SQL Server DevOps

Scott Sauber


scottsauber

Slides up at scottsauber.com

Titanium Sponsors

Platinum Sponsors

Gold Sponsors

Audience

- Developers + DBA's who want to automate their SQL Server builds and deployments
- Unsure where to get started
- Familiar with SQL Server and it's terminology
  - Tables, Views, Stored Procedures, Functions, Triggers, etc.
- Poll
  - Devs?
  - DBAs?
  - Managers?
  - Other?

Agenda

- What choices do I need to make?
- What are the tradeoffs?
- What tools are out there?
- What hurdles will I face?
- Demos
- Questions

Purpose

- Ramp up on how to implement automation to your SQL Server
- Know the options and tradeoffs of different approaches and tools

Who am I?

- Software Consultant at Lean TECHniques
- Developer (not a DBA) and big proponent of DevOps
- Successfully implemented SQL DevOps Pipeline for over a dozen db's
  - Including 25 year old SQL Server db
- Blog at scottsauber.com

scottsauber

Types of team interactions with DB's

- Devs write, review, and deploy the SQL. No dedicated DBA.
- Devs write the SQL and give to DBA to review and deploy.
- Devs tell DBA's what they want, DBA's write, review and deploy the SQL.

77%

Yes - our developers are responsible for both database and application development

23%

No - our team has dedicated database developers

What a manual workflow may look like today

- Compare approach (i.e. RedGate SQL Compare)
  - Developer/DBA works on a development DB
  - That DB is then compared to a Prod or Prod-like DB to compare changes
  - Tool generates script to deploy
  - Script is deployed to Prod
- SQL Script approach
  - Developer/DBA works on a development DB
  - Developer/DBA accumulates scripts
  - Developer/DBA runs scripts against Prod

What's wrong with these approaches?

- No Source Control
  - No traceability
  - No easy rollbacks
- Manual
  - Tedious
  - Easy for mistakes
  - Script order can get out of whack
- Development DB + Prod DB could be out of sync
  - Changes in behavior
  - Overwriting others (sprocs, views)
- Hard to pull in others changes (no forced CI)

What is DevOps?

"DevOps is the union of people, process, and products to enable continuous delivery of value to our end users."

- Donovan Brown

What is DevOps?

"DevOps is the union of **people**, **process**, and **products** to enable continuous delivery of value to our end users."

- Donovan Brown

What is DevOps?

"DevOps is the union of people, process, and products to enable continuous delivery of **value** to our **end users**."

- Donovan Brown

What is DevOps?

"You can't change culture and process with a credit card."

- Julie Gunderson

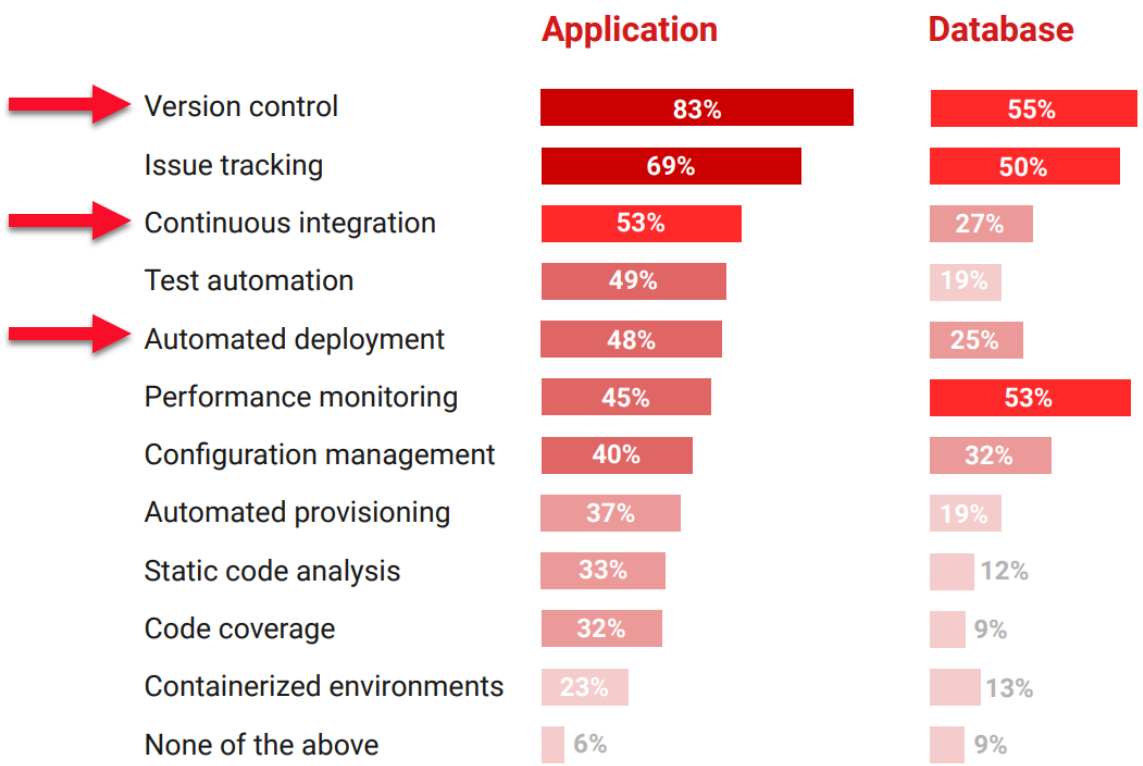
Desired Outcomes of SQL Server DevOpsifying

