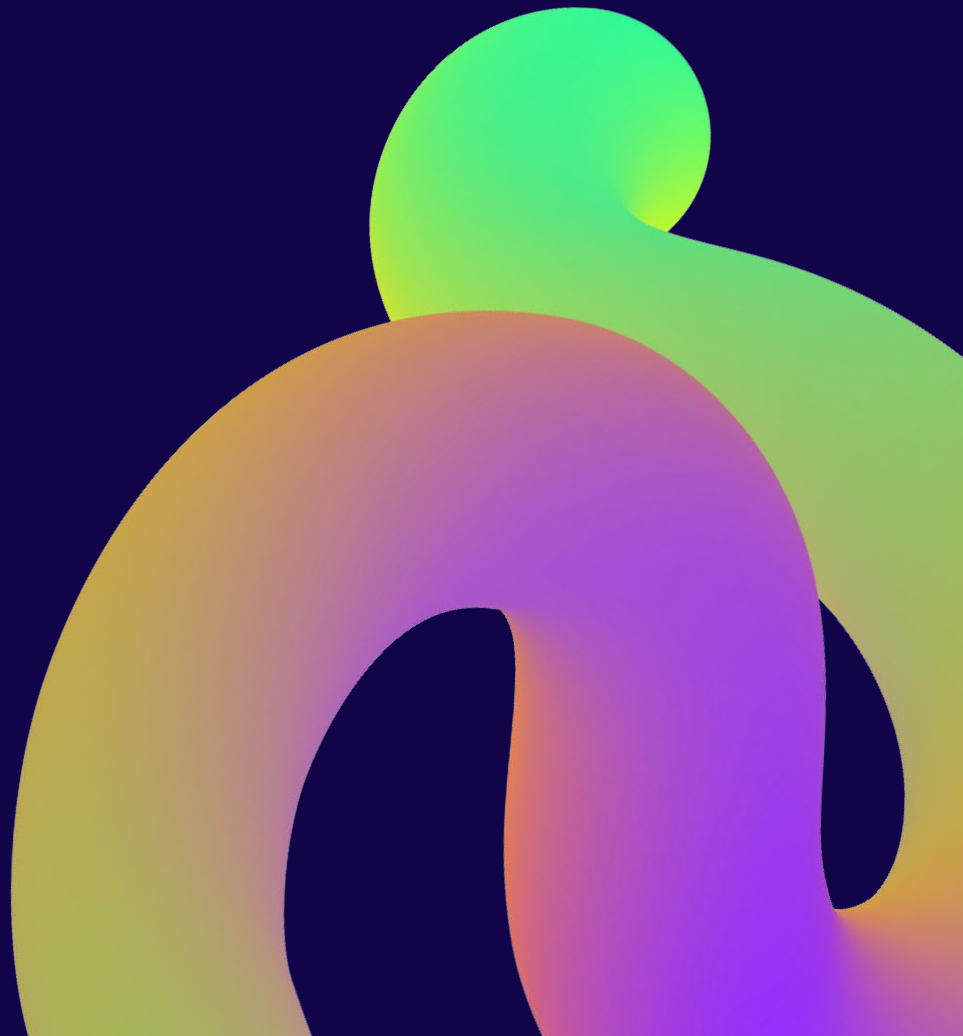
ON TOUR | NEW YORK

PASS

# Learn How to Build Workflow-Driven Database Provisioning in Azure DevOps




**Tonie Huizer**

Consultant

promicro

# Tonie
# Huizer

He/him

Freelance
DevOps consultant

linkedin.com/in/toniehuizer
github.com/promicroNL/events
www.promicro.nl

promicro

SeaQL.nl
Data, DevOps & SQL @ Sea!

redgate
COMMUNITY
AMBASSADOR

# Deep Dive Session Agenda / Timing

The agenda/timing for Deep Dive sessions on Day 1 is as follows:

| START TIME | END TIME | STATUS |
|---|---|---|
| 09:00 AM | 10:30 AM | Deep Dive #1 in session |
| 10:30 AM | 10:45 AM | Refreshment break |
| 10:45 AM | 12:15 PM | Deep Dive #1 in session |
| 12:15 PM | 1:15 PM | Lunch |
| 1:15 PM | 2:45 PM | Deep Dive #2 in session |
| 2:45 PM | 3:00 PM | Refreshment break |
| 3:00 PM | 4:30 PM | Deep Dive #2 in session |
| 5:00 PM | - | END OF DAY |

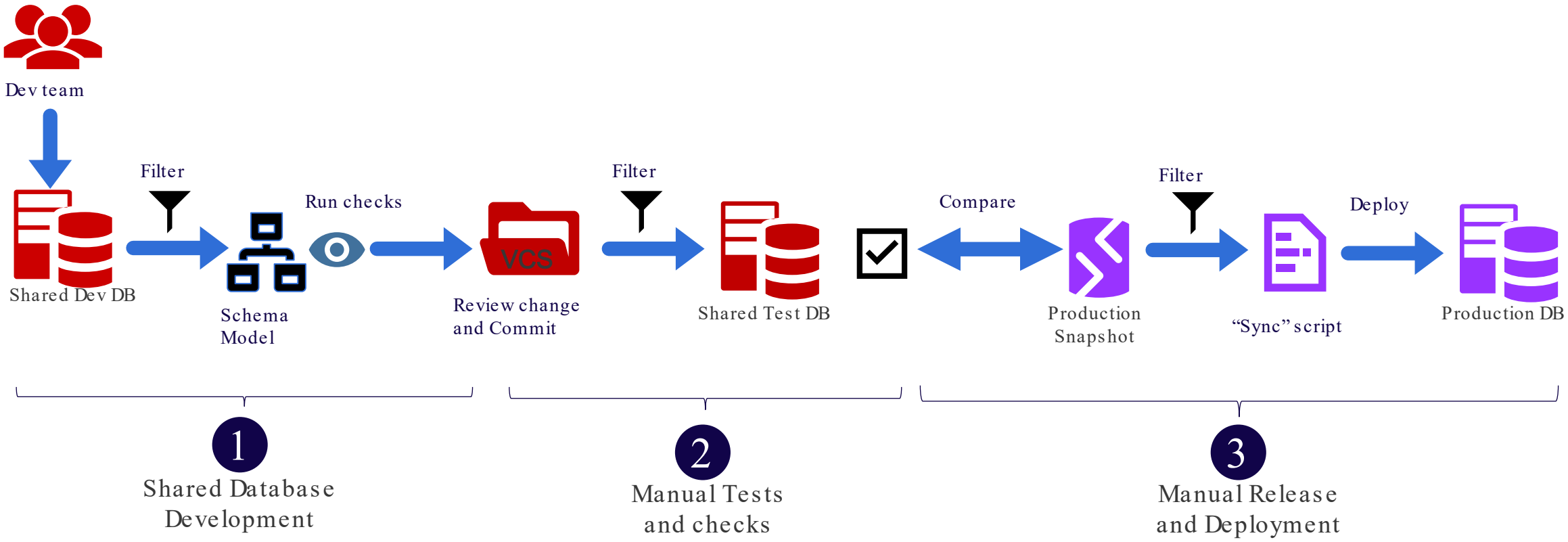
Please note that one Deep Dive session is a half-day.

