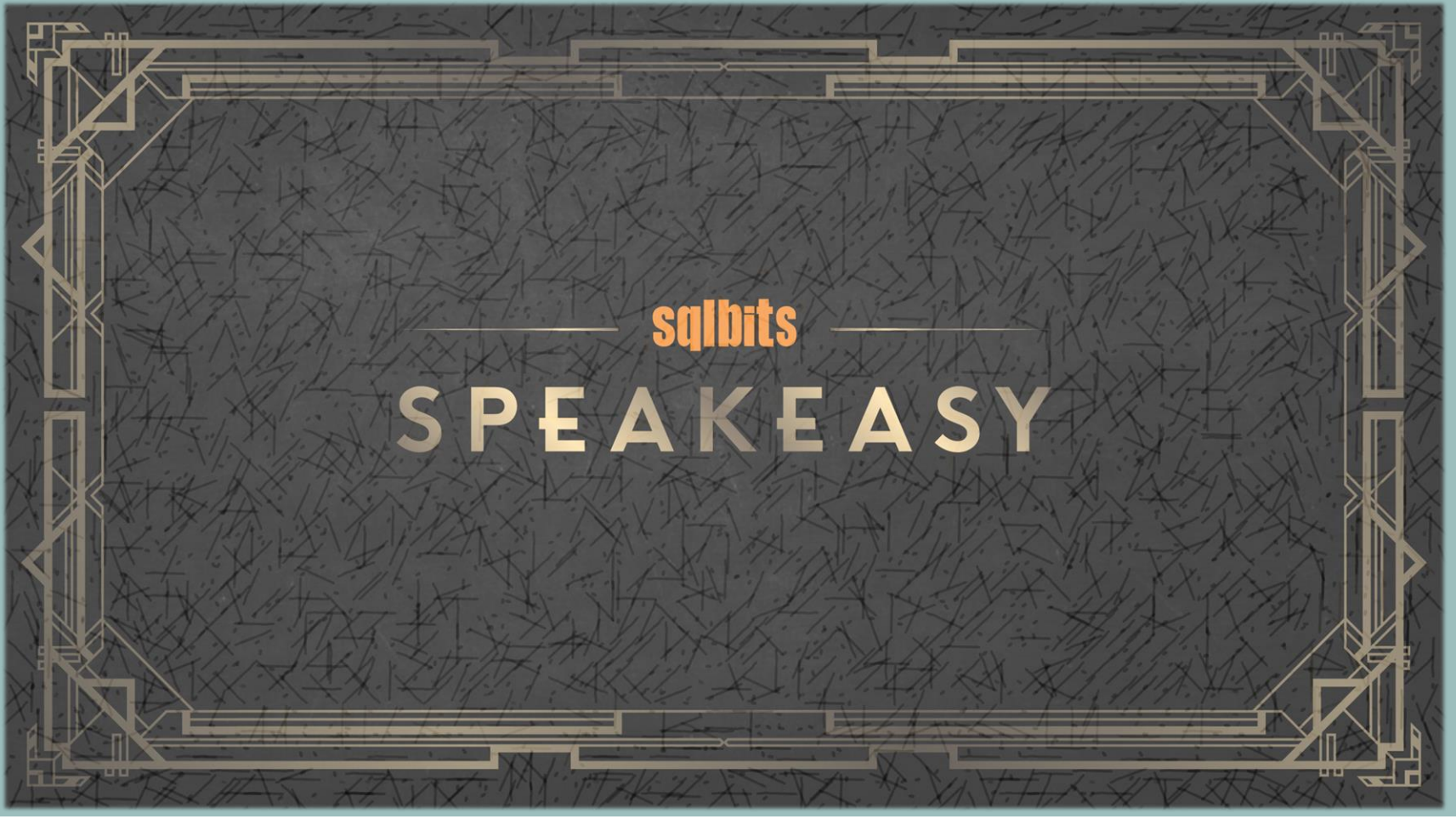


ADDING UNIT TESTS WITH TSQLT TO THE DATABASE DEPLOYMENT PIPELINE

Eduardo Piairo
@EdPiairo
#sqlbits

ADDING UNIT TESTS WITH TSQLT TO THE DATABASE DEPLOYMENT PIPELINE

Eduardo Piairo
@EdPiairo
#sqlbits

ABOUT ME

Adding unit tests with tSQLt to the database deployment pipeline

Eduardo Piairo

DevOps Coach @ Natixis | DevOps Porto Co-Founder |Friend of Redgate

NATIXIS

DEVOPS PORTO

FRIEND OF redgate 2019

@EdPiairo

eduardopiairo@gmail.com

https://www.eduardopiairo.com

https://pt.linkedin.com/in/eduardopiairo

TSQLT

Adding unit tests with tSQLt to the database deployment pipeline

- Database unit testing framework for Microsoft SQL Server
  - Allow to write T-SQL code as tests
  - Tests are automatically run within transactions
  - Provides a way to isolate code and tables using mocking
  - Output can be plain text or XML

