

Database Updates - Not As Easy As You Think

<https://github.com/christophwille>

All architecture is design, but not all design is architecture.

Architecture represents the significant design decisions that shape the form and function of a system, where significant is measured by the cost of change.

Grady Booch

High Optionality Programming

Software Architectures that Reduce Technical Debt

<https://petabridge.com/blog/high-optionality-programming-pt1/>

About Me

- My career in movie quotes
 - “The horror! The horror!” (Apocalypse Now)
 - “Nuke it from orbit” (Aliens)
 - “What did you see, old man?” (Godzilla)
- Just kidding

<https://github.com/christophwille/demosatconferences/tree/master/DotNetAustria0725>

A simplified problem statement

Yes, a demo

So... who is using...

- EF Migrations?
- Other?

Talk Target Db: SQL Server

- Not Cosmos (even though EF can do that)
- Not SQLite (even though EF can do that)
- ...and we'll get back to that

Agenda

- EF Migrations
- SQL Database Projects
- More Tools for You
- The Advice Section

Generic Approach Comparison

Model-based

- Easier (less control)
- Better for sprocs/functions
- Better for large/distributed teams
- Better for frequent changes
- Better for dependency nightmares
- Drift: rolled back
- Better for development


Migration-based

- More control (more responsibility)
- Better for data migrations
- Better for small teams
- Better for infrequent changes
- Better for simple data stores
- Drift: ignored
- Better for automation

EF Migrations

Minus any 101

EF Migrations

- A very good go-to solution
- Most likely also the best starting point
 - Do use ModelBuilder (indexes, precision, length do matter!)
- These days very mature
 - EF bundles
 - Custom migration operations 
 - <https://learn.microsoft.com/en-us/ef/core/managing-schemas/migrations/operations>
 - Specific Azure SQL support
- If you need a refresher, go to
 - <https://www.milanjovanovic.tech/blog/efcore-migrations-a-detailed-guide>

EF Migrations - a Moving Target

- <https://learn.microsoft.com/en-us/ef/core/what-is-new/ef-core-9.0/whatsnew#migrations>
 - Protection against concurrent migrations
 - Warn when multiple migration operations can't be run inside a transaction
 - Improved data seeding

EF Migrations – Surprises Included

- It will mostly detect column renames
 - Check the generated SQL
- Feature branch merges to main can break
 - Add a BVT
- It might OOM on you
- It doesn't make you database agnostic ⚠
 - SQL (sprocs, views), custom extensions, engine quirks
- Not everything is available
 - Column store, partitioning, column encryption, ...
- Your db experts talk SQL, not C#
 - True as well for eg FluentMigrator (less so DbUp)

Demo time

- Column rename
- Snapshot testing

SQL Database Projects

Tales from the crypt (No, It is alive!)

SQL Database Projects (Ms.Build.Sql)

- This thing has a history (SQL projects)
 - CI was an issue (non-xplat)
- Works now in VS, VSCode and SSMS
- Easily extract from existing database
- dacpac/bacpac are actually quite nice things

Demo time

- Time-travel to Pubs
- Deploying a dacpac

What about MSBuild.Sdk.SqlProj?

- Origin story
 - <https://jmezach.github.io/post/introducing-msbuild-sdk-sqlproj/>
- Does a lot more things
 - <https://github.com/rr-wfm/MSBuild.Sdk.SqlProj/blob/master/README.md>
- So which one to choose?
 - poc/DbPlayground comes with a sample

<https://github.com/ErikEJ/SqlServer.Rules> works with both!

More Tools for You

There is no one size fits all (or silver bullet)

Detour – PostgreSQL

- Why would you choose it over SQL Server?
 - De-Americanize (or OSS-ize)
 - Community around it (more OSS tools)
 - Yes, you guessed it: ARM64
- graphile-migrate
 - <https://github.com/graphile/migrate>

<input type="checkbox"/>	<input type="radio"/>	mcr.microsoft.com/mssc	2022-CU18-ubuntu-22.04	2b41d0be8283	2 months ago	1.65 GB	▶
<input type="checkbox"/>	<input type="radio"/>	postgres	17.4	f49abb9855df	2 months ago	438.41 MB	▶
<input type="checkbox"/>	<input type="radio"/>	postgres	15-alpine	afbf3abf6aeb	2 months ago	273.1 MB	▶

Detour – Azure Data Studio

<https://learn.microsoft.com/en-us/azure-data-studio/whats-happening-azure-data-studio>

- Why did it matter
 - Xplat tool
- Replacment (MS)
 - VSCode mssql (but not quite)
- Replacement (OSS)
 - DbGate <https://github.com/dbgate/dbgate>
 - Antares(*) <https://github.com/antares-sql/antares>

“Other” Migration Tools (Free & \$\$\$)

- <https://github.com/flyway/flyway>
 - <https://github.com/DbUp/DbUp>
 - <https://github.com/bytebase/bytebase>
 - <https://github.com/fluentmigrator/fluentmigrator>
 - <https://github.com/liquibase/liquibase>
 - <https://github.com/dbeaver/dbeaver/>
 - <https://github.com/lecaillon/Evolve>
 - <https://github.com/sqitchers/sqitch>
-
- Honorable mention <https://github.com/JasperFx/weasel>

Demo time (or homework)

- DbUp
 - <https://github.com/Azure-Samples/azure-sql-db-aspire?tab=readme-ov-file#aspire-hosted-sql-server--dbup--dab>
- ByteBase
 - <https://www.bytebase.com/docs/get-started/step-by-step/change-schema/>

The Advice Section

Your projects are different from mine!

Testing Migrations – No Empty Db Please

- Why?
 - Changing nullable column to non-nullable will only fail when there is data
 - Decreasing string column size will only fail when there is data
 - Decreasing precision – you guessed it
- Again, tools are available (Free & \$\$\$), eg
 - <https://github.com/shuttle-hq/synth>
 - <https://github.com/necatianslan/table-faker>
 - <https://blog.devart.com/generate-test-data-with-sql-data-generator.html>
 - Search for “populate database fake|random data”
 - DON'T USE PRODUCTION DATA!

Applying Migrations: Split DDL and DML

- Read “Don't Let Your Permissions be Hijacked!”
 - <https://www.sommarskog.se/perm-hijack.html>
- Lowdown – always use least privilege!
- Yes, I like security very much

Applying Migs: Backward-Compat Changes

- This is mandatory if HA is of any concern
- View it like non-breaking API changes
 - new columns must be nullable eg
- Might mean that changes need to be split into multiple parts

CI aka “Automated builds”

- If it is C#, that should be easy
- Is it platform dependent?
 - Linux / Windows
 - x64 / arm64
- Do yourself a favor and add a BVT

CD (Deployment or Delivery)

- Init Containers?
- During startup of the application?
- Separate CD step?

E.g. <https://www.milanjovanovic.tech/blog/streamlining-dotnet-9-deployment-with-github-actions-and-azure#extending-your-cicd-pipeline>

Roll back or Roll forward?

- Which one to choose when things go wrong?
- Recovery Time Objective, RTO
- Recovery Point Objective, RPO
- If you want a heated tabs-vs-spaces argument, go for it!

A (Not The) Future

- Copilot today actually works very nicely with ModelBuilder
- “can you create a new table for signup logging. Use an incremental counter for primary key and store the timestamp for each log”
 - <https://newsletter.pragmaticengineer.com/p/mcp?open=false#%C2%A7lm-layer-above-postgresql>

Summary

- Choosing a database can be an expensive decision
- Think beyond day 0 (high optionality)
- Don't limit yourself to the .NET ecosystem
- “It depends” – as usual