Using DACPAC for Database Change Management

Rolf Tesmer
Data Solution Architect (DSA)
Melbourne, Australia

Microsoft

Agenda

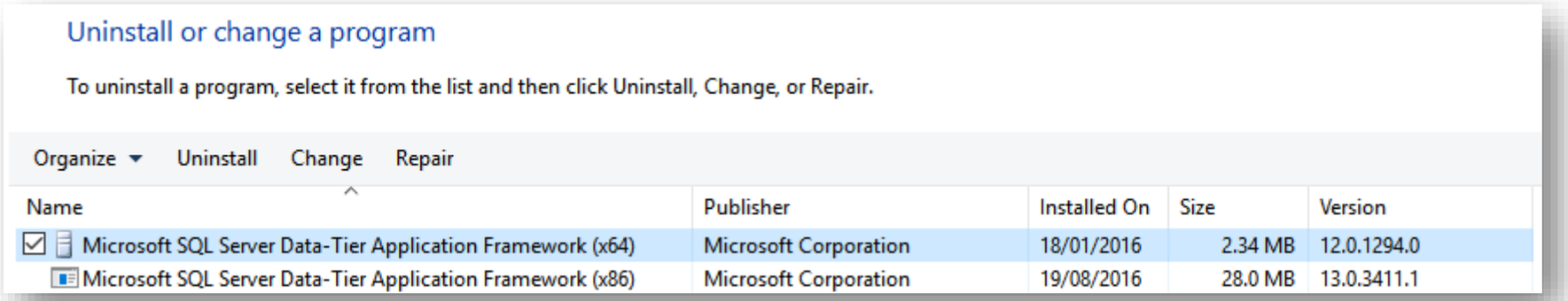
- What is a DACPAC
- Creating a DACPAC
- Deploying using DACPAC
- Advanced Methods with a DACPAC
- Troubleshooting a DACPAC
- Q&A

What is a DACPAC (and a BACPAC)

- **3 key definitions**
  1. **DAC** stands for *"Data-Tier Application"*
  2. *"A DAC is a self-contained unit of SQL Server database deployment that enables data-tier developers and database administrators to package SQL Server objects into a portable artifact called a DAC package, also known as a DACPAC"*
  3. *"A BACPAC is a related artifact that encapsulates the database schema as well as the data stored in the database."*
- A **binary file** with a **.dacpac** extension or **.bacpac** extension
- Its essentially a **zip file** containing a set of **database definition and data files**
- Every **DAC** contains a **model.xml** (*the entire database definition*) + other components
- Can only target a **single database**, DACPAC **cannot be cross-database**

Why use a DACPAC

- Primary use is to **deploy database objects to a SQL Server instance**
- Great method of **atomic control** for a **collection of database changes**
- To use them you need…
  - The "Data-tier Application Framework" (aka **DAC Framework**)
  - The framework is **already installed** on all SQL deployments (SQL 2008 R2+)
  - If present - the framework will appear in "*Programs and Features*"



Microsoft

Understanding a DACPAC

- For a database to be treated as a DAC it must be **registered**
  - The DAC **version** and **properties** are recorded in the **database metadata**
- In general **DAC is cross version supported** – from SQL 2008 R2
  - However - earlier SQL versions **cannot read** a DAC created in a later version
  - And - **not all objects are supported** within a DAC (see **here**)
- The key DACPAC functions
  - **REGISTER** – register the database as a DAC application
  - **UNREGISTER** - unregister the database as a DAC application
  - **EXTRACT** – extract the database in a DACPAC file
  - **DEPLOY** – deploy a DACPAC file onto a target SQL Server instance
  - **UPGRADE** – upgrade a database using a DACPAC file
- The key BACPAC functions
  - **EXPORT** – export a BACPAC file containing a DACPAC file with BCP data
  - **IMPORT** – import a BACPAC file to create a new database

Creating a DACPAC

- **From SSDT**
  - Function available via **VS Database Project**
  - Can "import' a **DACPAC** file to create a **new VS Database Project**
  - The **BUILD** output from a Database Project is a **DACPAC**
    - Writes DACPAC file to **...\bin\Debug\<Solution>.dacpac**
  - The **DACPAC** can be used to deploy the **entire** single database project solution
  - Forms a nice, simple, compartmentalised single database deployment unit

- **From SSMS**
  - "Extract Data-Tier Application" from an existing database – creates a DACPAC
  - Registers the database for DAC solutions
  - Not to be confused with "Export Data-Tier Application" – which creates a BACPAC

DEMONSTRATION
Creating a DACPAC

Deploying (aka Publishing) using a DACPAC

- **Key Points**
  - **Publish** and **Deploy** are often used interchangeably to rollout changes
  - Essentially the same as **upgrading**
- **From SSMS**
  - Deploy Data-Tier Application
  - *What for?* → Limited logging, not fully functioned, basic deployments
- **From SSDT**
  - In SQL Object Explorer Publish Data-Tier Application
  - *What for?* → Extensive options and logging, more than SSMS
- **From Command line Tool – SqlPackage.exe**
  - Lots of options and logging capability, more complex but fully featured
  - Located **C:\Program Files (x86)\Microsoft SQL Server\130\DAC\bin**
  - https://msdn.microsoft.com/en-us/library/hh550080(v=vs.103).aspx

