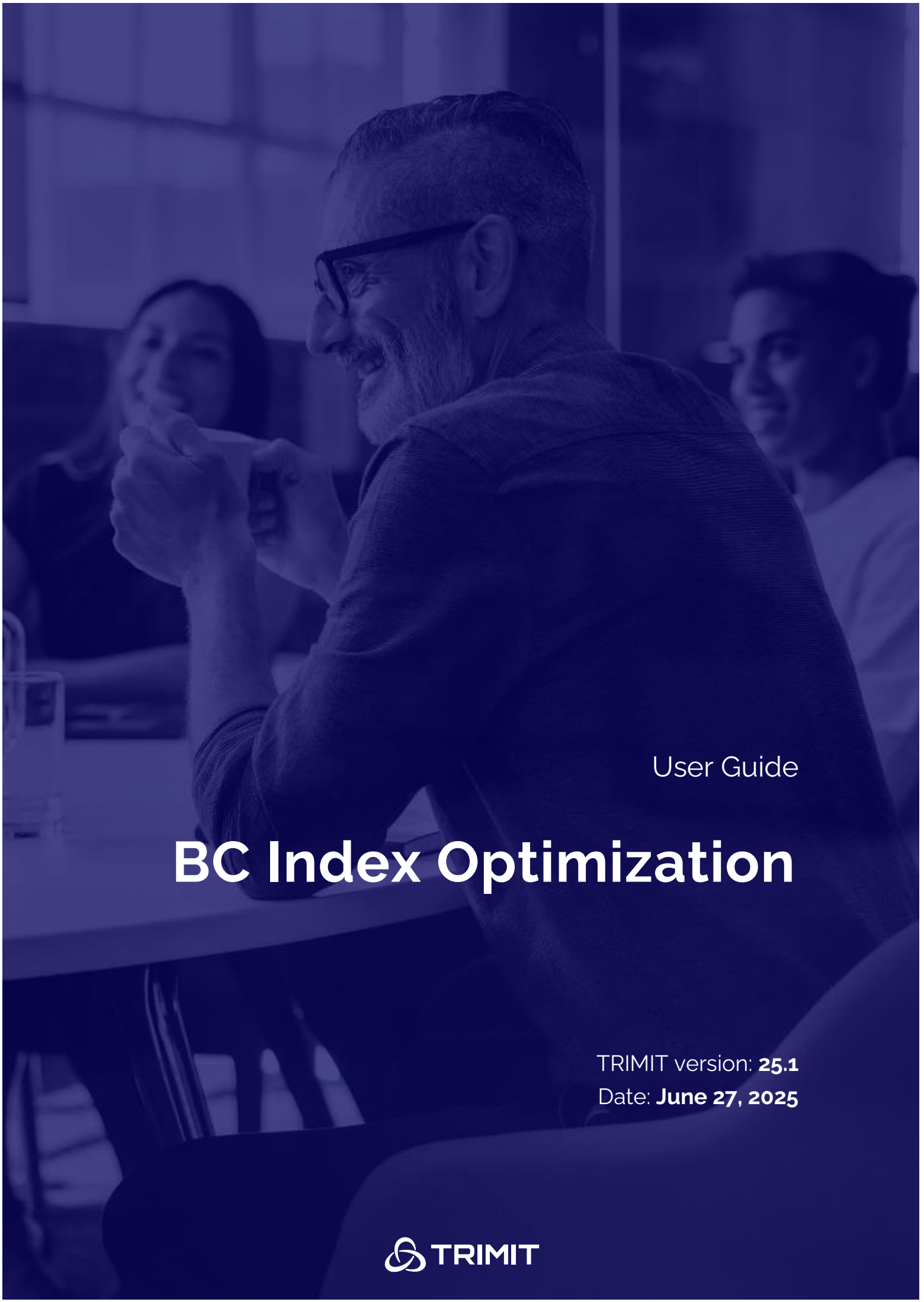

User Guide

BC Index Optimization

TRIMIT version: **25.1**

Date: **June 27, 2025**

TRIMIT

Table of Contents

Introduction

This guide outlines the process of identifying and implementing missing SQL indexes in Business Central (BC) environments. Proper indexing is critical for performance, especially in large datasets where queries must retrieve data quickly and efficiently.

