Performance Troubleshooting Plan – Stored Procedure Performance issue in Production

# Issue Overview

A stored procedure used in the application is exhibiting significant performance degradation in Production, largely due to the volume of data involved. The stored procedure is fed by a scalar/table-valued function, and together they are returning results far slower in Prod than in lower environments. This performance delta is not reproducible in Dev or TST, as those environments lack a comparable volume of data. Attempting to load production-scale data into these environments would interfere with active development and testing, potentially disrupting work in progress.

# Agreed Approach

To safely isolate and analyze the issue without impacting the existing Dev environment, we will:  
1. Create a separate storage database in Dev (referred to as `HDTStorage`).  
2. Copy the required database objects and data from Prod to `HDTStorage` using the SQL Server Import and Export Wizard:  
- Tables involved in the procedure  
- Indexes supporting the queries  
- The function(s) feeding the stored procedure  
- The stored procedure itself  
  
This allows us to simulate a production-like load in a sandboxed location while protecting the integrity of ongoing dev/test cycles.

# Steps Using SQL Server Import and Export Wizard

1. Open SQL Server Management Studio (SSMS).  
   2. Right-click on the source Prod database > Tasks > Export Data.  
   3. Set the destination to the `HDTStorage` (you may need to create this database first).  
   4. Select specific tables and views—include:  
   - All tables referenced by the function and stored procedure  
   - Any lookup/reference tables needed to ensure referential integrity  
   5. Ensure indexes are copied (you may need to script them manually if not included).  
   6. After export:  
   - Manually script and deploy the function and stored procedure  
   - Verify dependencies (linked servers, permissions, synonyms, etc.; if necessary)

7. Once the research and analysis are complete, it’s important to clean up any objects that were temporarily added to the storage database. This includes removing any copied tables, indexes, stored procedures, or functions that are no longer needed. Doing so helps keep the environment organized and prevents confusion or accidental usage in future development work.

# Additional Recommendations

To make this effort more sustainable and reduce the data burden in Prod long-term:  
  
1. Data Archiving Strategy  
Identify historical data that can be archived to a separate table, partition, or external storage solution (e.g., blob storage, cold storage). This improves query performance and reduces memory/IO pressure.  
  
2. Implement Query Hints or Rewrites  
Review the function and stored procedure logic for:  
- Scalar UDFs that could be rewritten as inline table-valued functions  
- Loops or cursors that can be replaced with set-based logic  
- Missing or inefficient joins  
  
3. Index Tuning & Statistics (currently being researched in this work)  
Run a detailed index analysis on the affected tables in Prod. Consider:  
- Adding filtered or covering indexes  
- Ensuring stats are up to date  
- Reviewing fragmentation levels  
  
4. Temp Table Optimization (if applicable)  
If the procedure uses temp tables or table variables, check for:  
- Proper indexing  
- Table variable misuse that’s causing poor cardinality estimation  
  
5. Query Store & Actual Execution Plan Review (currently being researched in this work)  
Use Query Store, or capture actual execution plans in the test database to compare with Prod and pinpoint regressions.