

Next-gen SQL projects

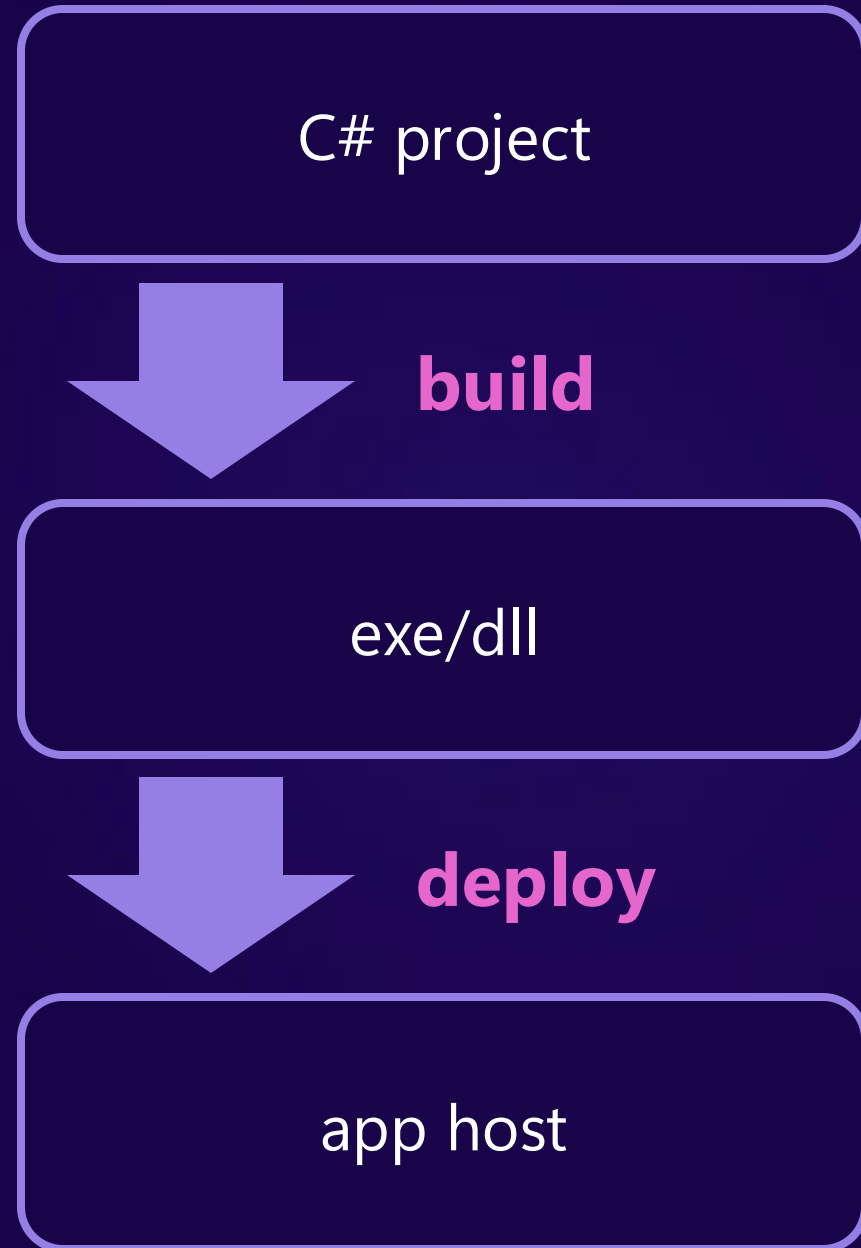
with Microsoft.Build.Sql



Drew Skwiers-Koballa

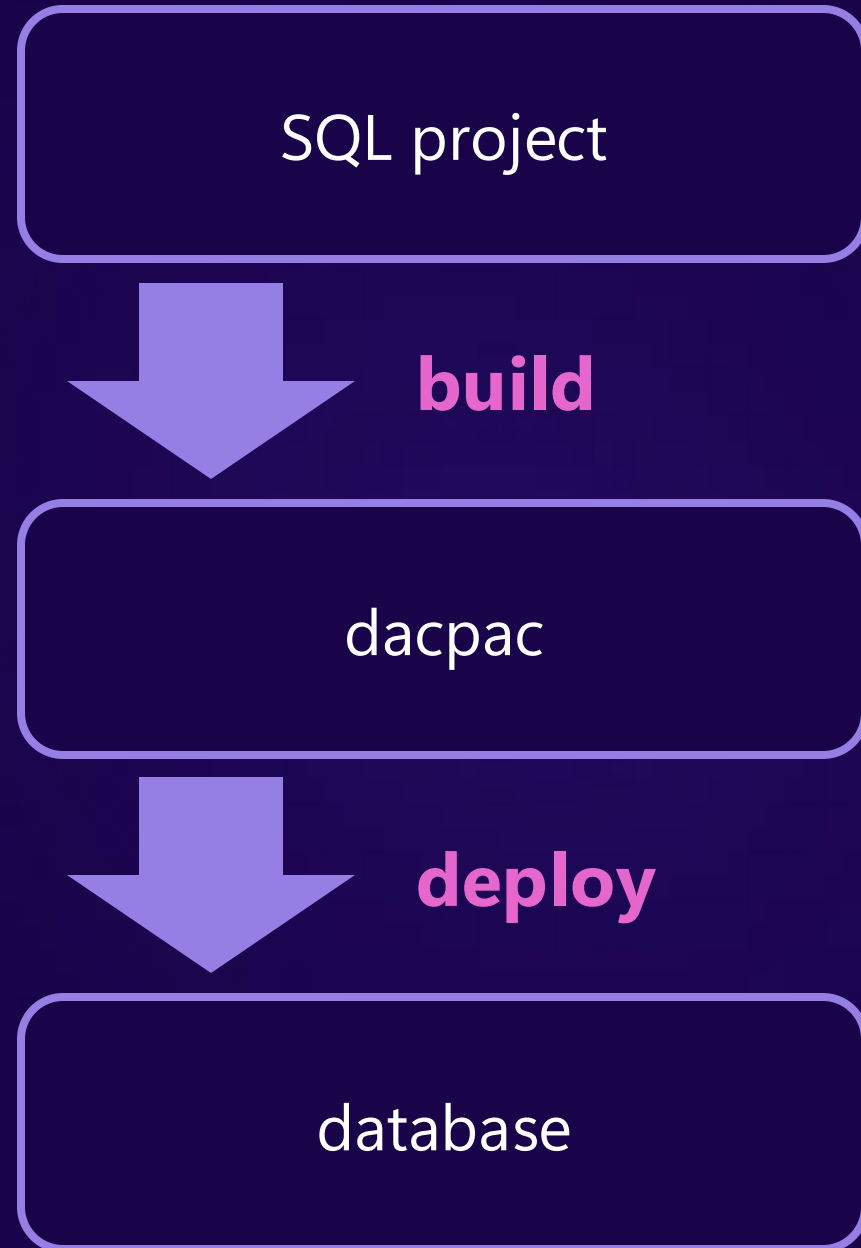
.NET project lifecycle

Build and ship



SQL project lifecycle

Build and ship





SQL (database) projects

.NET project type where SQL code is validated as a database object model

C# → SQL

CREATE obj (tables, stored procs, foreign keys...)

dacpac

Compiled artifact from a SQL project, contains a model representation of SQL object definitions

dll → dacpac

Microsoft.Build.Sql

Project SDK from Microsoft for SQL projects, iteration from original SQL projects (SQL Server Data Tools, SSDT)