In order of importance:

1. Database is source controlled
2. Database deployments are at most a single click of a button
3. Database builds for verification on each commit
4. Monitor production database for out-of-band changes
   - This one is fun because people.

Desired Outcomes of SQL Server DevOpsifying

14 **Which, if any, of these practices are already in place for your application or database development?**



scottsauber

Why is this hard?

- Source Control traditionally not built-in to SQL Server tools (SSMS)
- Sins have been committed in your legacy databases over time
  - Linked servers, DB hopping
- Requires Devs + DBA's to talk to each other
- DBA's think they are getting cut out
- Spoiler: they're not. The crappy part of their job is, so they can do more value add work.
- Redgate article on "Implementing DevOps Doesn't Get Rid Of Database Administrators"

Why is this hard?

- 66% of companies do not have automated builds and deploys for their databases
  - Per Redgate Survey
- The database is stateful, applications are not (or shouldn't be.)
- Rollback of an app is "delete all these files and replace them with these ones…."
- Rollback of a database requires thought

Let’s get to DevOpsing

Source Control: What

- Schema Structure
  - Tables, Views, Stored Procedures, etc.
- Static Data
  - Data required for the application to run successfully
  - Lookup Tables, Roles table for users in a system, Configuration values, etc.

Source Control: How - Methodologies

- Model-based
- Migration-based

Source Control: Model-based

- Build an “ideal model” of your DB.
- Let a tool figure out how to migrate your Production DB to that ideal model.
- Examples of tools: Redgate SQL Source Control and Microsoft DACPAC
- I do not prefer this approach
  - Scenarios like Column Renames
  - Minimal insight into “how” it got there.
- This approach is losing mindshare

Source Control: Migration-based

- Every change is scripted
- Scripts are committed to source control
- Scripts run in order (date-based or #-based)
- Which scripts have run are kept track in a table
- Write them up front during dev
- Run the same scripts in every environment
- Examples of tools: Redgate SQL Change Automation (hybrid), Flyway, DbUp, and RoundhousE
- Migration-based is my preferred approach

Source Control: How

- Database Code + Application Code should live together in the same Source Control Repository
- One Pull Request/Commit/Checkin for the application code and SQL code
- Ideally Final Schema and Migrations live together

Tool Review: Flyway

- Open source
- Migration-based approach
- Command line tool
- $950/yr for 10 schemas for Pro, $2950/yr/10 schemas for Enterprise

