**Purpose of document: Handout for Exercises**

**Autor: Uwe Ricken**

**Version: 1.00**

**Date of documentation: 15.01.2025**

**Date of approval: 16.02.2025**

**Approved by: Uwe Ricken – db Berater GmbH**

**Copyright: db Berater GmbH**

Unless otherwise specified, the documents are for your personal and non-commercial use. You may not modify copy, distribute, transmit, display, perform, reproduce, publish, license, create derivative works from, transfer, or sell any information, software, products or services.

Table of Content

[Purpose of this Document 3](#_Toc191045878)

[Prerequisites 4](#_Toc191045879)

[SQL Alias 4](#_Toc191045880)

[Exercise 1: Performance Tools – Examples 5](#_Toc191045881)

[Demo of Query Store 5](#_Toc191045882)

[Demo of Windows Admin Center 5](#_Toc191045883)

[SQL Antipatterns 6](#_Toc191045884)

[Scenario #1 7](#_Toc191045885)

[Problem / Description: 7](#_Toc191045886)

[Preparation 7](#_Toc191045887)

[Scenario #2 8](#_Toc191045888)

[Problem / Description 8](#_Toc191045889)

[Preparation 8](#_Toc191045890)

# Purpose of this Document

This document is a guide for the Hands On Exercises for the Workshop “Accelerate your T-SQL Code”.

In the " Accelerate your T-SQL Code " workshop, participants will engage in hands-on exercises designed to elevate their T-SQL coding skills through real-world scenarios and performance improvement techniques.

This session aims to identify and resolve common anti-patterns that hinder T-SQL performance. Attendees will work on 4 to 6 practical exercises that illustrate typical pitfalls and demonstrate best practices for optimizing TSQL code. Through these exercises, participants will gain a deeper understanding of query optimization, indexing strategies, and efficient data retrieval methods. The workshop will also cover the importance of avoiding costly operations such as excessive joins, subqueries, and non-sargable predicates. By the end of the session, participants will be equipped with the knowledge and skills to write more efficient, scalable, and maintainable T-SQL code.

# Prerequisites

## SQL Alias

To avoid changes in the workshop scripts it is recommended to create a SQL Alias for your Instance of Microsoft SQL Server you want to run all exercises. The Alias used in all JSON templates for SQLQueryStress and OSTRESS is “SQLServer”.

Ein Bild, das Text, Screenshot, Software, Computersymbol enthält.

KI-generierte Inhalte können fehlerhaft sein.

Figure 1: Create a SQL Alias for your SQL Server Instance

# Exercise 1: Performance Tools – Examples

## Demo of Query Store

* Execute the script “01 – demo of Query Store – preparation.sql” from the folder “02 – performance tools” on the Instance “SQLServer".
* Open the tool SQLQueryStress and load the prepared configuration file “01 – demo of Query Store.json” from the folder “SQL Query Stress” and execute the workload.
* Open Query Store from ERP\_DEMO and have a look to the execution plan
* Open the script “02 – demo of Query Store – optimization.sql” from the folder “02 – performance tools” on the instance “SQLServer”
* Implement the indexes step by step with execution of the workload from SQLQueryStress in between each optimization phase
* After the workload look to the Query Store and the result of the optimization process.

## Demo of Windows Admin Center

* Execute the script “01 – demo of Windows Admin Center – preparation” from the folder “02 – performance tools” on the Instance “SQLServer”.
* Open the script “Windows Admin Server Demo.json” from the folder “Windows Admin Center” and replace the name of the SQL Server Instance to the ORIGINAL Instance name of your installed Microsoft SQL Server Instance.
* It might be necessary to translate the other counters into English Counter Names, too!
* Start Windows Admin Center
* Open Windows Admin Center and import the "Windows Admin Server Demo.json from the "Windows Admin Center" folder
* Open the tool SQLQueryStress and load the prepared configuration file “02 – demo of Windows Admin Center.json” from the folder “SQL Query Stress” and execute the workload.
* Watch the counters in Windows Admin Center.
* Open the script “02 – demo of Query Store – optimization.sql” from the folder “02 – performance tools” on the instance “SQLServer”
* Implement the indexes step by step with execution of the workload from SQLQueryStress in between each optimization phase
* After the workload look to the Query Store and the result of the optimization process.

# SQL Antipatterns

* Before you start with the demo, create an extended event to cover all SQL Antipatterns occurring in the executed SQL queries.
* Open the script “**XEvent - SQL Antipatterns.sql**” from the folder “**05 - SQL Extended Events**” and execute the script.
* Execute the script “**01 - SQL Antipatterns - Preparation.sql**” from the folder “**03 – SQL Antipatterns**” and execute it to generate the exercise.

## High CPU / CONVERT\_IMPLICIT

* Open the script “02 - SQL Anti Paterns - High CPU.sql” from the folder “03 – SQ Antipatterns” and execute it.
* Follow the instructions of the Trainer / Speaker

# Scenario #1

## Problem / Description:

The development team love to work with user defined functions (UDF).

So they decided to create an UDF which calculates the status of any customer by year.

The calculation is a simple math:

* A customer: More or equal than 20 orders in a given year
* B customer: 10 - 19 orders for a given year
* C customer: 05 - 09 orders for a given year
* D customer: 01 - 04 orders for a given year
* Z customer: no orders for a given year

## Preparation

* Execute the script “**01 - scenario 01 - preparation.sql**” from the folder “Scenario 01” to implement all indexes and constraints
* Execute the script “**02 - scenario 01 - user defined function.sql**” from the folder “Scenario 01” to implement the user defined function [dbo].[calculate\_customer\_category].
* With the script “**02 - scenario 01 - verify function.sql**” you can check the proper functionality of the function.
* Execute the script “**03 - scenario 01 - get\_customer\_classification.sql**” to create the stored procedure for the stress tests with SQLQueryStress or OSTRESS.
* Start Windows Admin Center and load the template “**Workshop – scenario 01.json**” from the folder “**Windows Admin Center**”.
* It may be necessary to adjust parameters such as drive letters individually.
* For the "Logical Disk" instance, select the drive on which your system database TEMPDB is located.
* Start SQLQueryStress and load the template “**Workshop - scenario 01.json**” from the folder “SQL Query Stress”  
  OR  
  execute OSTRESS by executing the file “scenario 01.cmd from the folder “SQL ostress”

# Scenario #2

## Problem / Description

A software uses a table to queue jobs. Whenever a new job is to be queued,

its details are written into a table. The table grows very quickly, as up to

100,000 jobs can be scheduled in one hour.

* Sometimes the table is growing very fast
* The maintenance job cannot scale.
* The table is growing
* The system suffers from performance problems

## Preparation

* Execute the script “**01 - scenario 02 - preparation.sql**” from the folder “Scenario 02” to implement all tables, indexes and constraints
* Execute the script “**02 - scenario 02 – maintenance procedure.sql**” from the folder “Scenario 02” to implement the stored procedure [dbo].[jobqueue\_delete].
* Open the script “**02 - scenario 01 – stress query.sql**”. We need it for the upcoming tests!
* Start Windows Admin Center and load the template “**Workshop – scenario 02.json**” from the folder “**Windows Admin Center**”.
* Execute the previously opened stress query and watch the metrics.
* Load “**Workshop – Scenario 02.json**” in SQLQueryStress and check the metrics.

# Scenario #3