Cross-platform .NET support

publish

API name for deploying a dacpac, creates a new database or updates existing

Toolbox basics



SqlPackage CLI

```
dotnet tool install -g microsoft.sqlpackage
```



SQL projects templates

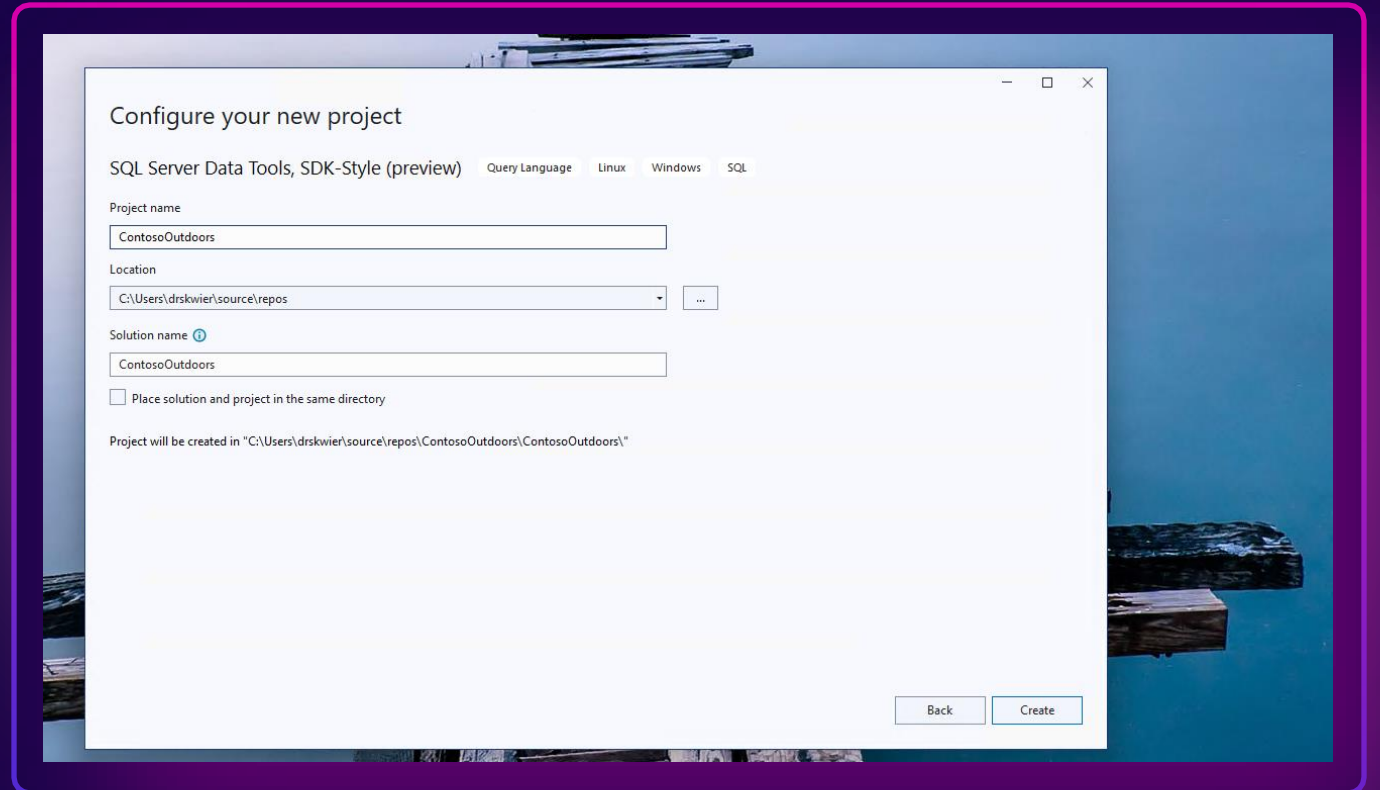
```
dotnet new install microsoft.build.sql.templates
```

Everything else is dotnet

Getting started

Use a new or existing database

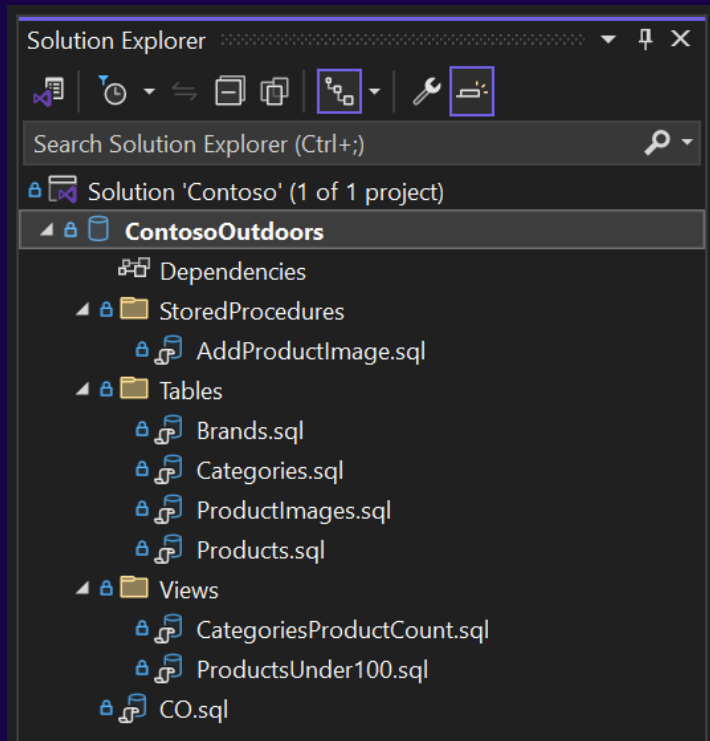
- SqlPackage extract to SQL files
- dotnet new sqlproj
- “Create project from existing database”
- New project