# What to expect of today's session

| Time | Segment |
|---|---|
| 09:00–09:10 | **Welcome** |
| 09:10–09:30 | **Why Our Database DevOps Wasn’t Working (Yet)** |
| 09:30–10:00 | **What we changed: A Workflow-Driven Approach** |
| 10:00–10:30 | **Unattended Automation: Provisioning, PowerShell & YAML** |
| 10:30–10:45 | **Refreshment Break** |
| 10:45–11:15 | **Changing priorities: Database Stashing** |
| 11:15–11:45 | **House keeping: Lifecycle & Cleanup** |
| 11:45–12:15 | **Wrap-up: Take aways & Q&A** |

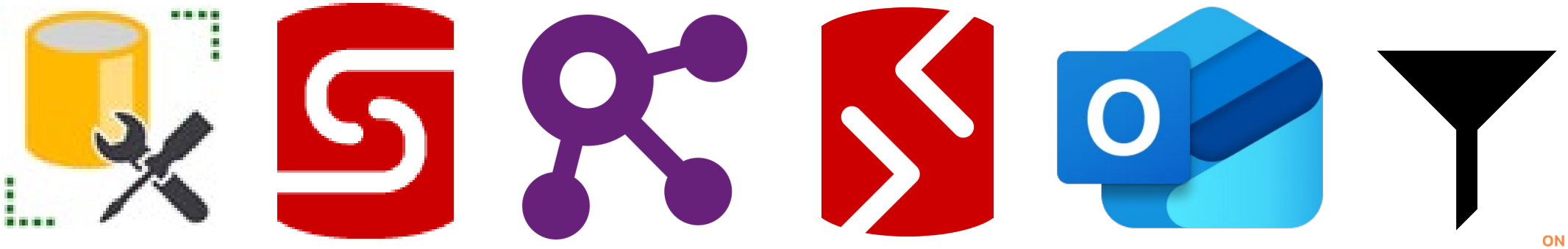
# Why Our Database DevOps Wasn’t Working (Yet)

