SQL Server unit testing con tSQLt Docker e GitHub Actions

Sergio Govoni

slide and demo: https://bit.ly/sqlstart2024

Microsoft® Most Valuable Professional

SQL START! 2024

Speaker bio

Sergio Govoni

Microsoft® MVP Most Valuable Professional

twitter.com/segovoni

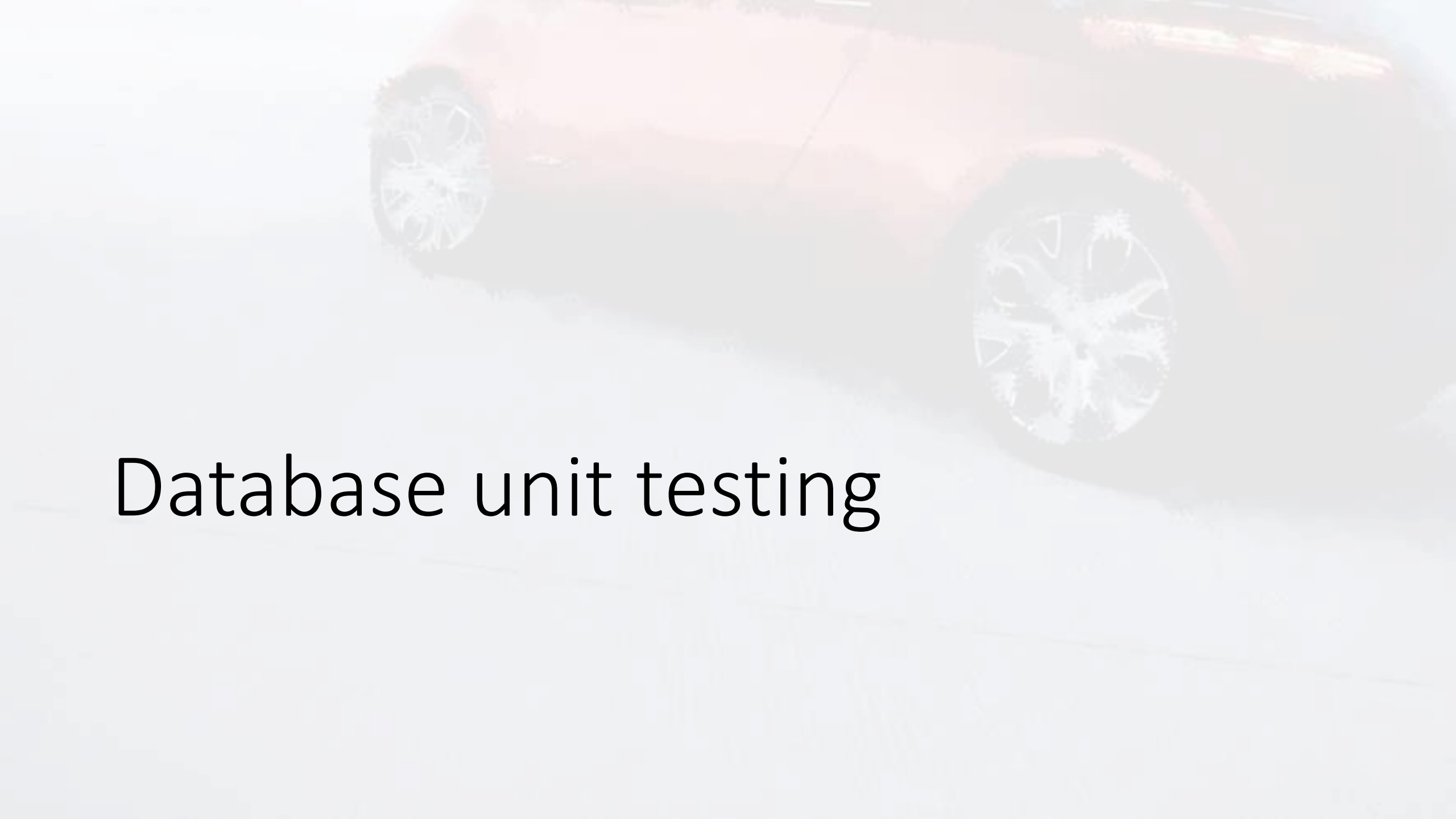
github.com/segovoni

linkedin.com/in/sgovoni

Sponsors & Organizers

Agenda

- Introduction to
  - Unit testing
  - tSQLt
  - Docker
  - GitHub Actions
- Case history
- Configure and run unit tests for TSQL code in GitHub



Database unit testing

Types of database unit tests

- Structural Testing
- Functional Testing
- Non-functional Testing

What is a unit test?