Latest IDE updates

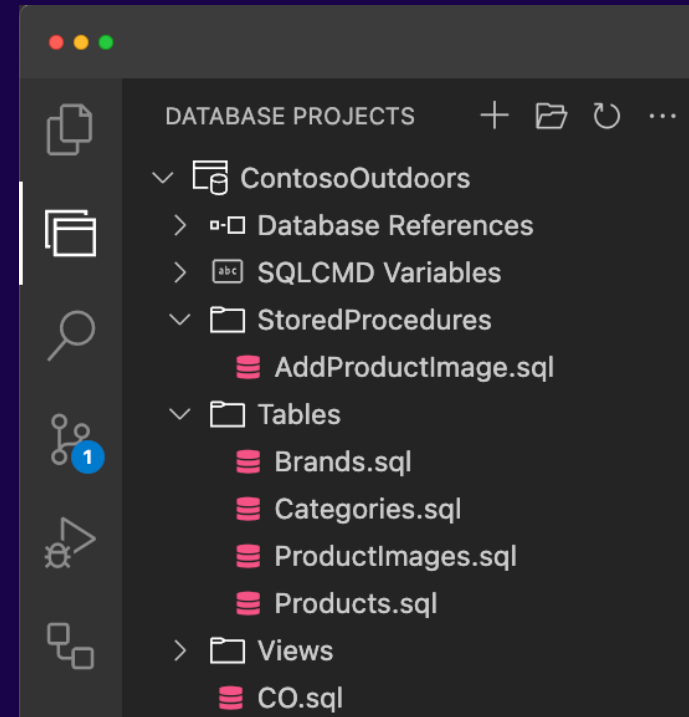
Visual Studio 2022 17.12

- Introduces SDK-style SQL projects (preview) component
- Ongoing development to bring more SSDT features to 17.13 and beyond



VS Code

- SQL Database Projects extension
- Ongoing development to add features, including schema compare



Code check-in



Is it valid?

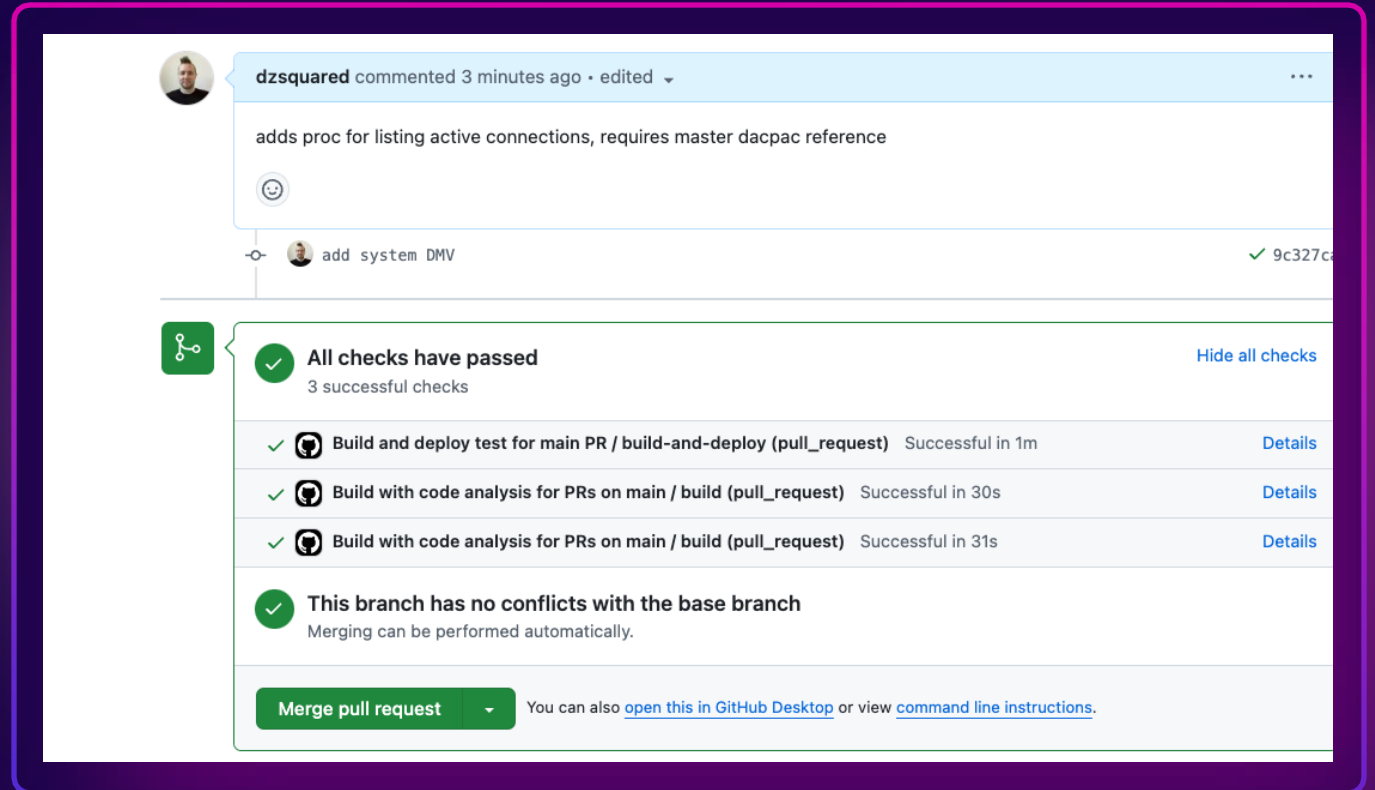


Is it good?