# Our start: No real Database DevOps

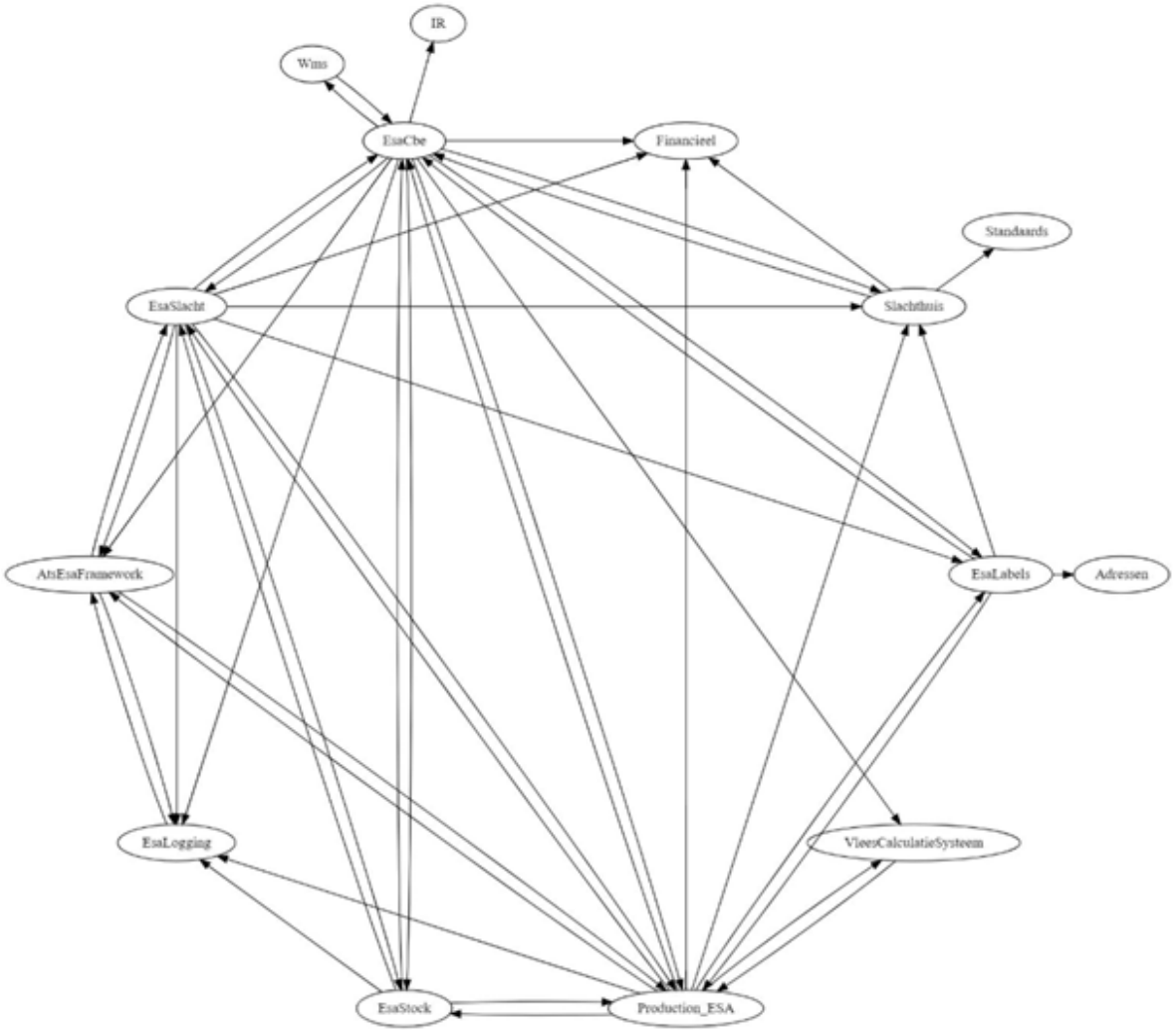


# To sum up

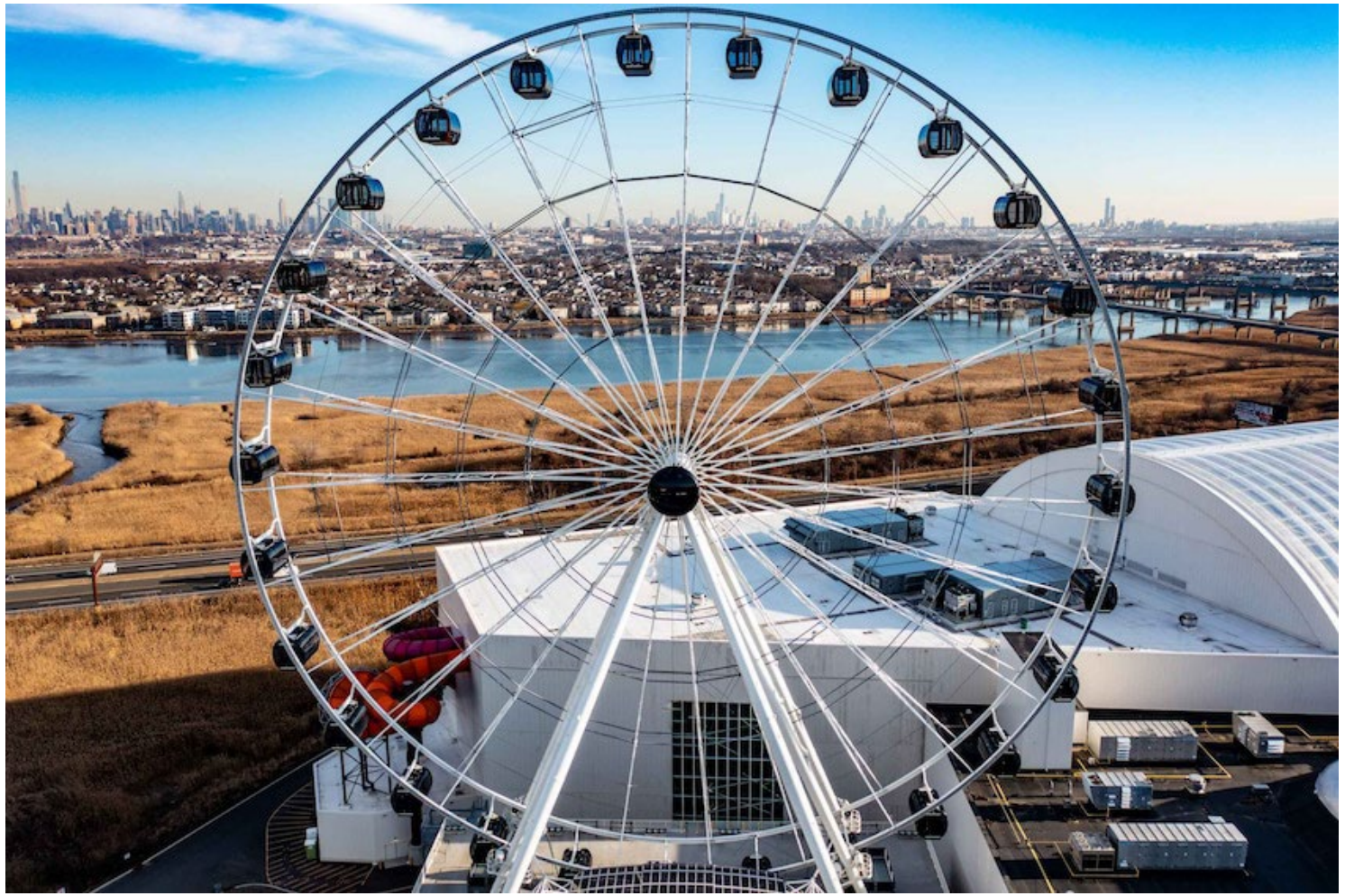
- Shared database
- Source control
- Manual
- Repeated actions
- “DevOps” tools:



