto:olaf.lischke@lischke-edv.de)