CI checks on SQL projects

Build with SqlCodeAnalysis

- *dotnet build* covers validating SQL syntax and model integrity
- *SqlCodeAnalysis* property enables code analysis checks
- *Package references* extend code analysis from default set
- *Service containers* provide an ephemeral endpoint for unit tests



The screenshot displays a GitHub pull request interface. At the top, a comment by user 'dzsquared' states 'adds proc for listing active connections, requires master dacpac reference'. Below the comment, a commit titled 'add system DMV' is shown with a green checkmark and the commit hash '9c327ca'. The main section of the interface is a green box indicating that all checks have passed. It lists three successful checks: 'Build and deploy test for main PR / build-and-deploy (pull_request)' (Successful in 1m), 'Build with code analysis for PRs on main / build (pull_request)' (Successful in 30s), and 'Build with code analysis for PRs on main / build (pull_request)' (Successful in 31s). Below these, a green checkmark indicates 'This branch has no conflicts with the base branch' and that 'Merging can be performed automatically.' At the bottom, there is a green button labeled 'Merge pull request' and a link to 'open this in GitHub Desktop' or view 'command line instructions'.

dzsquared commented 3 minutes ago • edited

adds proc for listing active connections, requires master dacpac reference

add system DMV

9c327ca

All checks have passed
3 successful checks

Hide all checks

- ✓ Build and deploy test for main PR / build-and-deploy (pull_request) Successful in 1m Details
- ✓ Build with code analysis for PRs on main / build (pull_request) Successful in 30s Details
- ✓ Build with code analysis for PRs on main / build (pull_request) Successful in 31s Details