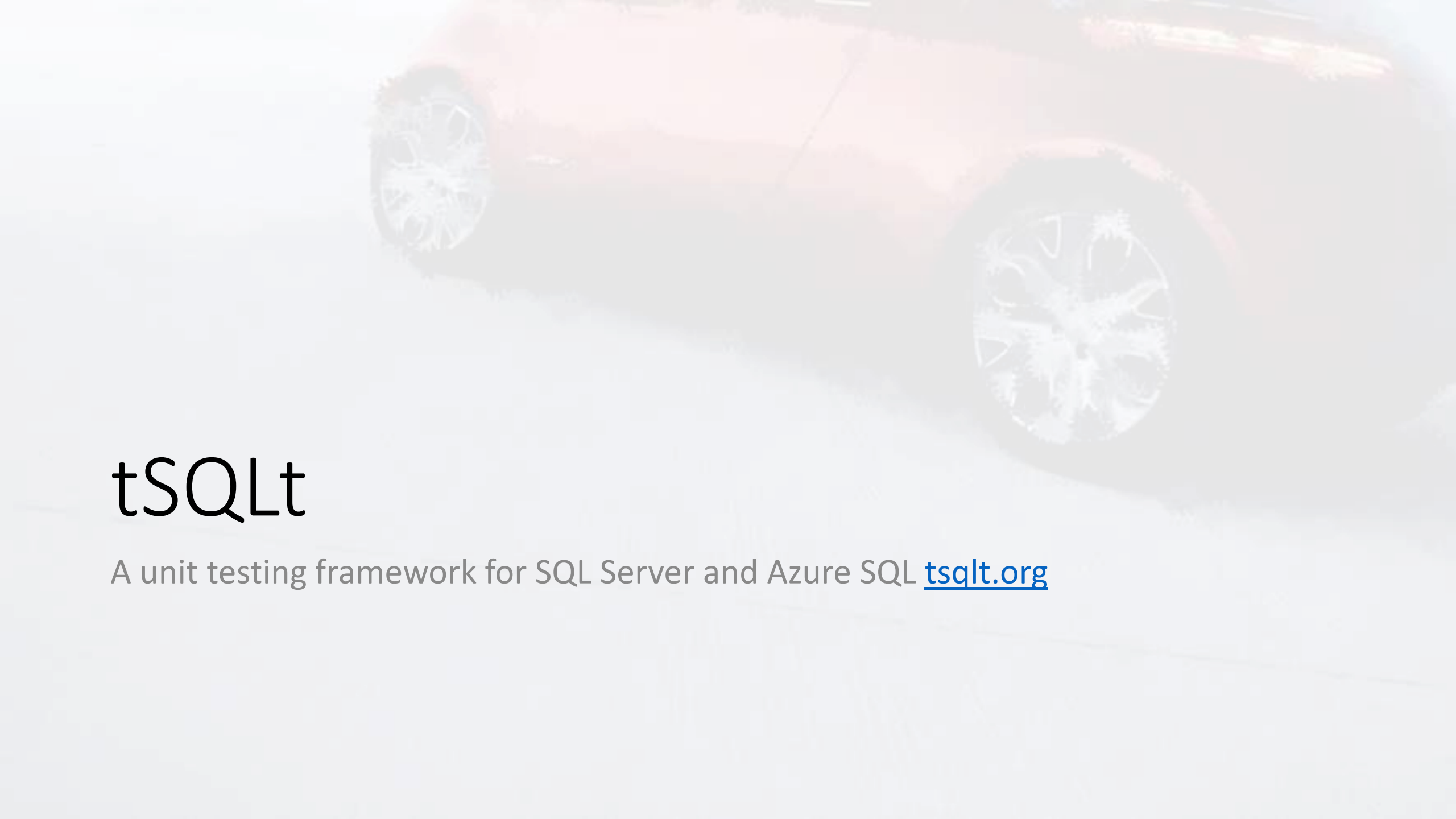
- Unit testing is a software testing level that aims to test a discrete piece of code. The “unit” refers to the **smallest** piece of code that can be tested **separately**
- Unit test
  - Must be repeatable
  - Isolates the code under test from the rest of the code
  - Doesn’t test how the unit interacts with other units

What is a unit test?