TSQLT INSTALL

Adding unit tests with tSQLt to the database deployment pipeline

- **tSQLt.class.sql**
  - CLR
    - clr enabled
    - clr strict security
  - Should be installed in the development database

WHAT'S A TSQLT TEST?

Adding unit tests with tSQLt to the database deployment pipeline

- Stored Procedure
  - Starts with the word **test**
  - Must be in a schema that contains the extended property tSQLt.TestClass = 1
    - tSQLt.NewTestClass
  - Each test is wrapped in a transaction
    - Modifications are rolled back and the results saved

WHAT'S A TSQLT TEST?

Adding unit tests with tSQLt to the database deployment pipeline

- Benefits
  - Business requirements documentation
  - Code refactoring
  - Isolation - unrelated changes do not affect other parts of the system
  - Help structure code into distinct components – keep it small

TSQLT CREATE

Adding unit tests with tSQLt to the database deployment pipeline

```
EXEC tSQLt.NewTestClass my_new_test_class';
GO

CREATE PROCEDURE my_new_test_class.[test something important]
AS
BEGIN
-------Assemble
    --This section is for code that sets up the environment

-------Act
    --  Execute the code under test like a stored procedure, function or view
    --  and capture the results in variables or tables.

-------Assert
    --  Compare the expected and actual
END;
```

TSQLT RUN

Adding unit tests with tSQLt to the database deployment pipeline

- **tSQLt.RunAll**
  - Execute all the tests
- **tSQLt.Run '[your_test_class].[your_test]**
  - Execute a specific test
- **tSQLt.Run '[your_test_class]'**
  - Execute a specific test class

WHAT CAN BE TESTED?

Adding unit tests with tSQLt to the database deployment pipeline

- Stored Procedures
- Functions
- Views
- Tables
- Tables constrains that are critical

TESTING (WHITOUT DATA)

Adding unit tests with tSQLt to the database deployment pipeline

- Unit tests is about testing code
  - You do not need a database full of data, you need the opposite
  - Makes creating unit tests easy
- Data setup
  - Only the necessary data for making the test work is needed
- Mocking
  - tSQLt.FakeTable
  - tSQLt.FakeFunction
  - tSQLt.SpyProcedure

ISOLATING OBJECTS

Adding unit tests with tSQLt to the database deployment pipeline

- tSQLt.FakeTable
  - Fakes the original table without constrains
  - Isolate the table from constrains – I don't need unnecessary data
- tSQL.FakeFunction
  - Simply replaces the original function
  - Allow to simplify the logic
- tSQLt.SpyProcedure
  - Replace the original SP with a spy
  - The spy will record the parameters that were passed to it