✓ This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

Database references



Dacpac reference

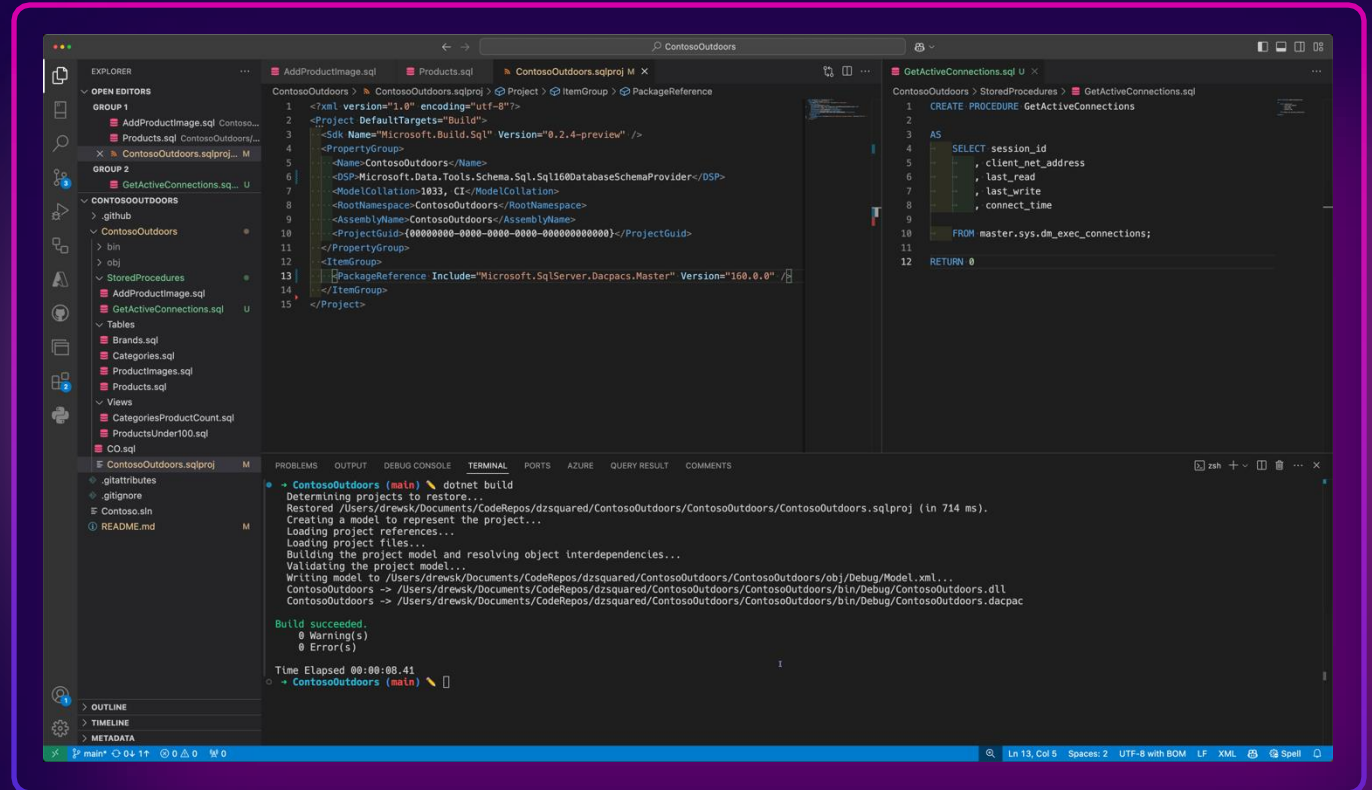


Package reference

Reference additional database objects

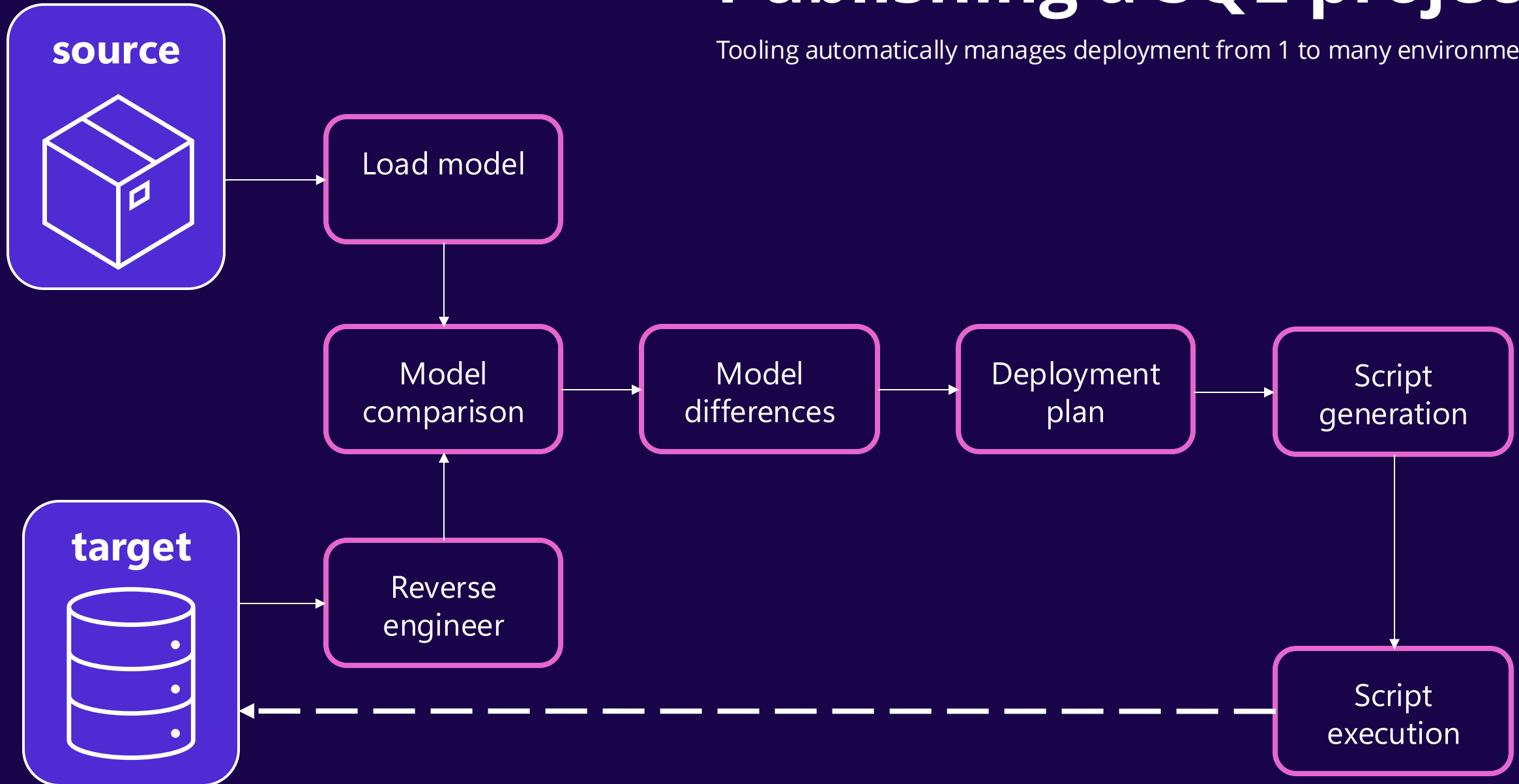
Architect smarter

- *dotnet pack* to prepare your SQL project for publishing to a package feed
- Improve build times by moving infrequently changing objects to a static package
- Reference objects across databases (3-part naming with SQL Server)
- *System dacpacs* published for package references



Publishing a SQL project

Tooling automatically manages deployment from 1 to many environments



Deployment options



Publish



Deploy report



Script

Choose by environment

Dynamic but lightweight

- Directly publish to apply the calculated deployment plan to a SQL database
- Generate the deploy report to monitor for unwanted changes
- Output deployment script to review T-SQL generated for the deployment plan

The screenshot displays a GitHub Actions workflow run interface. The title bar shows a green checkmark and the text "Build and generate deployment script from main for prod environment #2".

