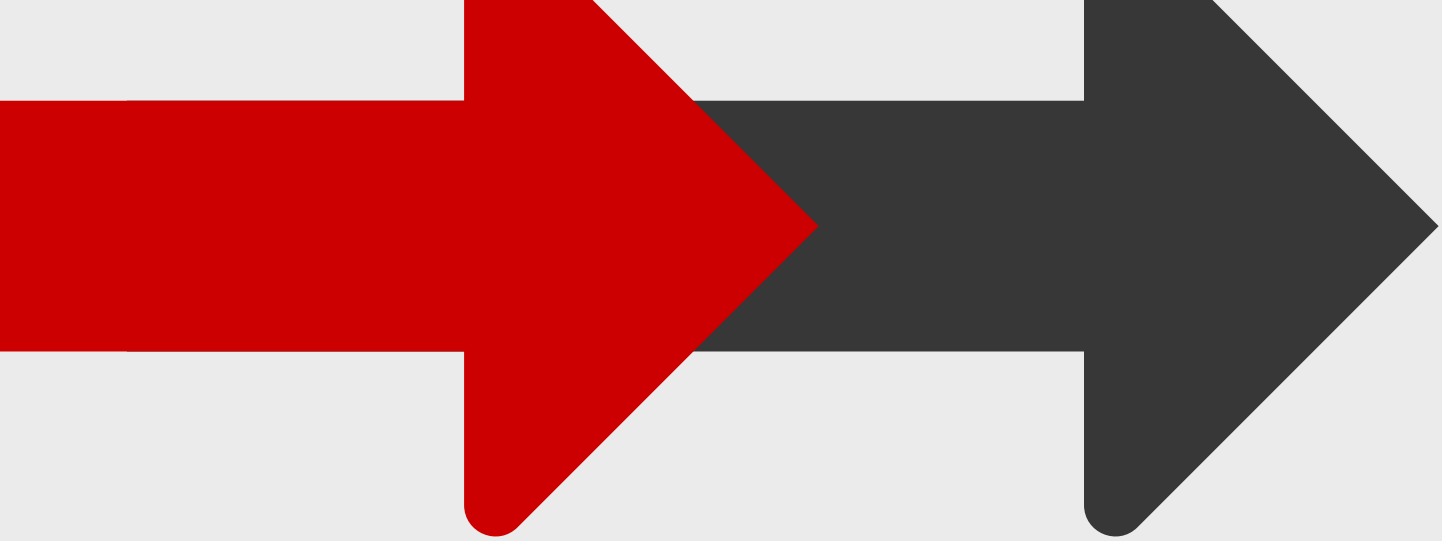


State- or migrations-based database development

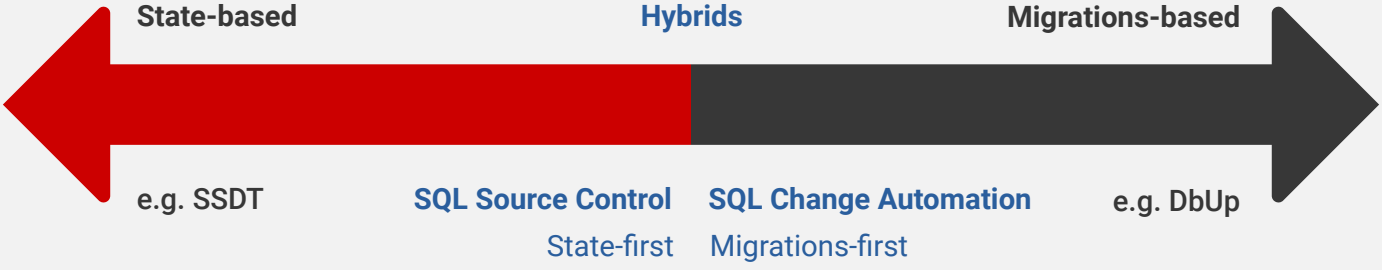
redgate

State- or migrations-based database development

While those of us in the SQL Server world have benefited from tools which help us design the end-state of the database and work out how to make the changes correctly, script-based processes have moved into the background. However, SQL Change Automation now gives new life to script-based deployment processes.

This paper outlines the differences between what's referred to as the model or state-based approach and the migrations-based approach, and the corresponding Redgate tools. Both approaches, at their root, are about which is your source of truth: the definition of how you want the database to look, or the scripts you produce to handle the upgrade.

State-based or migrations-based tools typically sit at either end of a spectrum and are exclusive in their approach. Redgate's tools primarily follow one approach or the other, but both SQL Source Control and SQL Change Automation benefit from hybrid features, explained below, designed to address the shortcomings associated with the extreme ends of the spectrum:



SQL Source Control is primarily state-based but, unlike pure state-based tools such as Microsoft's SQL Server Data Tools (SSDT), it can mix and match state with the occasional migration.

SQL Change Automation on the other hand is primarily migrations-based, but rather than being a pure migrations-based tool like DbUp, it has the option of using state to manage textual objects.

redgate

	SQL Source Control, *the state-first approach*	SQL Change Automation, *the migrations-first approach*
Definition	Aims to enable a single step between the current database state and the desired database state. The current state of each object in the database is versioned as a CREATE script. **SQL Source Control** compares the source to the target and auto-generates a script to synchronize the two states using the SQL Compare engine.	Captures individual change scripts during development to effect larger, iterative database migrations. **SQL Change Automation** auto-generates numerically ordered migration scripts using the SQL Compare engine. Changes (often ALTER commands) are organized in SQL scripts and run in order to migrate a database from one version to the next.
Development model	Both SQL Source Control and SQL Change Automation support the preferred connected development model, allowing users to modify the database directly in SQL Server Management Studio (SSMS).	
IDE	SQL Source Control integrates your version control system directly within SSMS.	SQL Change Automation plugs into Visual Studio (VS). In support of the connected model, changes made in SSMS are recognized by SQL Change Automation and can be imported into the Project in VS to be versioned as part of your solution.
Scenario	Provides an easy development workflow for large teams who need to collaborate on complicated databases with multiple dependencies.	Works well for databases which are simple data stores where most of the changes are table refactorings or data migrations.
Deployment	Deployment scripts are inferred at deployment time using the SQL Compare engine, requiring less discipline and database expertise by developers.	Developers and DBAs collaborate early in the development cycle to define the deployment steps, improving agility, testing and repeatability.
	Assisted deployments can be carried out via the SQL Compare UI, with full transparency to review changes prior to deployment.	Manual deployments can be executed via a package sql script.
Automation	Both SQL Source Control and SQL Change Automation have command lines and build/release tool integrations that make continuous integration and automated deployments simple.	
Drift	Using SQL Source Control or SQL Change Automation, developers benefit from drift detection functionality. Additionally, Redgate's free DLM Dashboard can alert users to drift as soon as it occurs.	
Hybrid features	Some changes such as column splits or similar data motion scenarios can't be easily managed via a comparison approach because the tools don't understand the context of the change. SQL Source Control offers an occasional migrations feature that enables users to hard code certain parts of the deployment process.	Migrations-based approaches can introduce a "last one in wins" scenario, in which the last check-in could overwrite previous migrations. To avoid this, users can take advantage of SQL Change Automation's Programmable Objects feature for a state-based approach for managing functions and stored procedures.

www.red-gate.com/sql-source-control

www.red-gate.com/sql-change-automation

redgate