DevOps for Databases

scottsauber

Audience

- Developers + DBA's who want to automate their database builds and deployments

- Unsure where to get started

- Poll
  - Devs?
  - DBAs?
  - Managers?
  - Other?

scottsauber

Agenda

- What choices do I need to make?

- What are the tradeoffs?

- What tools are out there?

- What hurdles will I face?

- Demo

- Questions

scottsauber

Goals

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

Who am I?

- Director of Engineering at Lean TECHniques
- Co-organizer of Iowa .NET User Group
- Automated dozens of databases
- Friend of Redgate
- Blog at scottsauber.com

scottsauber

Typical manual workflows for database changes

- Devs write, review, and deploy the SQL

- 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

scottsauber

What a manual workflow may look like today

- Compare approach (i.e. RedGate SQL Compare, pgadmin4)
  - 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
- Gather SQL Scripts approach
  - Developer/DBA works on a development DB
  - Developer/DBA accumulates scripts
  - Developer/DBA runs scripts against Prod

scottsauber

What's wrong with these approaches?

- No Source Control
  - No traceability

- 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)

scottsauber

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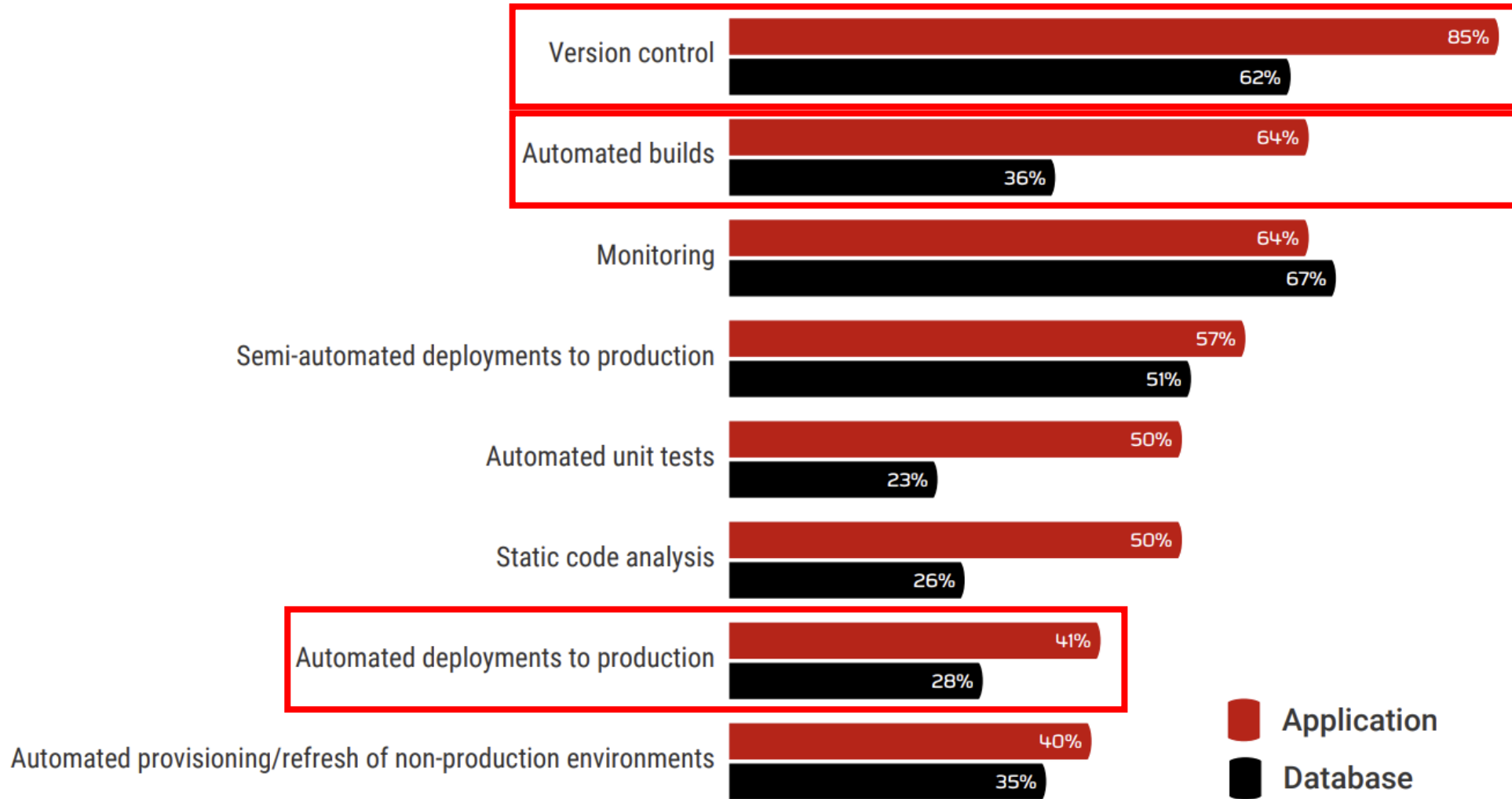
Donovan Brown

Microsoft

Desired Outputs of Database 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

scottsauber

Adoption of DevOps practices: application vs. databases

Practice	Application	Database
Version control	85%	62%
Automated builds	64%	36%
Monitoring	64%	67%
Semi-automated deployments to production	57%	51%
Automated unit tests	50%	23%
Static code analysis	50%	26%
Automated deployments to production	41%	28%
Automated provisioning/refresh of non-production environments	40%	35%

scottsauber

Why is this hard?

- 72%+ of companies do not have automated builds and deploys for their databases

- Source Control traditionally not built-in to Database 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.

scottsauber

Why is this hard?

- 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

scottsauber

Why is this hard?

- It's a lot easier on greenfield databases
- Start it from the very beginning of a new database

scottsauber

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.

scottsauber

Source Control: How - Methodologies

- Model-based
- Migration-based

scottsauber

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

scottsauber

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: Flyway, EF Migrations, DbUp, and RoundhousE

- Migration-based is my preferred approach

scottsauber

Source Control: How

- Database Code + Application Code should live together in the same Source Control Repository
  - Assuming not a shared database between many apps
- One Pull Request/Commit for the application code and SQL code
- Ideally Final Schema and Migrations live together

scottsauber

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
- GitHub Actions
- Azure DevOps
- Jenkins
- TeamCity

scottsauber

Automated Deployments: How

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

scottsauber

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 main

6. Code is built independently to verify the commit

7. Deployments are then a button push

scottsauber

Demo using
Flyway + GitHub Actions

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

- Multi-tenant challenges

scottsauber

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

scottsauber

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

scottsauber

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

scottsauber

You can do this…
…does your organization want to?

Resources

- https://github.com/scottsauber/database-devops
- Redgate Simple Talk Blogs
- Redgate Database DevOps Blogs
- State of DB DevOps 2021 Survey
- DB DevOps with Jeffrey Palermo and Paul Stovell

Slides at scottsauber.com

scottsauber

Questions?

Follow up: ssauber@leantechniques.com

Slides up at scottsauber.com

scottsauber

Thanks!

scottsauber