Summary

Manually triggered 4 minutes ago | Status: **Success** | Total duration: 1m 11s | Billable time: 2m | Artifacts: 1

Jobs

- build-and-deploy

Run details

- Usage
- Workflow file

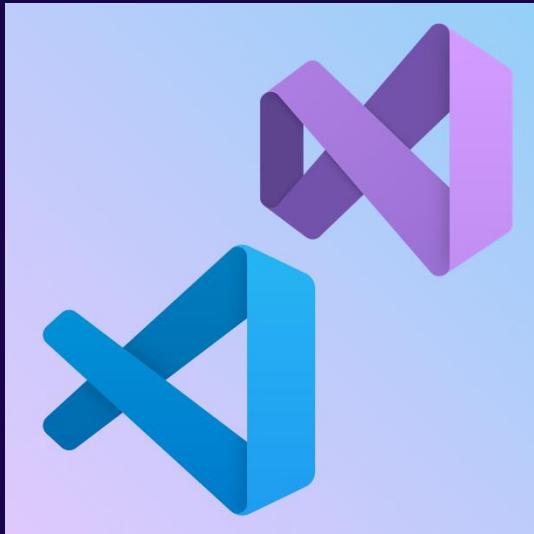
manual-deployscript-prod.yml
on: workflow_dispatch