Upgrading (Importing) using a DACPAC

- **Key Terminology Points**
  - **DRIFT** – report on changes made since the last time it was registered/deployed
  - **DEPLOY** – report on the changes that WOULD be made for a deployment

- **Upgrading** and **Publishing/Deploying** used Interchangeably
  - **Only SSMS** offers separate upgrading path to publishing path
  - All other methods do **publishing only** (*which integrates an upgrade path*)

- **Upgrading** and **Publishing** – Process Performed by Tools
  1. Check if source database has been updated out of band = **DRIFT report**
  2. Check schema differences between database and DACPAC = **DEPLOY report**
  3. Identifies data impacts that will be caused by the deployment (ie table drops, etc.)
  4. **Applies changes from DACPAC to a database to bring database inline with DACPAC**
  5. Process complete

DEMONSTRATION
Deploying and Upgrading a DACPAC

Advanced Methods with a DACPAC

- **DAC Comparisons, Database Comparisons**
  - Can compare **current DACPAC vs new DACPAC** in SSDT for differences
  - *(Good practice to do so before deployment)*
  - **Or at minimum**… compare a new DACPAC to the actual deployed database
  - https://msdn.microsoft.com/en-us/library/ee633948.aspx

- **Unpacking DACPAC**
  - Can review internal contents of DACPAC as its just a text file
  - Good to identify DACPAC contents, however **not necessary** in normal BAU operations
  - https://msdn.microsoft.com/en-us/library/ee633768.aspx

- **Pre and Post Deployment SQL Scripts**
  - Can **add additional scripts** as necessary which get bundled into the DACPAC for deployment
  - Good to add additional components such as SQL Agent Jobs, etc
  - https://msdn.microsoft.com/en-us/library/jj889461(v=vs.103).aspx

A Crash Course in Using BACPAC Files

- Essentially a **BACPAC** is **both** the schema (DACPAC) and table data
- Can be performed via **SSMS <u>only</u>**, there is **no option via SSDT**

- **Extract** - https://msdn.microsoft.com/en-us/library/hh213241.aspx
  - Creates a **DACPAC** with a "data" folder
  - All **selected tables** will be exported as **SQL BCP files**

- **Import** - https://msdn.microsoft.com/en-us/library/hh710052.aspx
  - Creates a **new database** based on definitions in the **DACPAC**
  - **Loads the data** in the "data" folder into the database
  - **Single operation** – cannot import over existing databases

DEMONSTRATION
Using a BACPAC

Microsoft

APPENDIX & REFERENCES

References

- Microsoft Data Tier Applications
  - https://msdn.microsoft.com/en-us/library/ee210546.aspx
- DACPAC Brain Dump
  - http://sqlblog.com/blogs/jamie_thomson/archive/2014/01/18/dacpac-braindump.aspx
- DACPAC for Database Lifecycle
  - https://www.simple-talk.com/sql/database-delivery/microsoft-and-database-lifecycle-management-dlm-the-dacpac/
- DAC Support for SQL Server Versions
  - https://technet.microsoft.com/en-us/library/ee210549(v=sql.130).aspx
- DAC in SQL Server 2008 R2
  - https://msdn.microsoft.com/en-us/library/ff381683(SQL.100).aspx
- Download Data Tier Application Framework
  - https://www.microsoft.com/en-us/download/details.aspx?id=52673

Common Questions and Answers

What permissions are required – to use a DACPAC

See Permissions here - https://msdn.microsoft.com/en-us/library/ee210546.aspx
See permissions here - https://msdn.microsoft.com/en-us/library/ee633653.aspx
See permissions here - https://msdn.microsoft.com/en-us/library/hh231291.aspx

How does "$Token" replacement work with DACPAC deployment (SQLCMD Variables)

See here - https://msdn.microsoft.com/en-us/library/hh272681(v=vs.103).aspx
Variables can be used to substitute tokens for servernames, environments, domains, logins, code paths, passwords, other complex logic, etc.

Can DAC deployment work across two databases

DACPAC is designed to be representative of one single complete database.
So you cannot deploy to multiple databases, and you cannot deploy multiple different DACPAC to the same database (well you can, but the 2nd DAC will overwrite the first DAC as they represent the current state only)

What DAC "version" gets created when a newer tool (ie SSMS 2016) extracts a DACPAC from an older SQL (ie SQL 2014)

This article spells out version differences, including DAC versions, between older and newer SQL versions/deployments - https://msdn.microsoft.com/en-us/library/ee210549.aspx

When DAC extracts users how are passwords protected

All database objects and instance level objects (ie logins mapped to database users) are extracted into a DACPAC.
The login passwords are stored in the extracted file in secure hashed format.

When DAC exports a database to a BACPAC what transaction is used on the source database

The export will use the default setting on the database – which when unchanged is READ COMMITTED.
As such the export is NOT considered transactionally consistent as it performs a SELECT * FROM <TABLE> each of the tables at different times. This is however the lowest impact and fastest export method.
To create a transactionally consistent copy the database must be either single user, read only, or taken from a separate copy (ie a backup/restore of a prod database where the BACPAC export is the only connection on it)