ISOLATING OBJECTS

Adding unit tests with tSQLt to the database deployment pipeline

- ApplyConstraint
- RemoveObjectIfExists
- ApplyTrigger
- RemoveObject

ASSERTS

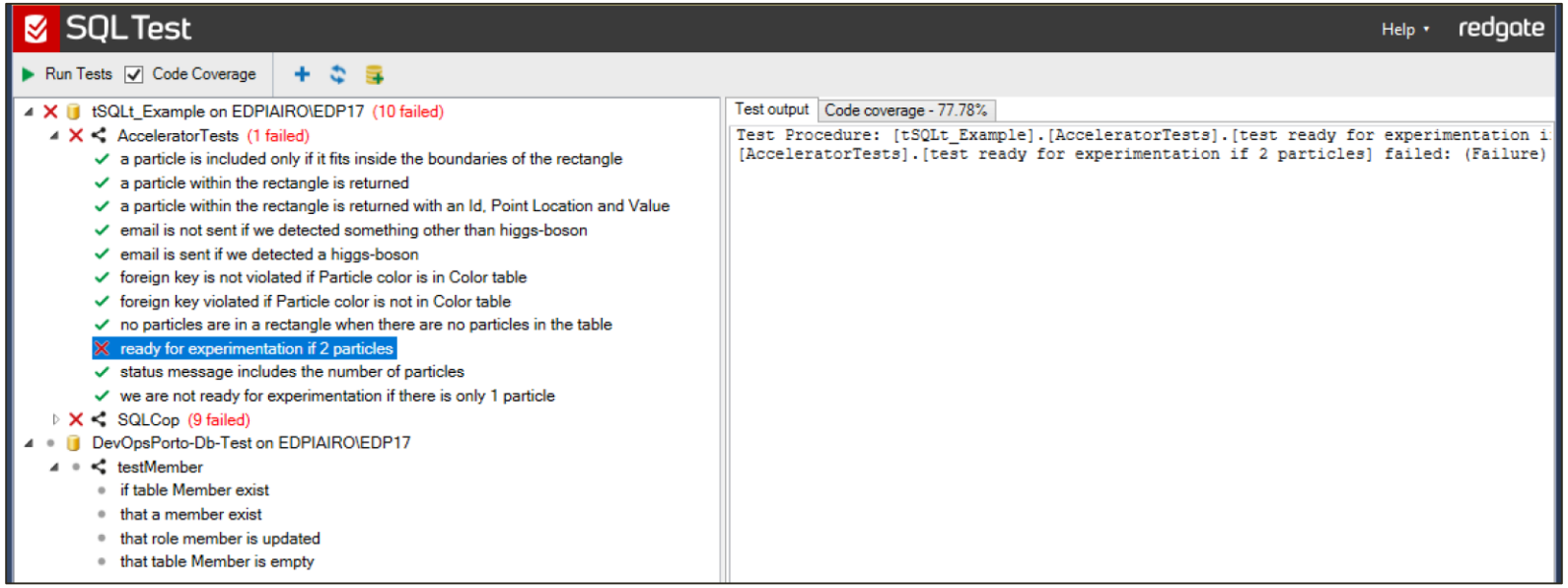
Adding unit tests with tSQLt to the database deployment pipeline

- tSQLt.AsserEquals
- tSQLr.AssertEqualsTable
- tSQLt.AssertEmptyTable
- tSQLt.AssertEqualsString
- tSQLt.AssertEqualsTableSchema
- tSQLt.AssertLike
- tSQLt.AssertNotEquals
- AssertObjectDoesNotExist
- AssertObjectExists
- AssertResultSetsHaveSameMetaData
- Fail

SQLTEST

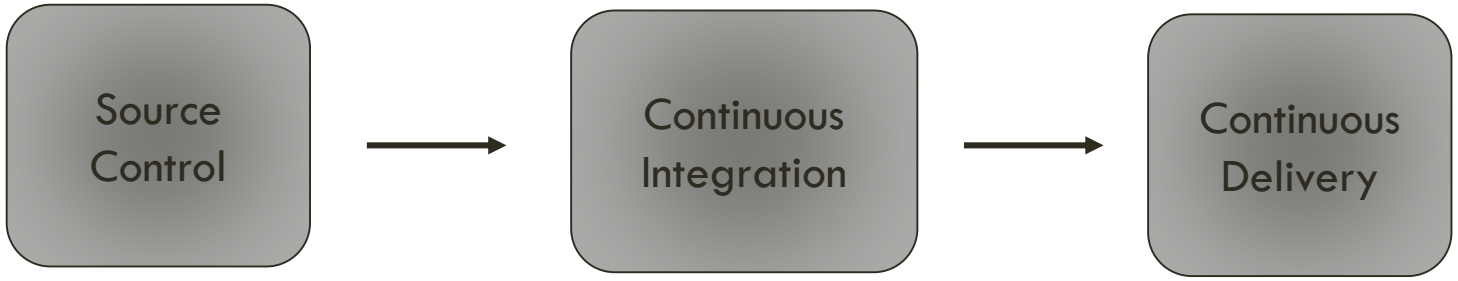
Adding unit tests with tSQLt to the database deployment pipeline

- Redgate SQL Test
  - Add-in for SSMS for creating and running unit tests
  - Measure the code coverage of those tests



