



Wie eine Datenbank deployen?



Thorsten Kansy (tkansy@dotnetconsulting.eu)

Meine Person- Thorsten Kansy

Freier Consultant, Software Architekt,
Entwickler, Trainer & Fachautor



Azure Cosmos DB

Mein Service- Ihr Benefit

- Individuelle Inhouse Trainings
- (Online on-demand) Projektbegleitung
- Beratung
 - Problemanalyse und Lösungen
 - Technologieentscheidungen



Agenda

- Datenbank-Projekte
 - Publishing
- Schema & Data Compare
- Data-tier Applications
 - DACPACs
 - BACPACs
- Deployment mit .NET (Core)
- ~~Migration durch EF (Core)~~

Alle Änderungen von Hand protokollieren

- Excel, Word, ...
- Order mit T-SQL Skripten



Datenbank-Backup

- (Ziel-) Struktur mit Daten
 - Datenbank wird neu erstellt/ überschrieben
 - Restore nur auf gleicher/ höherer Version möglich



Ebenfalls kein Thema



Verwendete Software



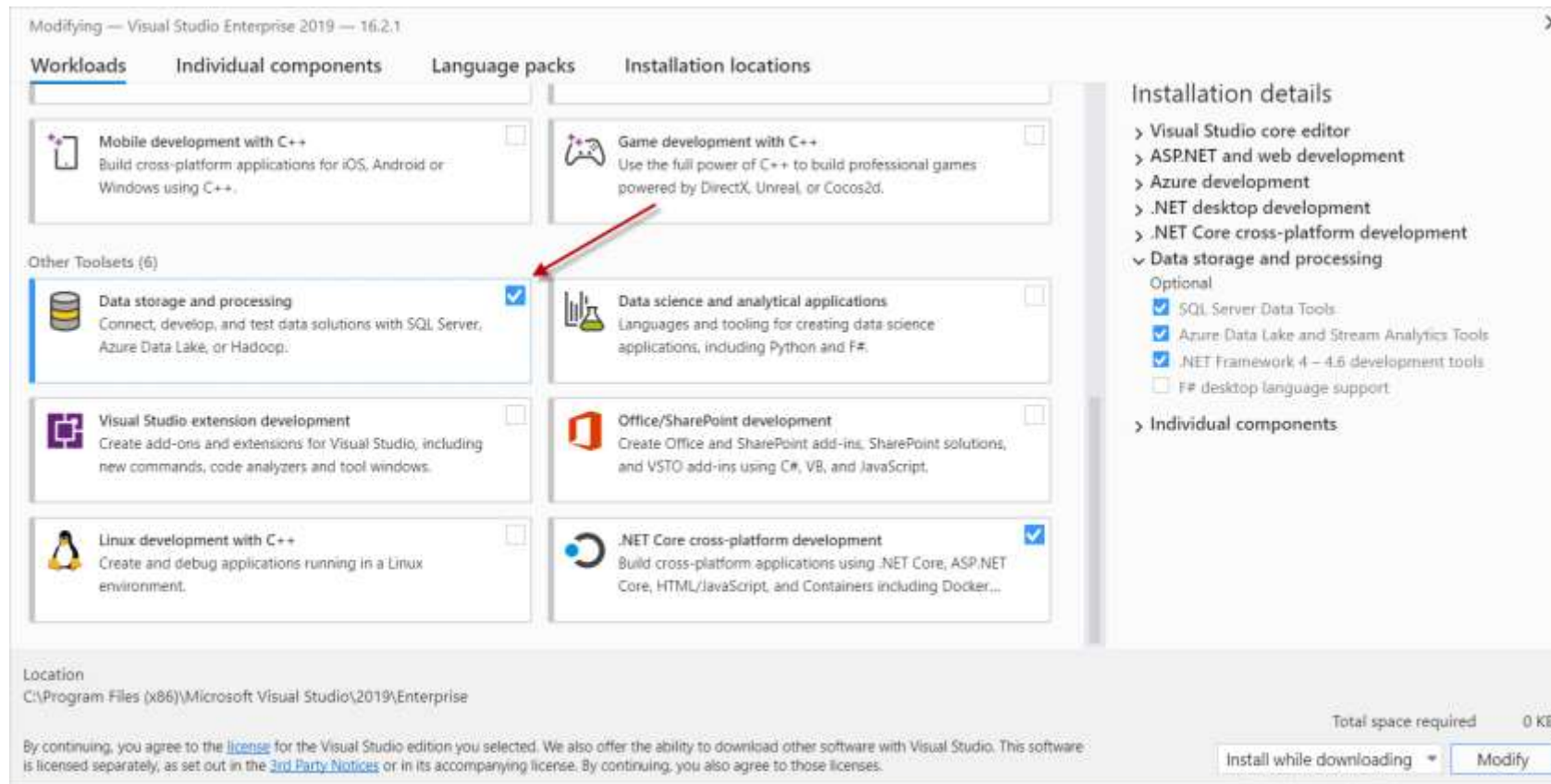
Aktuelle Versionen

Verwendete Software

- Visual Studio $\geq$ 2017
 - Alle Editions
 - SQL Server Data Tools (SSDT)
- SQL Server
 - On-Premises, Virtueller Maschine oder Docker Image ($\geq$ 2012)
 - SQL Azure
- SQL Server Management Studio 17.x/ 18.x

SQL Server Data Tools (SSDT) installieren

Installation via Visual Studio Installer (“Data storage and processing”)





Datenbank-Projekte

Datenbank-Projekte

- „Quellcode“ einer Datenbank
 - Nur Schema, kein Inhalt
 - Design-Werkzeug
 - Datenbank-Projekte als Teil einer Solution
- CLR-Integration
 - Prozeduren & Funktionen mit .NET (C# & VB.NET)

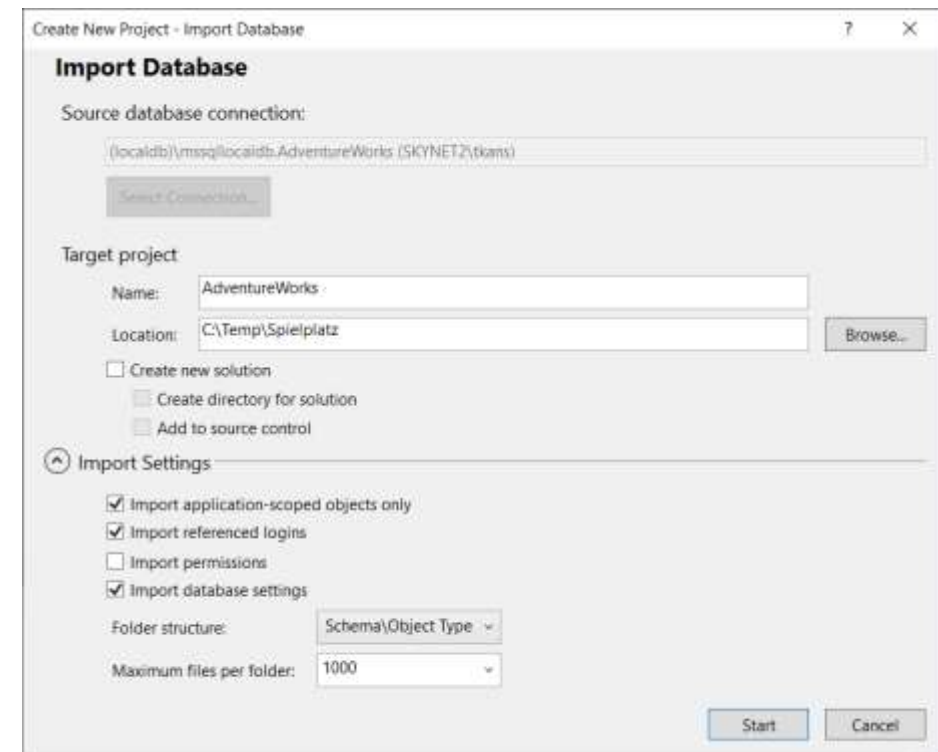
In Praxis-Projekte sollte das Datenbank-Projekt **oder** die Datenbank führend bei Änderungen sein