- Unit test must test one question at a time and that question should reflect a requirement for our code
- Unit testing is about confirming that all the individual parts work, not that they work together
- Unit tests is usually written by the Development Team

What does a unit test give me?

- Unit tests convey safety
- Unit tests provide documentation of the software requirements
- Unit tests are preparatory to the design phase (TDD methodology), they force you to think how to organize properly
- Unit tests simplify the error checking process



tSQLt

A unit testing framework for SQL Server and Azure SQL tsqlt.org

Introduction to tSQLt

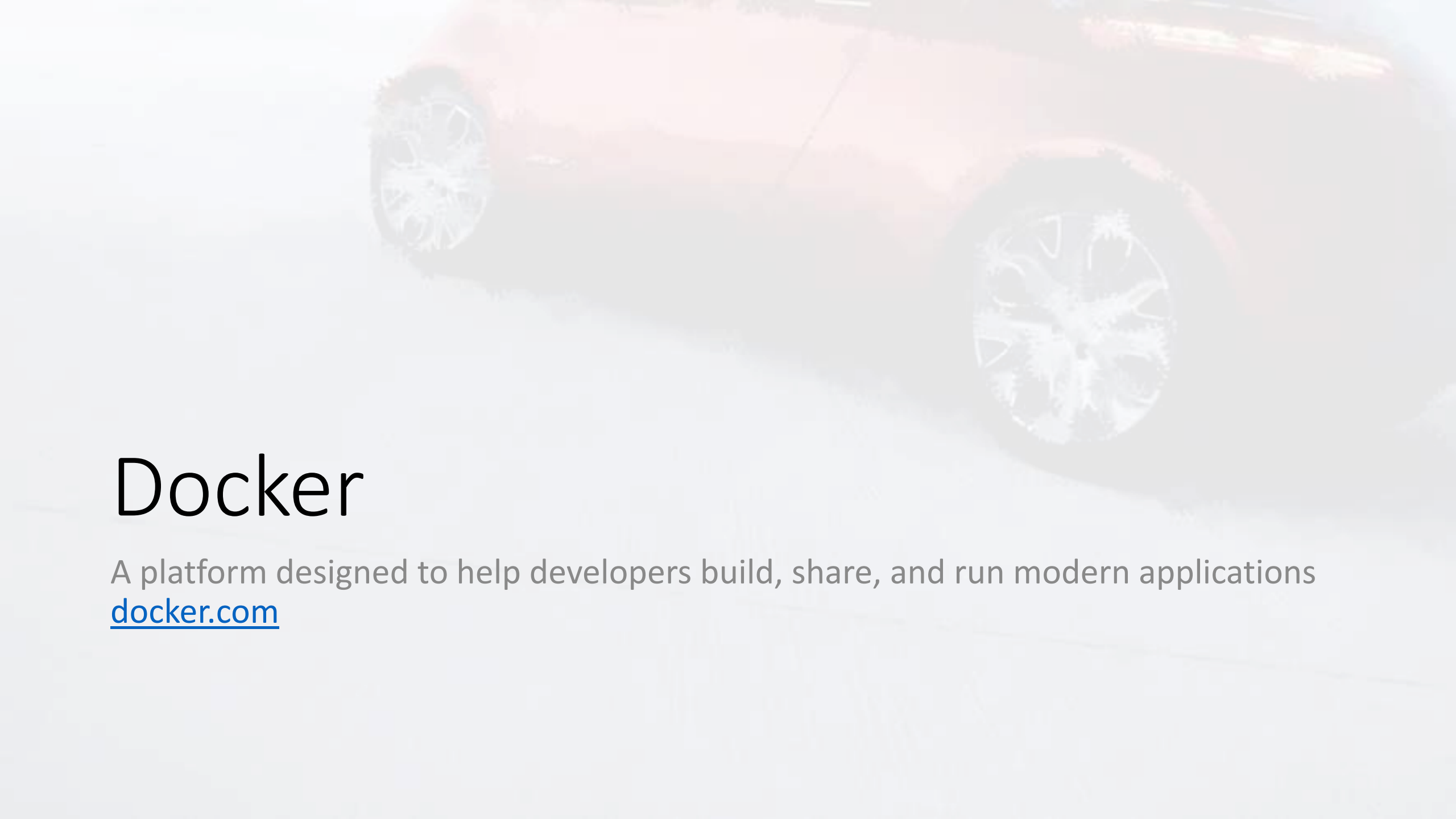
- The framework tSQLt was developed by Sebastian Meine and Dennis Lloyd, it's an open-source **framework** for implementing **unit tests** in T-SQL for SQL Server and Azure SQL Database
- It works with
  - All editions of SQL Server starting from SQL Server 2005 SP2
  - Azure SQL Database
- It requires SQL CLR enabled