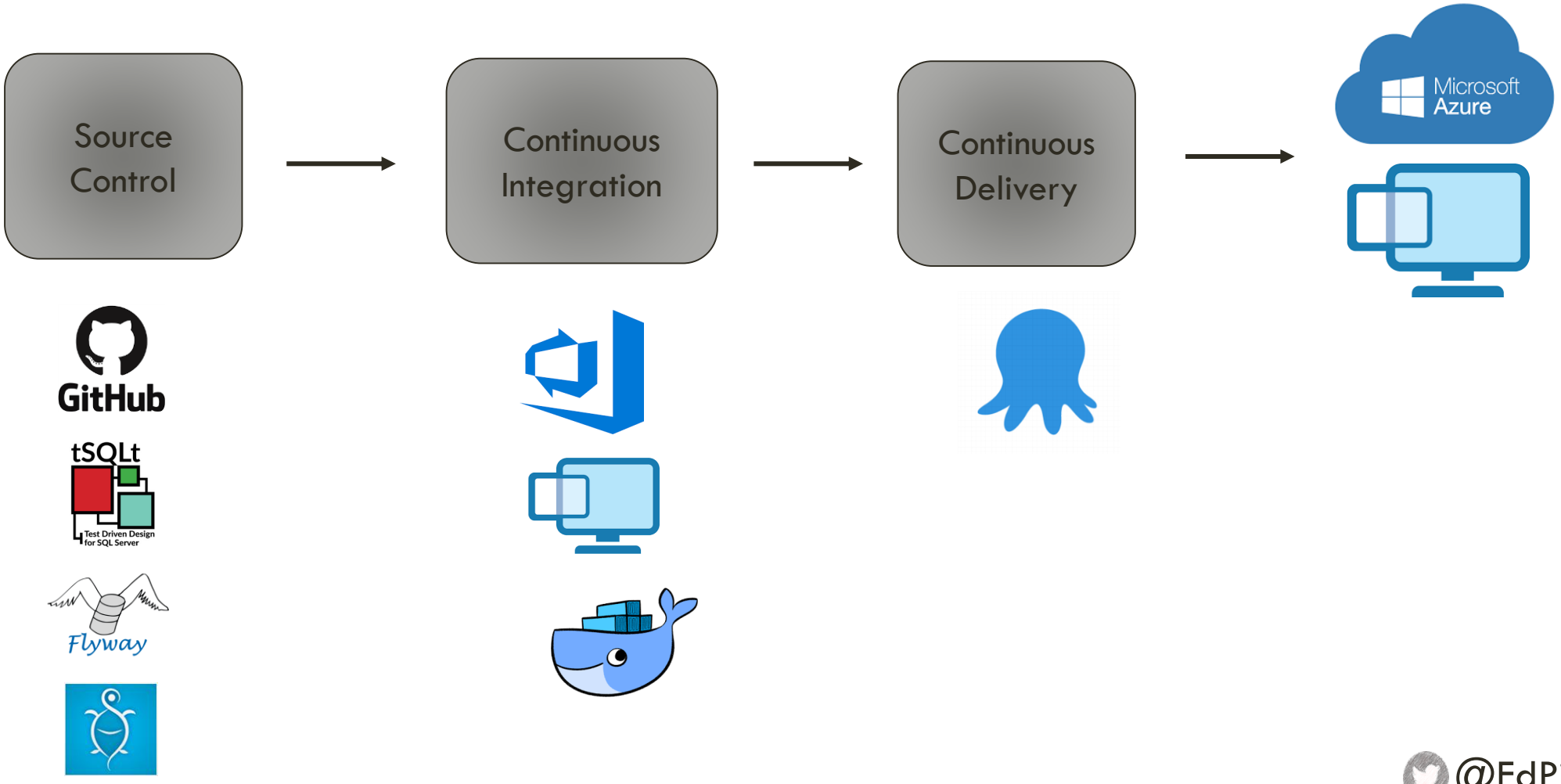
ADDING TSQLT TO THE DEPLOYMENT PIPEPLINE

Adding unit tests with tSQLt to the database deployment pipeline



ADDING TSQLT TO THE DEPLOYMENT PIPEPLINE

Adding unit tests with tSQLt to the database deployment pipeline



SOURCE CONTROL

Adding unit tests with tSQLt to the database deployment pipeline

- GitHub
  - T-SQL migrations
  - tSQLt tests
  - Pester tests
  - Building scripts

CONTINUOUS INTEGRATION

Adding unit tests with tSQLt to the database deployment pipeline

- VSTS
  - TEST environment setup (Local Machine | Docker)
    - Test and Report
  - Build artefact

CONTINUOUS DELIVERY

Adding unit tests with tSQLt to the database deployment pipeline

- Octopus Deploy
  - Local Machine
  - Azure

REFERENCES & MATERIALS

Adding unit tests with tSQLt to the database deployment pipeline

- References
  - https://tsqlt.org/user-guide/
  - https://courses.agilesql.club/
- Source code available @ GitHub
  - https://github.com/eduardopiairo/devopsporto-db

Q&A

Adding unit tests with tSQLt to the database deployment pipeline

@EdPiairo

eduardopiairo@gmail.com

https://www.eduardopiairo.com

https://pt.linkedin.com/in/eduardopiairo