# Database DevOps – the challenge

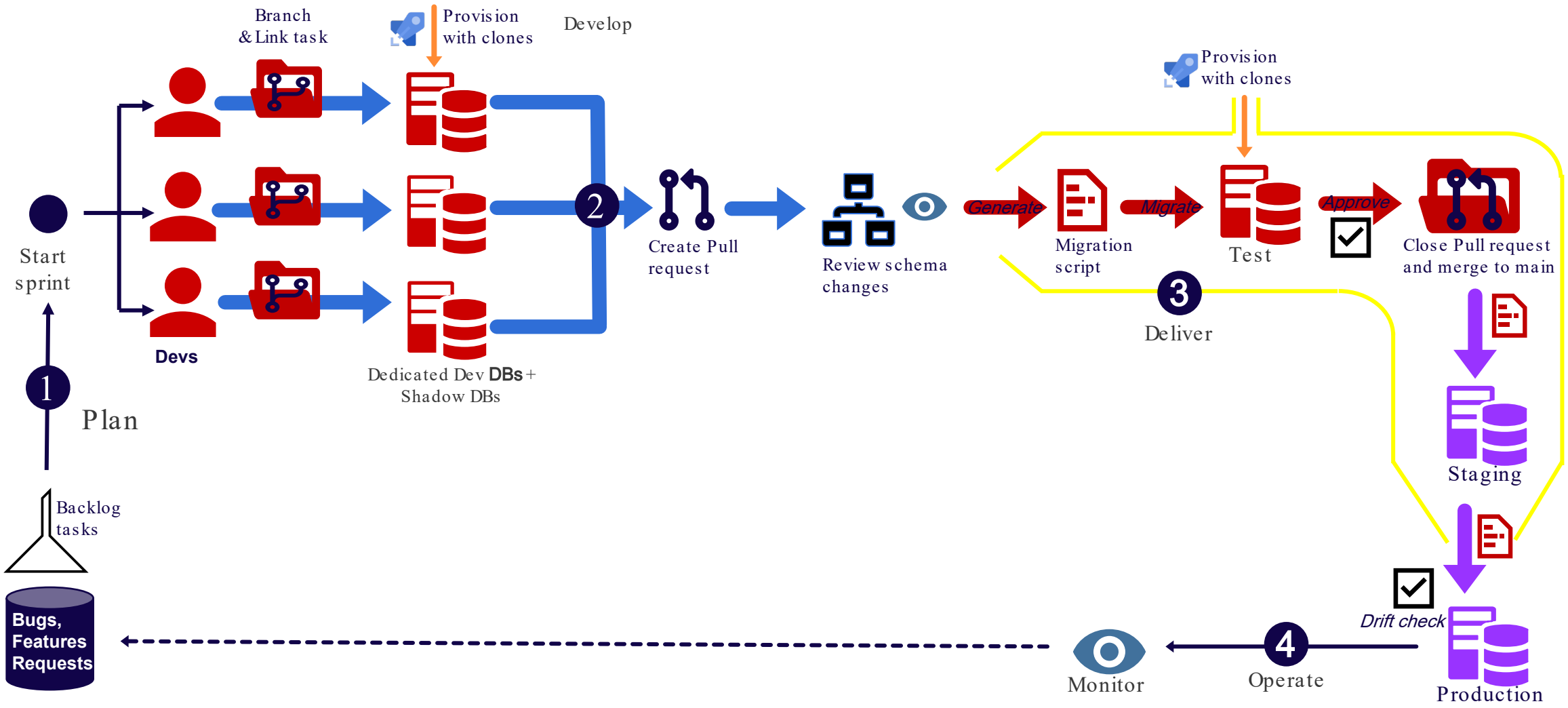


# Database DevOps – the challenge



# Our Solution: Workflow-Driven Provisioning

# The result: Database DevOps



# Demo time

# What did we change?

- Choose a VCS to collaborate and work in parallel
- Introduced dedicated disposable environments
- Turned code into immutable artifacts
- Automate delivery processes with approvals
- Tested in parallel, just like development
- Implement Pull Request Release workflow

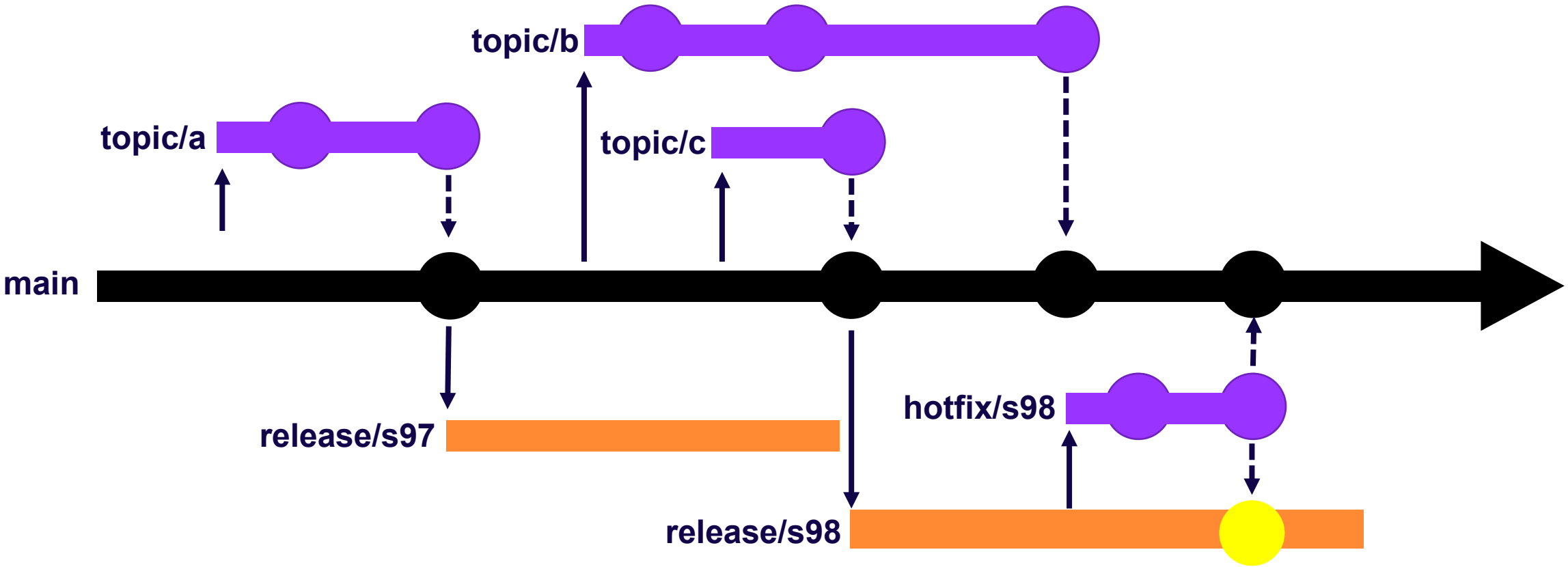
# Switching to git

- Migrate a VCS
- Adopt a branching strategy
- Create a branch naming convention

# Branching strategy

- Release flow
  - Short living topics
  - Main branch always in release state
  - Hotfix the release, cherry pick main

