Xpirit

Think ahead. Act now.

Handling Data in Containers & Testing

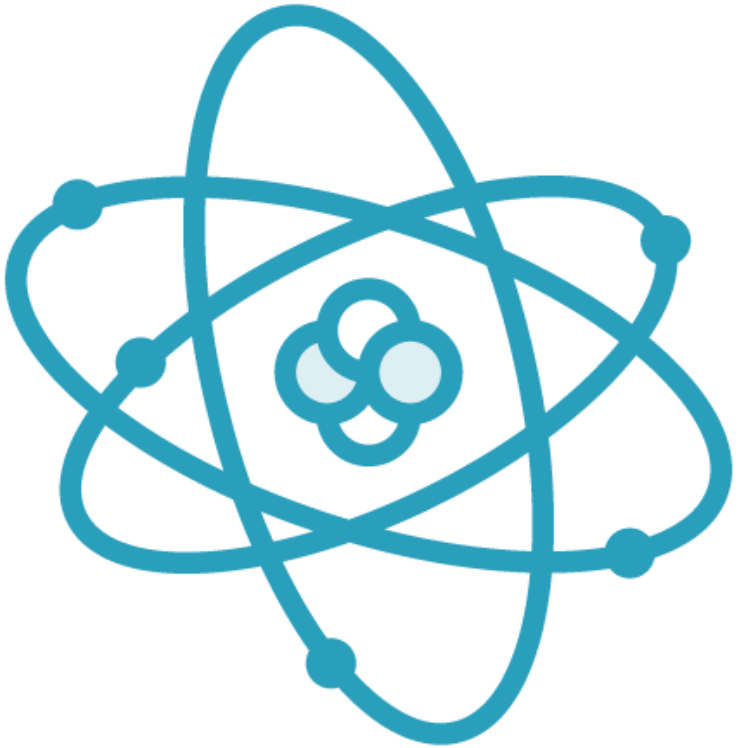
Marcel de Vries

Overview

- Handling Data in Containers
- Setting up Volume Mappings to Manage State
- Setting up Data Volumes
- Running SQL Server in a Container
- Leveraging Immutability for Testing

Handling Data in Containers

Container Images are Immutable



When I save state in the container, next time I start the state is gone

How do I keep the state?

Commit the state as a new image

Save the state to the host

Setting up Volume Mappings to Manage State

Volume Mappings

Maps a host folder to a container folder

Host folder can contain data

On Windows the container folder can not have data

On Linux the data is copied to the host

Volume Mappings

Mapping based on logical name

Defined on the host

Provides host agnostic mapping

Volume map chaining

A volume can point to another container volume map

Demo

Using the Volume Mapping Commands and Setting up the Data Volumes to Handle the Data

Running SQL Server in a Container

SQL Server Container

Why?

Run against multiple versions of SQL server
E.g. Linux & Windows, Express, Full, etc

Do you want to keep the data?
Yes-> create volume mapping
No-> Keep files on container disk

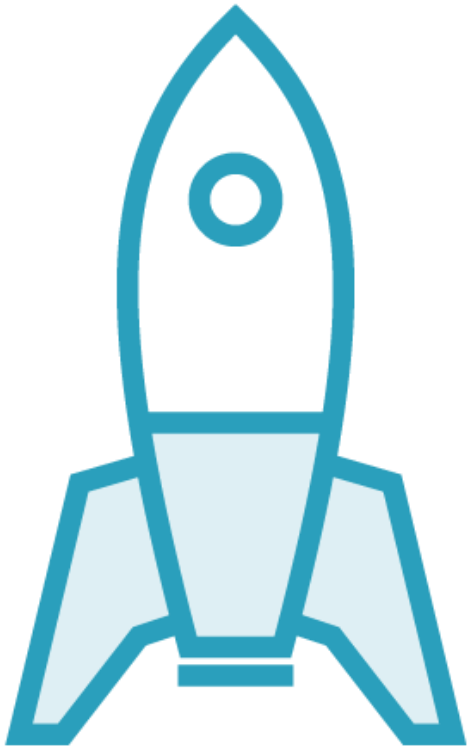
Demo

Creating and Running SQL Server Images

Xpirit
Think ahead. Act now.

Leveraging Immutability for Testing

Leverage Immutability



Every new container instance is always clean

Great way to set up your integration tests

E.g.: Build containers with multiple schema's to validate update of application

Docker Compose

When using Docker-compose you can name the container instance

Reachable on the network

Easy way to switch/swap containers with same name but other
Version of SQL
Version of your database schema

Demo

Integration Test Using a SQL Container That Resets After the Tests

Summary

- Handling Data in Containers
- Setting up Volume Mappings to Manage State
- Setting up Data Volumes
- Running SQL Server in a Container
- Leveraging Immutability for Testing