Benefits of using tSQLt

- Unit tests will be written in T-SQL, you don’t need to learn new programming language
- Data manipulation will be rolled back at the end of the test, so you don’t need any data cleanup
- Mock objects are supported
- tSQLt can be integrated into SSDT projects or 3rd party tools
- Tests can be grouped within a single schema
- You can use a setup routine for a group of tests or class
- The output can be in plain text or XML

tSQLt Setup

- Download tSQLt scripts from tsqlt.org/downloads
- Enable CLR at the SQL Server instance level
- In each development database you want to install tSQLt
  - Enable TRUSTWORTHY property
  - Execute tSQLt.class.sql



Docker

A platform designed to help developers build, share, and run modern applications

docker.com

Introduction to Docker

- Docker is one of the most popular systems for running applications in isolable, minimal and easily deployable environments called **containers**
- Since SQL Server 2017, the SQL Server Engine can run in a Docker container
- A typical usage of running SQL Server in a Docker container concerns the **automation** of software **tests**

GitHub Actions

Introduction to GitHub Actions

- GitHub Actions is a continuous integration and continuous delivery (CI/CD) platform that allows you to automate your build, test, and deployment pipeline
- You can create **workflows** that build and test every pull request to your repository
- GitHub provides Linux, Windows, and macOS virtual machines to run your workflows

