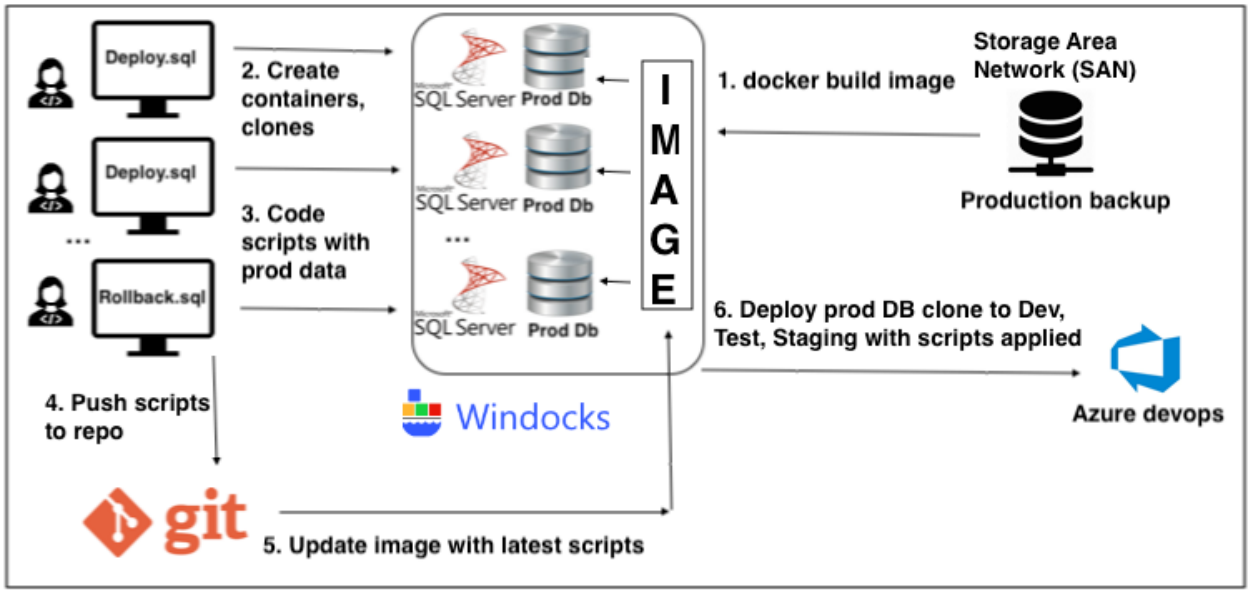
Devops best practices for SQL Server script development

Summary

- Develop not just **deployment** scripts but also **rollback** scripts to roll back releases if needed
- Each script developer uses **his own writeable production database clone** for development
- Write scripts **like you write application code**. Push script code often to a Git or Azure repo
- Each script developer can create his individual database environment to have production data with the latest upgrade script from the glt repo applied
- When scripts are pushed to the Git repo, deploy the production database (with the latest deployment scripts applied)to the devops pipeline DEV, TEST, STAGING environments

Steps



Step 1: A database administrator logs in to the Windocks machine and builds an image PRODIMAGE-Date with the production database using the web application or

docker build -t PRODIMAGE-Date path\to\dockerfile.

This image is used to create containers with the production data clone and the git repo with upgrade scripts pulled into the container. The docker file looks like:

```
FROM mssql-2016
SETUPCLONING FULL customers C:\windocks\dbbackups\customerdatafull.bak
ENV USE_DOCKERFILE_TO_CREATE_CONTAINER=1
RUN "C:\Program Files\Git\cmd\git.exe" clone https://github.com/WinDocks/git-db-scripts-runtime.git scripts
```

Use a production database backup or a SAN to provide the production database(s) for cloning.

Step 2: Each developer creates her own SQL Server container with a clone of the production database using the Windocks web application

Step 3: Each developer writes and tests deployment and rollback scripts in his own container to modify or create schema, stored procedures, and data. All DB changes are in scripts

Step 4: Eventually, the developers arrive at a stable version of upgrade scripts. Developers push the scripts to a Git or Azure repo. The git push triggers a devops pipeline

Step 5: The latest git repo scripts are automatically available to the image and fresh containers created from the image

Step 6: A devops pipeline step deploys a DEV database environment with production data and the fresh upgrade scripts applied to the production data. This can be done in one of two ways:

a) Call the REST API on the Windocks server with curl or Powershell Invoke-RestMethod
See the Windocks rest api doc.
b) Use psexec to run the following commands on the Windocks server
psexec \\remote -u remote\administrator -p adminpass create.bat

where create.bat contains:

docker create -e RUN="scripts\upgradescript1.sql" PRODIMAGE-Date
// Parse to get container ID
docker start <containerId>

The next devops pipeline step runs tests on the DB to test the upgrade scripts. If they pass, then the DEV database environment is ready. The devops pipeline similarly deploys TEST and STAGING database environments with production data and the fresh upgrade scripts applied.

Developers continue debugging and modifying upgrade scripts in their individual SQL Server containers with production data clones. Each time they check in updated scripts, the devops pipeline runs again and deploys fresh DEV, TEST, and STAGING database environments with the latest upgrade scripts applied to production data.

.NET pipeline steps can use these DEV, TEST, STAGING database environments with the upgrade scripts applied.