# Release flow in action



# Create a branch naming convention

Topic / hotfix

<branch category> /**<hot fixed release - >bug**<TicketId>-PascalCasingDescription

hotfix/s100-mms12220-FireFighting
topic/mms12345-MyDescription

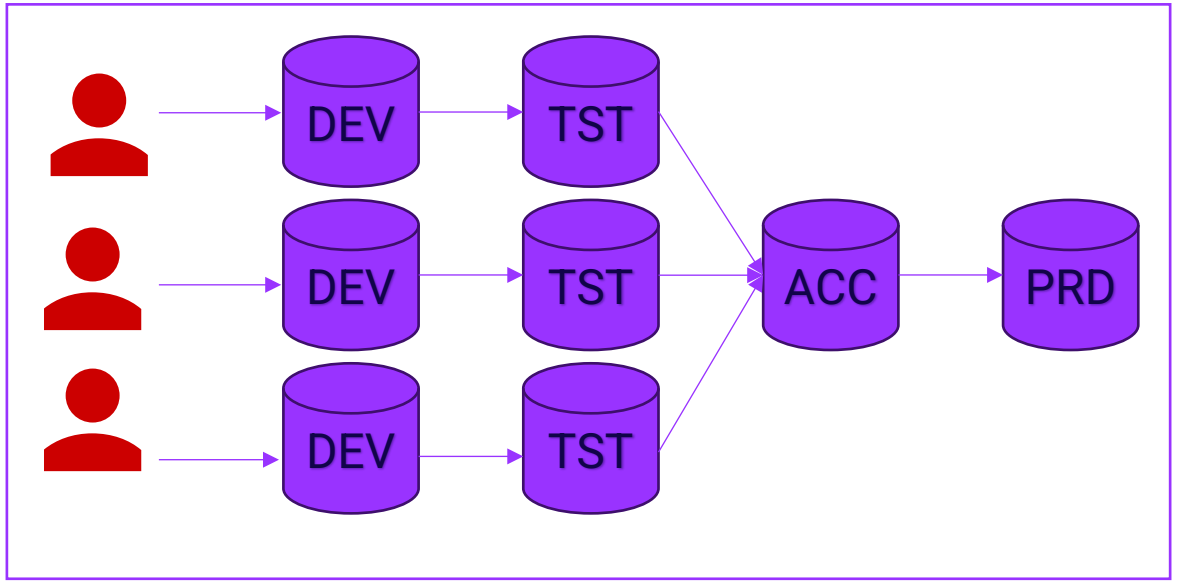
Release branch

<branch category>/ **<unique identification>**

release/s100

# Dedicated environment

- What is it?
- Why do you want it?
- Complex initial setup



# Dedicated disposable example options


| Manual | Code tools | Containers | Clones |
|---|---|---|---|

# Provisioning Should Follow the Work

# A Simpler, Smarter Provisioning Model

**1. Pick up a work item**

From the
Azure DevOps
Board

**2. Create a Git branch**

Linked this work item
→ triggers build
→ provisions a database

**3. Do you work**

For Develop
→ approve release
for test → PR workflow

**4. Dispose environment**

Automatic teardown