build-and-deploy 1m 2s

Artifacts
Produced during runtime

Name	Size
deployment-script	1.65 KB

SQL projects tooling ecosystem



Visual Studio and VS Code

Develop, analyze, and compare database objects



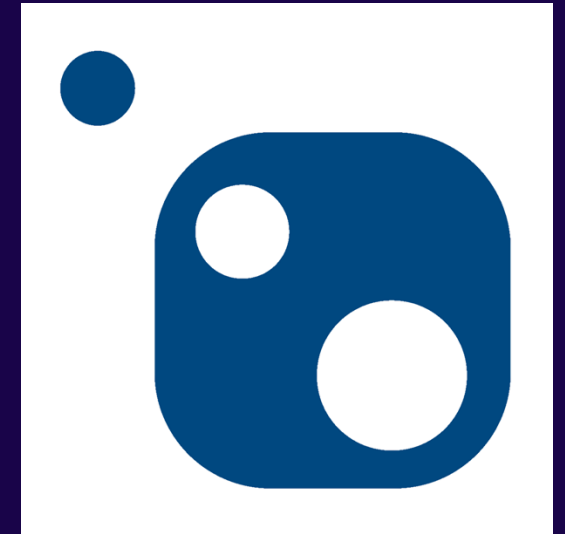
GitHub and Azure DevOps tasks

Streamline SQL project deployment from CI/CD environments



SqlPackage CLI

Automate transposing between files and databases



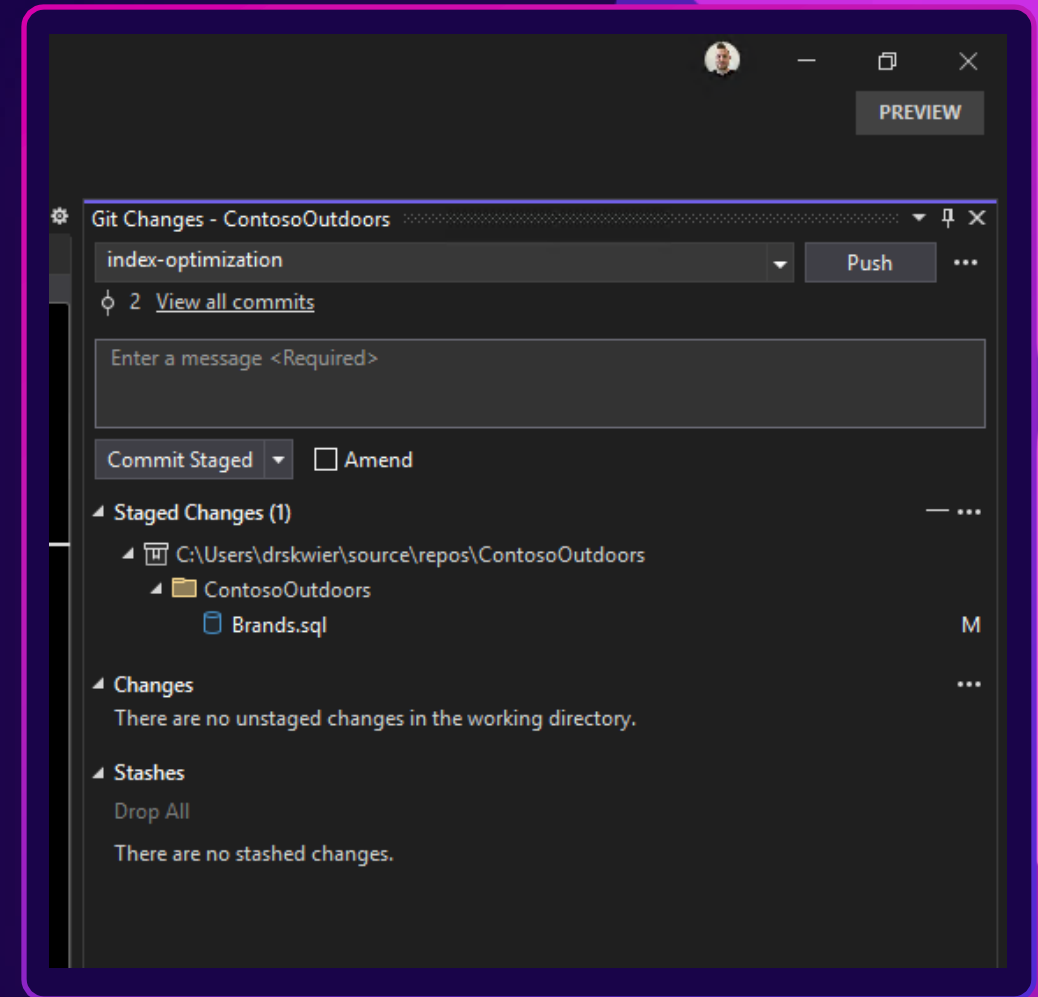
API, class, and model packages

Extensibility for core APIs includes loading analysis rules and database parts from packages

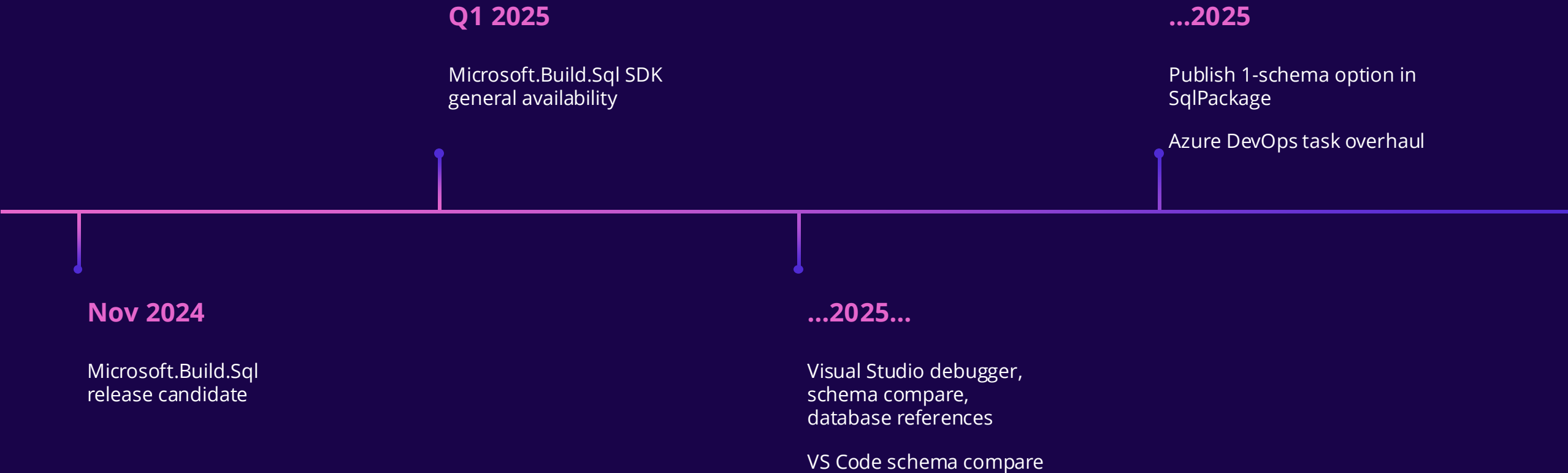


SQL Server Management Studio 21
now in preview

aka.ms/ssms21-preview-blog



SQL projects roadmap





Learn more



Documentation
aka.ms/sqlprojects



SQL projects examples



Clone the repo
aka.ms/sqlprojects-samples



Get .NET 9



Download .NET 9
aka.ms/get-dotnet-9

Thank you