Datenbank-Projekte

- Erzeugen aus...
 - SQL Server Datenbank
 - SQL Server Datenbank (SQL Server Object Explorer)
 - DACPAC (Data tier-Application)
 - SQL-Skript
- Alle Objekte sind „Create“-.sql-Dateien
 - VS „compiliert“ jedes Objekt => Fehler und Warnungen
 - ideal für Git, SVN & Co
 - *.jfm-Dateien ignorieren!
- Objekte können refactored werden

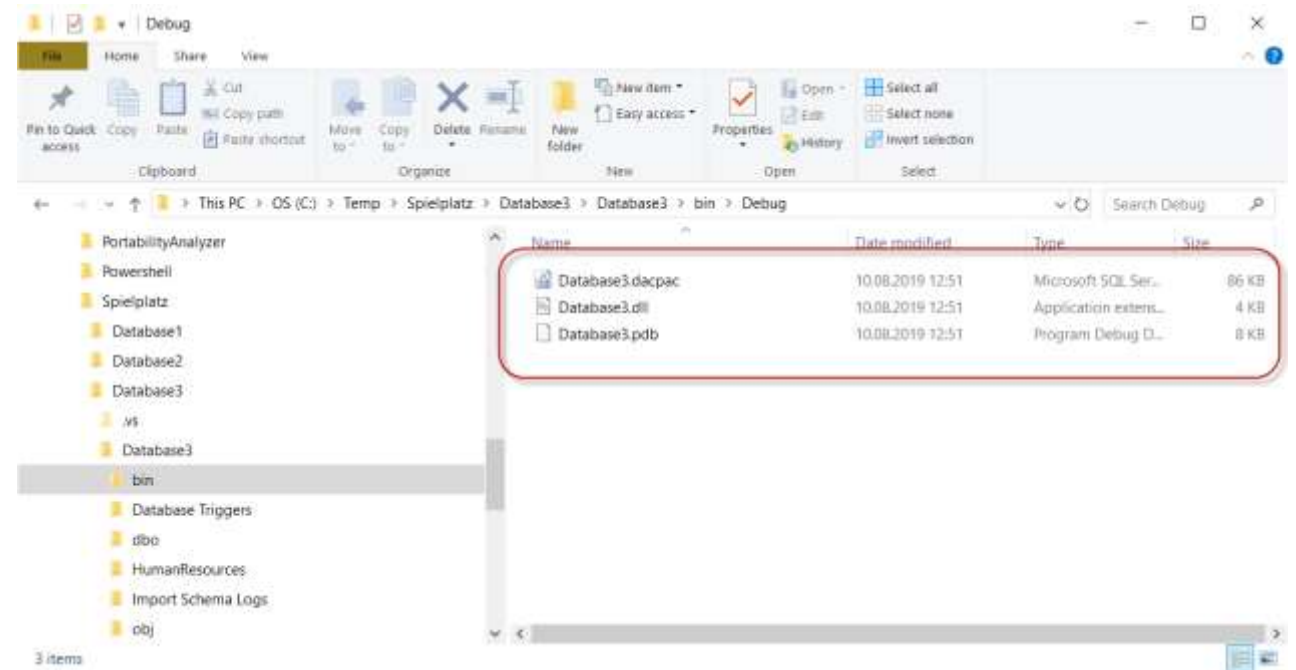
Datenbank-Projekt anlegen

- Importieren aus
 - Bestehende Datenbank
 - Data-Tier Application (DACPAC)
 - Skript



Unter der Haube

- Kompilieren
 - Data-tier Application (DACPAC)
 - Optional: SQL Skript
 - DDL & PDB für CLR-Integration



Demo



Publishing



Publishing

- Compilierung
- Publishing
 - via Connection
 - als DACPAC
 - als SQL-Skript
- Viele Detail-Einstellungen
 - Publishing Profile

Demo



Schema & Data Compare

Schema Comapre

- Vergleich der Struktur zweier Datenbank
 - Datenbank
 - Datenbank-Projekt
 - Data-tier Application (DACPAC)
- Ziel anpassen
- Erzeugung von Delta-Skripten
- *.scmp-Dateien

Demo

Data Compare

- Inhaltsvergleich zwischen zwei Datenbanken
- Primary Key muss vorhanden sein
- Ziel anpassen
- Erzeugung von Delta-Skripten
- *.dcmp-Dateien