For a clean
Dev/test environment

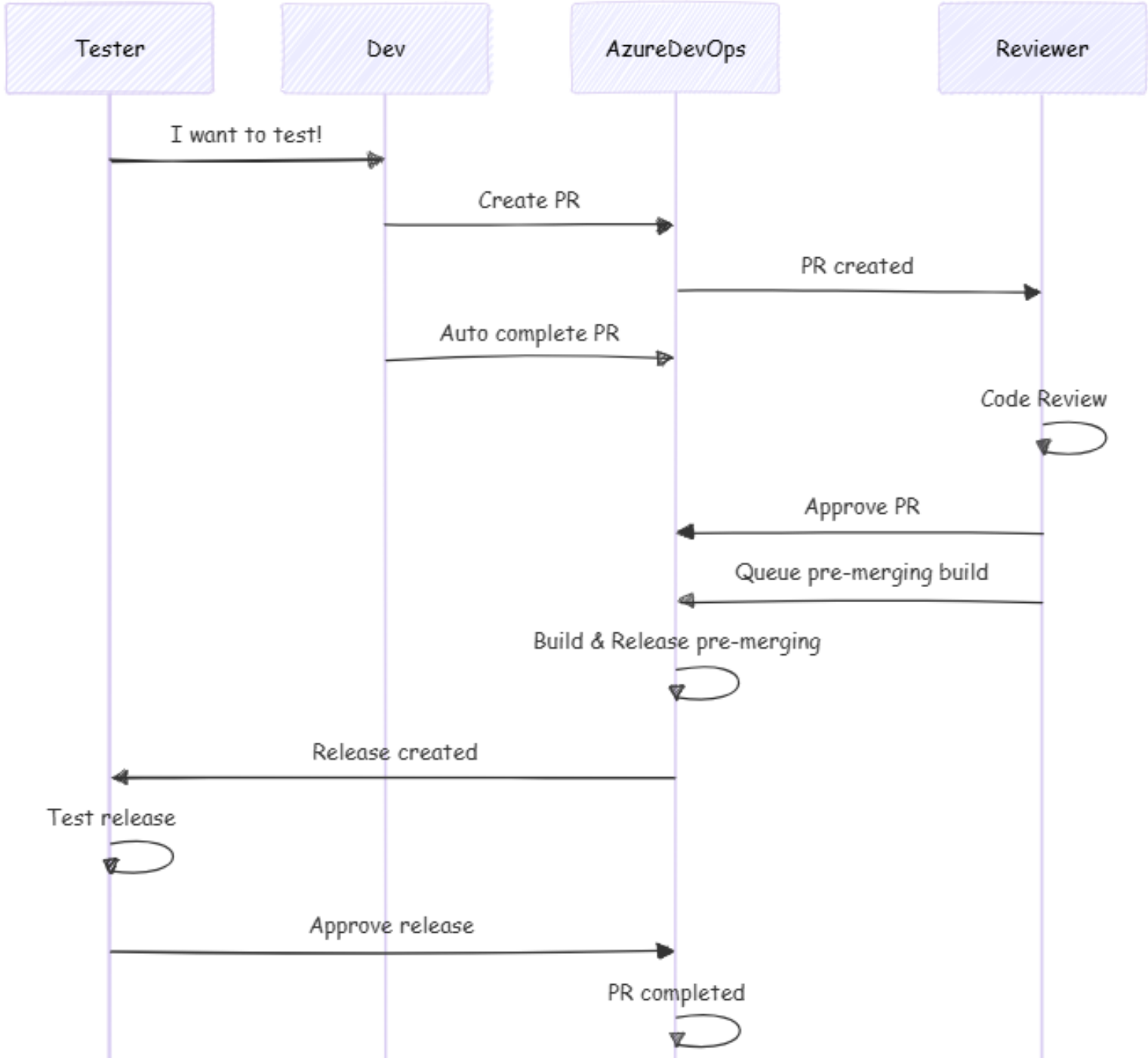
# Automate repetitive tasks

- Use pipelines, but first…
  - Start local to automate
  - Use verbose logging
  - Avoid inline PowerShell

# Demo time

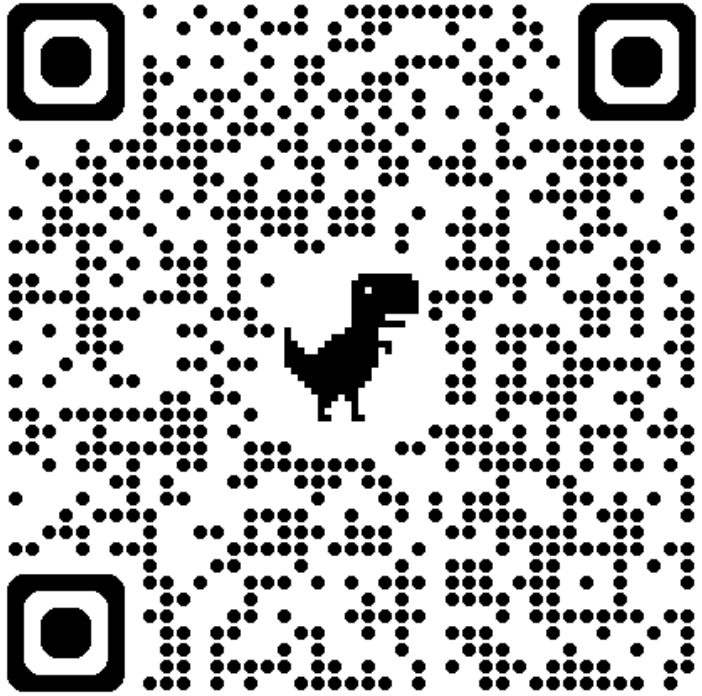
# What did we just see?

- 00-utils.ps1 – Shared Utility Library
- 01-precheck.ps1 – Environment Validation
- 02-create-db.ps1 – Create Database
- 03-apply-sql.ps1 – Apply Migrations
- 04-verify.ps1 – Post Migration Verification


Tester
Dev
AzureDevOps
Reviewer
I want to test!
Create PR
PR created
Auto complete PR
Code Review
Approve PR
Queue pre-merging build
Build & Release pre-merging
Release created
Test release
Approve release
PR completed

# Demo time

# What did we just see?



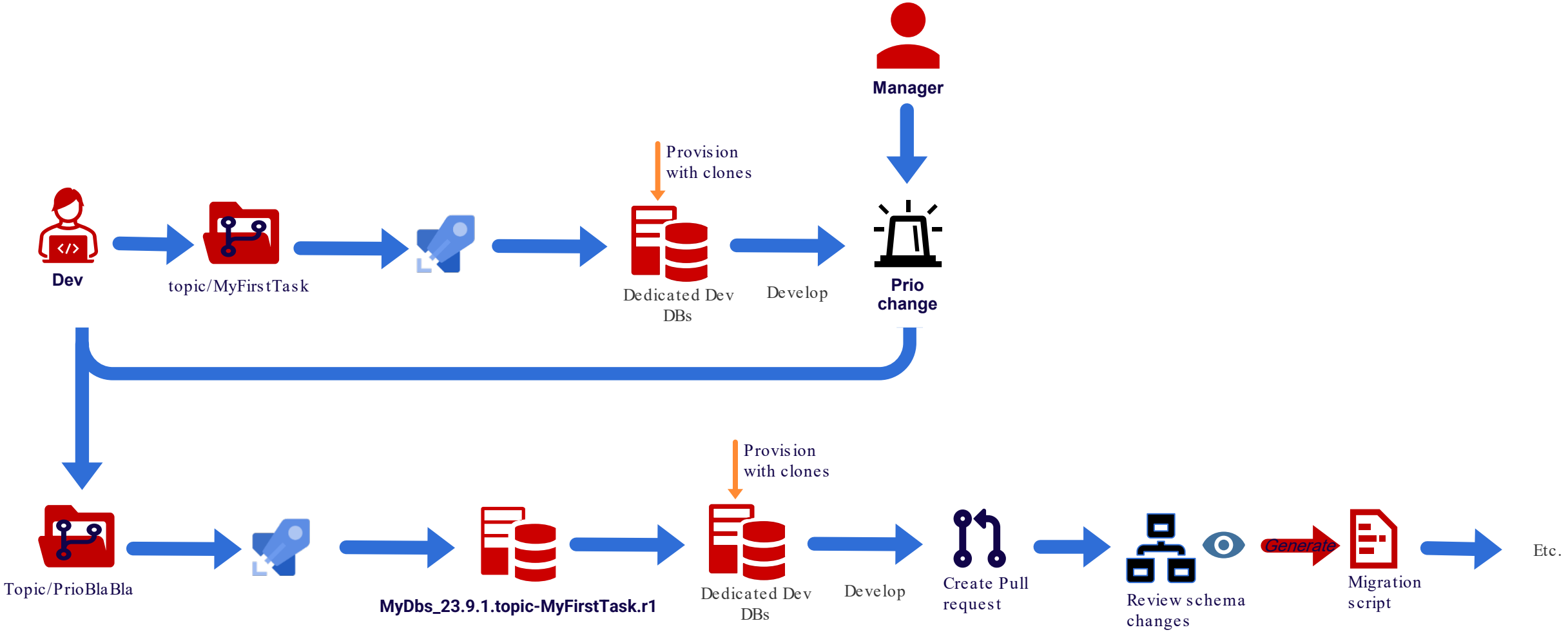
- Configuration of branch policies:
  - PR workflow