Flyway Demo

- Demo
- Config
- Baseline
- Migrate

Tool Review: Redgate SQL Change Automation

- Migration-first approach, but hybrid
- Also supports showing the final state of your application
  - Not used for deployment
- Visual Studio Extension
- Make changes in DB, use VS to get those changes as migration scripts
- SSMS Extension in Beta
- Builds your DB from scratch as a dry-run
- Integrates with MSBuild, Azure DevOps, TeamCity, and Octopus Deploy
- Need SQL Toolbelt (which comes with 13 other tools)
- $3095/user for 1 yr of support, $4333/user for 3 yrs

Redgate SQL Change Automation Demo

- Demo
- Schema
- Static data
- Stored Procedure

Automated Builds: How

- Responsibilities:
  - Take migrations and deploy them to an independent DB
  - Spin up new DB for you or have dedicated CI DB
- Use a Build tool
- Azure DevOps
- Jenkins
- TeamCity

Automated Deployments: How

- Responsibilities:
  - Deploy to each Environment
  - Swap out secrets (i.e. connection strings)
- Use Deployment tool
- Octopus Deploy
- Azure DevOps
- Jenkins
- TeamCity
- Bamboo

Proposed Workflow

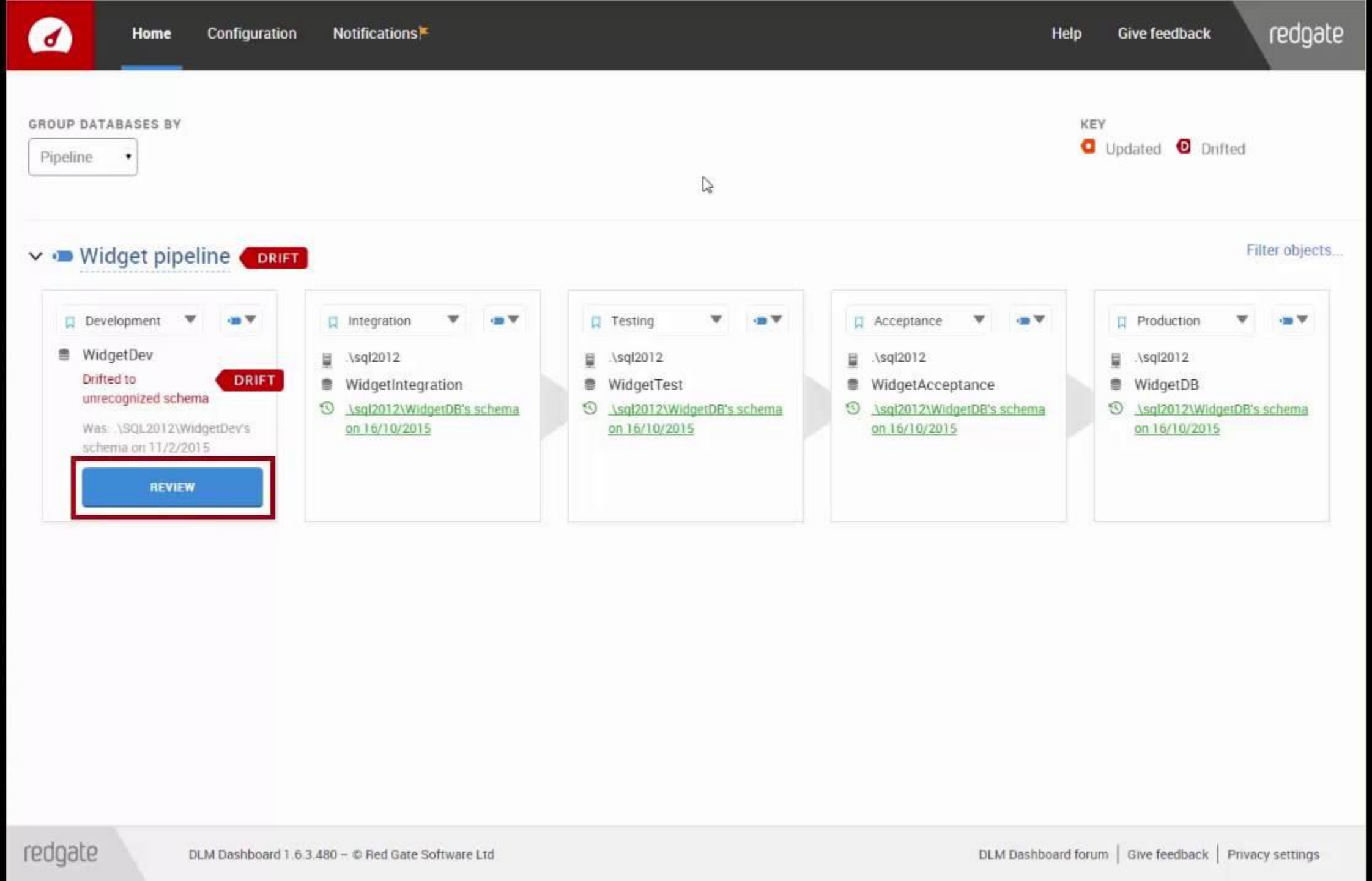
1. Developer adds their application code and SQL code
2. Developer commits to a branch and sends a Pull Request for code review
3. Another developer reviews the application code
4. DBA/senior person reviews SQL code
5. When both are approved, the code is merged into master
6. Code is built independently to verify the commit
7. Deployments are then a button push