Demo



Data-tier Applications

Data-tier Applications

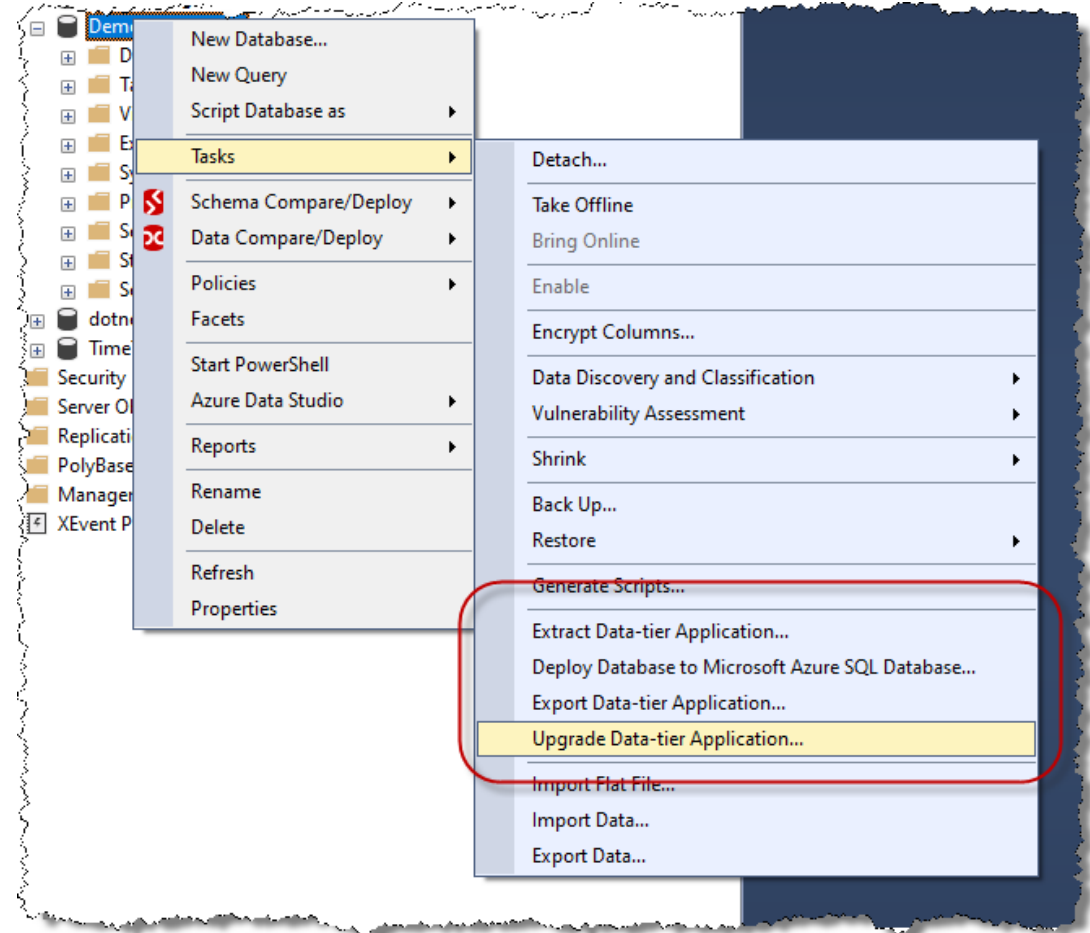
- DACPAC-Dateien
 - Nur Struktur
- BACPAC-Dateien
 - Struktur mit Daten
 - Export nach SQL Azure möglich
 - Export nach niedrigeren SQL Server Versionen möglich

DACPAC-Dateien

- (Ziel-) Struktur **ohne** Daten
 - Führt nur Anpassungen durch wo nötig
- Ein kompiliertes Datenbank-Projekt
 - Snapshot aus Database-Projekt
- Deploy/ Upgrade & Extract via
 - SQL Server Management Studio (SSMS)
 - .NET (Core)
 - Powershell
 - ...