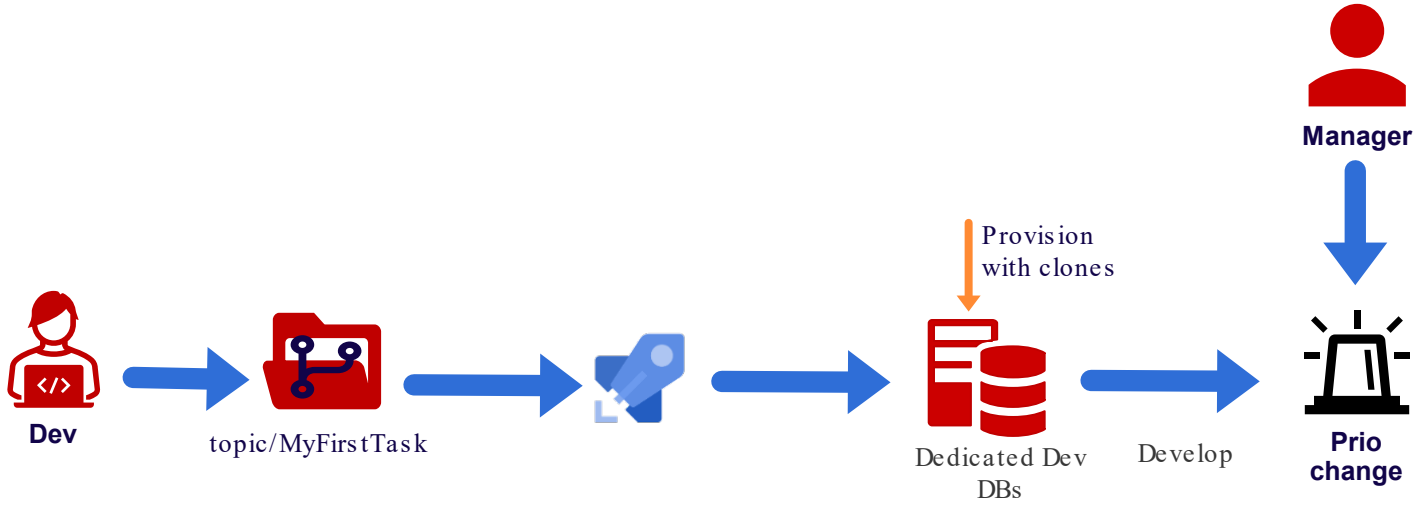
- 06-resolve-approver-env.ps1 – Map Approver Environment
  - Queries Azure DevOps to find who approved
  - Sets Approver and TargetEnvironment for downstream jobs