Background Indexes in SQL tables significantly improve read performance by allowing the database engine to locate data quickly. However, indexes also have trade-offs:

- Slower write operations: Indexes must be updated when data is inserted, modified, or deleted.
- Increased storage requirements: Multiple indexes can increase the database size, which impacts:
    - Backup and restore times
    - Cloud storage costs, especially for development and sandbox environments in BC SaaS (Business Central online)

This guide provides a methodical approach to identifying and applying missing indexes in both on-premises and cloud environments.

The audience for this document is the Consultant or very skilled User.
Be aware of not changing settings and parameters in a live database without consulting the implementing partner.

Scope

This document applies to two types of BC installations:

- On-Premises (typically BC13 and earlier) with full SQL Server access
- Cloud (BC26 and later) running in Microsoft-hosted environments

Note:
Although BC SaaS versions prior to BC26 (e.g., BC25) offer basic missing index visibility, they lack proper impact assessment tools. Therefore, their usefulness in performance tuning is limited.

Goal

The primary objective is to optimize one or more well-defined work scenarios from the customer's daily operations. Performance improvements should be measurable, ideally in scenarios that complete within 5–10 minutes and can be repeated consistently.

Index Optimization Workflow

Step 1: Define and Document Performance Scenarios

- Customer provides one or more detailed business processes where performance is a concern.
- The process must be measurable using a stopwatch or BC's Page Scripting tools.
  Tip: If the customer can provide a recording using the Page Scripting tool in Business Central, it will be much easier to measure the impact of the index optimization.
  The exported YAML file can be reused to run and compare the same scenario before and after the optimization.
  Learn more about Page Scripting here: Microsoft Learn - Page Scripting

Step 2: Prepare Sandbox Environment

- Make a fresh copy of the production environment into a new sandbox.
  This will reset SQL Server cache and missing index statistics.

Note:
Because optimization of one scenario can often have a positive effect on other scenarios, we recommend installing and using the index-optimization App **before Step 3** in any subsequent scenarios. For the same reason, each scenario should be handled and measured independently to ensure accurate results.

Step 3: Run the Scenario

- Execute the defined business process in the sandbox.
- Record the execution time as a performance baseline.

Step 4: Identify Missing Indexes

- On-Premises:
  - Run a SQL script (e.g., DMVs for missing index suggestions) to identify high-impact missing indexes.
- Cloud (BC26+):
  - Use the "Missing Indexes" page in BC to view recommended indexes and their estimated performance impact.

Focus on indexes with the highest estimated improvement and those affecting frequently used queries.

Step 5: Create a New Sandbox Copy

- Create a second sandbox from production to ensure clean statistics.
  This will reset SQL cache and missing index statistics again to ensure consistency.

Step 6: Implement Suggested Indexes

- On-Premises (BC13 and earlier):
  - Add supplemental indexes directly to table objects using C/AL.
- Cloud (BC26+):
  - Create a new App with table extensions that define the recommended indexes.
  - Deploy the App to the sandbox environment.

Step 7: Re-Test the Scenario

- Run the same business process in the updated sandbox.
- Record the execution time again.
- Compare to the baseline from Step 3.

Step 8: Apply Indexes to Production (if Improvement Is Validated)

- If performance has improved significantly:
  - On-Premises: Apply the same index definitions directly in the production table objects.
  - Cloud: Package the tested App and deploy it through the standard App update pipeline to production.

Additional Considerations

- Index translation: Translating SQL Server index suggestions into C/AL or AL code may not be direct. Developers must understand index structure, key fields, and include columns.
- AL Development Skills: For BC26+, a developer must be able to create and publish Apps containing table extensions with indexes.
- Monitoring post-deployment: Track performance after deploying indexes to ensure ongoing benefits and no negative side effects.

Need Assistance?

If you are unsure how to convert index recommendations into AL or need help building and deploying Apps with indexes, please create an Service Request via Create service request.