Case history

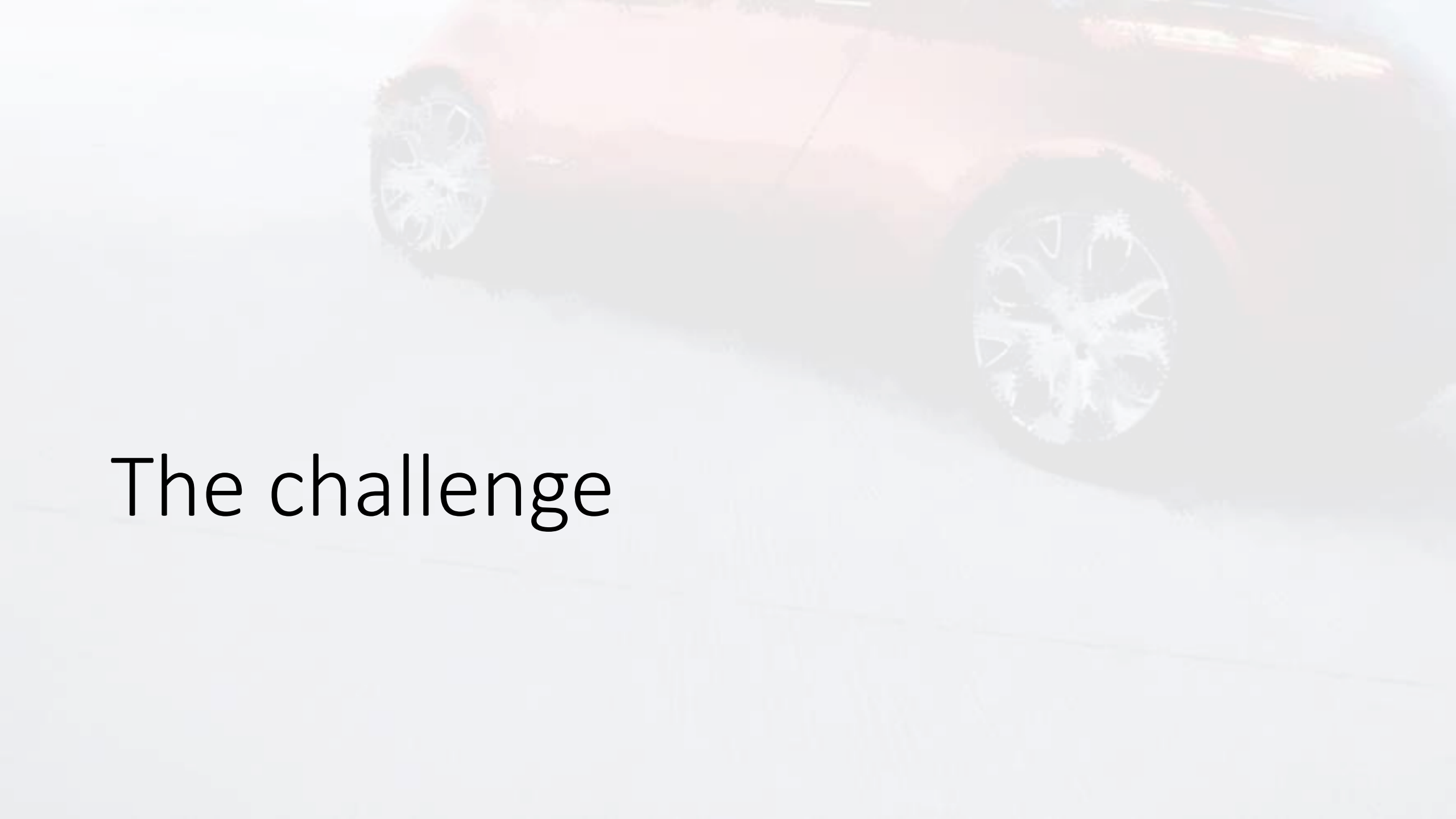
Case history

- The Company Adventure Works LTD wishes to always have a warehouse stock of no less than 10 units for each product
- You developed a Trigger to prevent the insertion of new products with values less than 10 as a “safety stock”
- To make our trigger simple, it will only respond to the OnInsert event, for INSERT statements
- The creation of new purchase orders and production orders are based on the safety stock level

Starting point

The implementation of the trigger and related unit tests has been done, all files are ready in your repository!

Let’s start the challenge!



The challenge

The challenge

- The challenge is to **automate** the execution of the test cases at each commit on the main branch of the repository
- GitHub Actions is our CI/CD platform
  - It supports the use of Docker containers
  - It is intimately integrated into GitHub, the source control for our TSQL code
  - There is a GitHub Action to install tSQLt

Understand and manage GitHub workflow

- A workflow is a configurable automated process that will run one or more jobs
- Workflows are defined with a YAML file stored in the same repository which holds the source code
- The workflows will be triggered when an event occurs in the repository, but it can also be activated manually or according to a defined schedule

DEMO

Resources

- SQL Server unit testing with tSQLt, Docker, and GitHub Actions
  - https://github.com/microsoft/sql-server-samples/tree/master/samples/containers/unit-testing/tsqlt-docker
- Unit testing
  - What it is and why it is important for T-SQL code!
    - https://medium.com/@segovoni/unit-testing-what-it-is-and-why-it-is-important-for-t-sql-code-7e9df7ca8bfe
  - The tSQLt framework and the execution of a test!
    - https://segovoni.medium.com/unit-testing-the-tsqlt-framework-and-the-execution-of-a-test-e4d135c3e343
  - How to write your first unit test for T-SQL code
    - https://segovoni.medium.com/unit-testing-how-to-write-your-first-unit-test-for-t-sql-code-3bc1533acbbc
- GitHub repository used during the session
  - https://bit.ly/3KKGMv9

Summary

- Unit tests are not just meant to verify that requirements have been met once, prior to release
- The real game changer is represented by the possibility of repeating the checks during the development
- The repeatability of the tests provides the ability to automate them, an essential condition for integrating automatic tests within a Continuous Integration platform

Thanks

Questions?

 github.com/segovoni

 twitter.com/segovoni

 linkedin.com/in/sgovoni