Workflow Demo

- Demo
- Add Migration
- Reviewed by DBA/senior person
- Build server
- Deploy changes to databases
- Promote through environments

Tool Review: Redgate DLM Dashboard

- Monitors and alerts on SQL Server schema changes
- Audit history of changes
- Finds when someone bypasses the pipeline


Home
Configuration
Notifications
Help
Give feedback
redgate
GROUP DATABASES BY
Pipeline
KEY
Updated
Drifted
Widget pipeline
DRIFT
Filter objects...
Development
WidgetDev
Drifted to unrecognized schema
DRIFT
Was: \SQL2012\WidgetDev's schema on 11/2/2015
REVIEW
Integration
.\sql2012
WidgetIntegration
\sql2012\WidgetDB's schema on 16/10/2015
Testing
.\sql2012
WidgetTest
\sql2012\WidgetDB's schema on 16/10/2015
Acceptance
.\sql2012
WidgetAcceptance
\sql2012\WidgetDB's schema on 16/10/2015
Production
.\sql2012
WidgetDB
\sql2012\WidgetDB's schema on 16/10/2015
redgate
DLM Dashboard 1.6.3.480 – © Red Gate Software Ltd
DLM Dashboard forum
Give feedback
Privacy settings

Common Gotcha's Building The Database

- You will pay for the sins of your ancestors
- Linked Servers
- Cross-Database Hopping
- Old Stored Procedures or Views Referencing Old Tables/Columns
- Temp Table creation in Stored Procedures

A Word On Rollbacks

- They are usually not worth the headache
- Why did the deployments succeed in Dev, UAT, etc. but not in Production?
  - Almost always a failure in people and/or process
- How do you rollback something destructive (DROP, DELETE, TRUNCATE, etc.)?
- Contextual
- Tradeoffs
  - Restore from backup but lose data in between deployment and restore.
- Instead: Roll forward

People Challenges

- Mindset shift
- The more you can force “no one has Prod” access the better
- Force everything to go through the pipeline.
- Danny the Deployer
  - Doesn’t fully buy in to Source Controlling the DB
  - Goes directly to Prod without Source Controlling
  - Inevitably causes pain later
  - “I’ll just do this, this one time.”
- Devs + DBA’s Need To Work Together
- Customer focus

Takeaways

- Choose a migrations-based approach
- Source Control your DB
- Auto Deploy the Source Controlled Migrations
- Tools you can use
- Gotcha’s – tools, existing DB sins, and people
- You can do this – the question is, does your organization want to?

Resources

- Redgate Simple Talk Blogs
- Redgate Database DevOps Blogs
- Redgate 2019 State of DevOps Survey
- Redgate Training on SQL Change Automation
- Redgate YouTube Channel on SQL Change Automation
- DB DevOps with Jeffrey Palermo and Paul Stovell

And Now For Something Completely Different

- Also giving a HTTP Security Headers talk
- 1 PM Room 2209 on Friday

Questions?

Thanks!