Deployment mittels SSMS

- Deploy: DACPAC einspielen
- Upgrade: DACPAC aktualisieren
- Extract: DACPAC erstellen



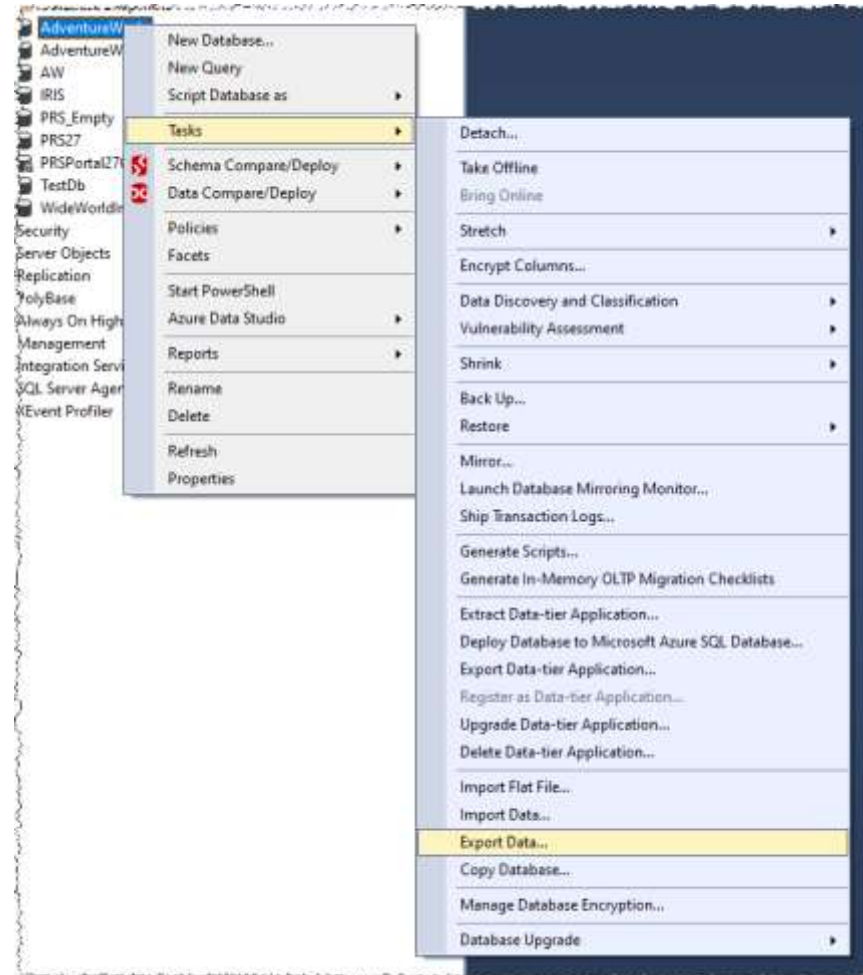
Demo

BACPAC-Dateien

- (Ziel-) Struktur **mit** Daten
 - Arbeitet nur mit leeren Datenbank
- Import & Export
 - SQL Server Management Studio (SSMS)
 - .NET (Core)
 - Powershell
 - ...

Deployment mittels SSMS

- Import: BACPAC einspielen
- Export: BACPAC erstellen



Demo

DACPAC via .NET

```
private static void deployDACPAC()
{
    // DacService Instanz erzeugen und mit Zielserver verbinden
    var dacService = new DacServices("server = " + serverName);

    // DACPAC laden
    var dacpac = DacPackage.Load(dacpacPath);

    // Update, wenn die Datenbank bereits existiert.
    bool updateDatabase = existsDatabase();

    // Deployment ausführen
    dacService.Deploy(dacpac, targetDbName, updateDatabase);
}
```

Install-Package Microsoft.SqlServer.SqlManagementObjects

BACPAC via .NET

```
private static void importBACPAC()
{
    // DacService Instanz erzeugen und mit Zielserver verbinden
    var dacService = new DacServices("server = " + serverName);

    // BACPAC laden
    var bacpac = BacPackage.Load(bacpacPath);

    // Import ausführen
    dacService.ImportBacpac(bacpac, targetDbName);
}
```

```
\Program Files (x86)\Microsoft SQL Server Management Studio
18\Common7\IDE\Extensions\Application\Microsoft.SqlServer.Dac.dll
```

Demo

Fragen? Jetzt oder später!



www.dotnetconsulting.eu

Links



www.dotnetconsulting.eu



[@Tkansy](https://twitter.com/Tkansy)



tkansy@dotnetconsulting.eu