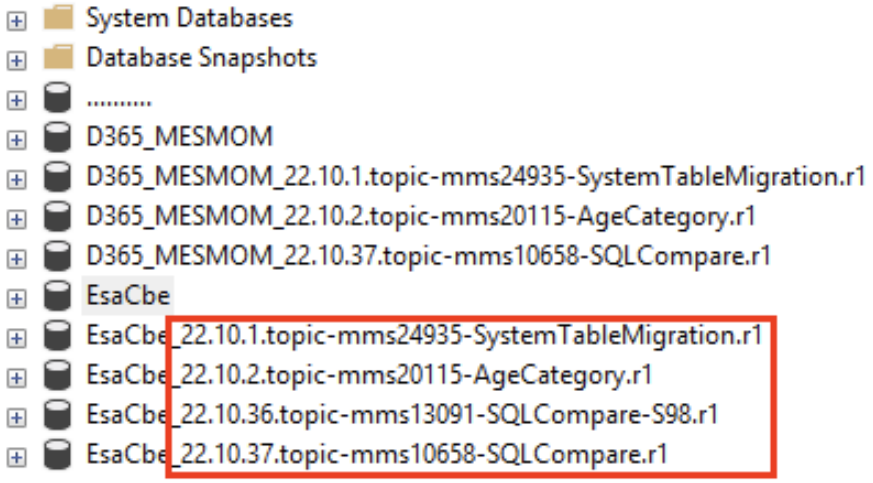
# Database Stashing

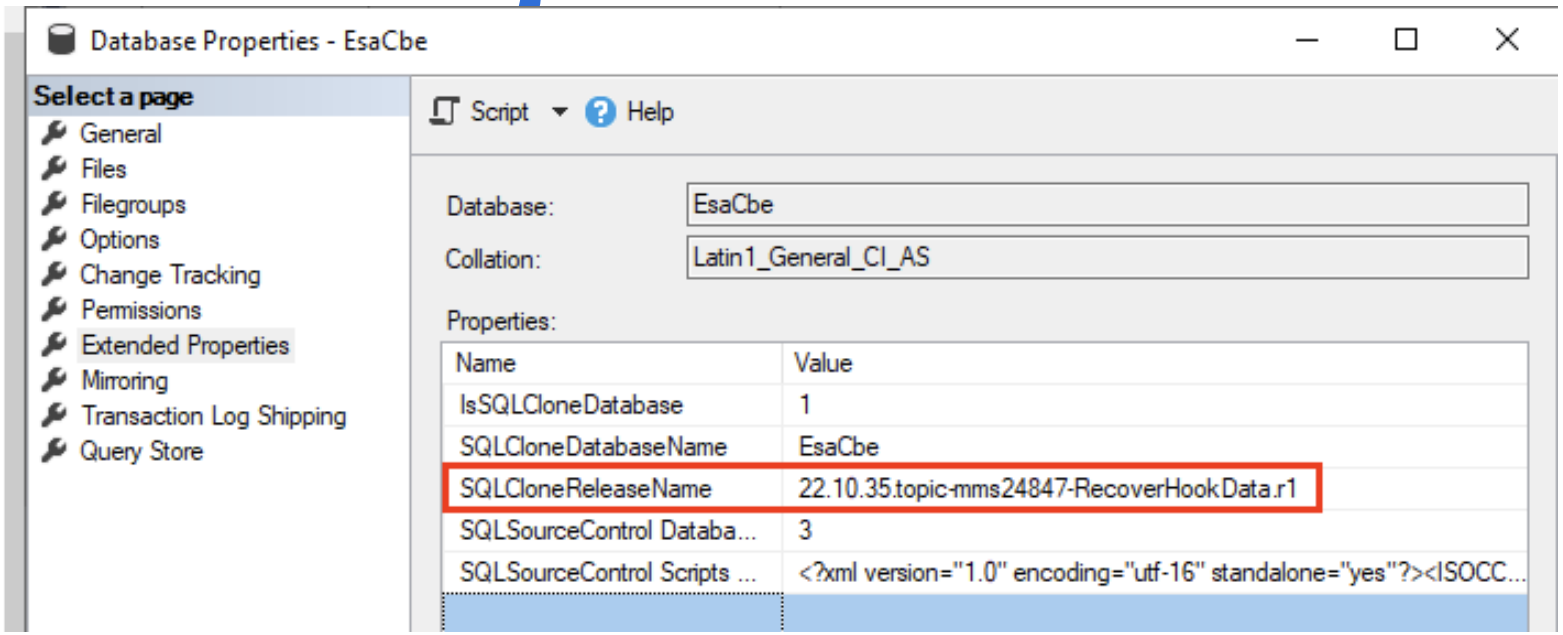
# The concept of database stashing

- Changing priorities
- git stash but then for DBs
- Personal instance / environment
- Through AzDo not on your system


Manager
Provision
with clones
Dev
topic/MyFirstTask
Dedicated Dev
DBs
Develop
Prio
change
Provision
with clones
Topic/PrioBlaBla
MyDbs_23.9.1.topic-MyFirstTask.r1
Dedicated Dev
DBs
Develop
Create Pull
request
Review schema
changes
Generate
Migration
script
Etc.


Manager
Provision
with clones
Dev
topic/MyFirstTask
Dedicated Dev
DBs
Develop
Prio
change


System Databases
Database Snapshots
.....
D365_MESMOM
D365_MESMOM_22.10.1.topic-mms24935-SystemTableMigration.r1
D365_MESMOM_22.10.2.topic-mms20115-AgeCategory.r1
D365_MESMOM_22.10.37.topic-mms10658-SQLCompare.r1
EsaCbe
EsaCbe_22.10.1.topic-mms24935-SystemTableMigration.r1
EsaCbe_22.10.2.topic-mms20115-AgeCategory.r1
EsaCbe_22.10.36.topic-mms13091-SQLCompare-S98.r1
EsaCbe_22.10.37.topic-mms10658-SQLCompare.r1


Database Properties - EsaCbe
Select a page
General
Files
Filegroups
Options
Change Tracking
Permissions
Extended Properties
Mirroring
Transaction Log Shipping
Query Store
Script
Help
Database:
EsaCbe
Collation:
Latin1_General_CI_AS
Properties:
Name
Value
IsSQLCloneDatabase
1
SQLCloneDatabaseName
EsaCbe
SQLCloneReleaseName
22.10.35.topic-mms24847-RecoverHookData.r1
SQLSourceControl Databa...
3
SQLSourceControl Scripts ...
<?xml version="1.0" encoding="utf-16" standalone="yes"?><ISOCC...

# Demo time

# What did we just see?

- 05-stash.ps1 – Stash Database
  - Renames the database with the current build name, preserving it for later inspection or rollback.
  - Only stashes when the existing BuildName extended property matches
- 02-create-db.ps1 – Create Database or Unstash

# Demo time

# Lifecycle & Clean-up

# Housekeeping for Provisioned Environments

- These environments are disposable
  - created per branch or PR
- Automatically deleted when:
  - The build is completed successfully
- Fully automated via scheduled pipelines

# Demo time

# What did we just see?

- 07-cleanup-stash.ps1 – Remove Old Stashed DBs
  - Lists stashed databases (name + build suffix)
  - Checks each build’s status via Azure DevOps
  - Drops any that belong to completed builds to free up resources

# Housekeeping for Provisioned Environments

- Cleanup snapshots / images / etc:
  - Unused images (not linked to any active environment)
  - Outdated images
- Regular refresh jobs:
  - Run on weekends
  - Capture fresh production snapshots for the next workweek

# Wrap-up:
# Take aways & Q&A

# Take aways

- No work item, no environment
- Use dedicated over shared
- Easily switch out the heart; the provisioning part
- Use scripts but only those that live in git
- Automation rocks…
  - but without logging it's too much black box

# Questions?

# Thank you

I appreciate the time you spent with me.

Please reach out if you have any questions!


## Tonie Huizer

linkedin.com/in/toniehuizer

github.com/promicroNL/events

www.promicro.nl