

PkMS nTier Load Testing Overview.

Please note this document does not “replace” the PkMS documentation, rather it offers some additional “hints” and “suggestions”.

Load Testing PkMS nTier can often identify system performance problems and bottle-necks that are slowing down warehouse operations. Once the load test has identified problem transactions, these transactions can be studied in more detail and improved.

Approach to Load Testing

The approach to load testing is usually as follows:

- 1. Identify Transactions to be Tested**
The transactions that are used the most in the warehouse are the best ones to load test. These may include RF client transactions (RF Receiving), submitted transactions (Wave Server, ASN Bridge), and GUI Transactions (Case Inquiry).
- 2. Identify Testing Scenarios**
Scenarios for testing must be identified. This first involves determining the “best guess” level of the maximum normal load on each transaction to be tested. This load is usually determined by the max number of users and how many of each transaction each of those users is liable to perform in an hour during peak warehouse operations. This is referred to as the 100% level. For example, if post-wave during peak operations, up to 25 users can each perform 50 packing tasks, then the 100% level is 1750 packing tasks per hour.
- 3. Setup Testing Environment**
A PkMS environment and database should be setup to run the load testing in. This environment preferably is on separate hardware from the production systems, so the load test will not affect production performance AND the load test results are skewed by production events. Only people involved in the load testing should have access to this environment. PkMS Configuration should mirror the production configuration as much as possible to get accurate results. It is also important to install operating and database monitoring tools to monitor the load test.
- 4. Design/Create/Test Testing Scripts**
Once the transaction are all identified, then test scripts must be recorded and coded using a automated testing tool such as Load Runner. Load runner and other tools can record scripts, but some tweaking of the script code is usually required. Once coded, test scripts can be run in isolation to test the functionality of the script. Small amounts of user created data are usually sufficient for the initial testing.
- 5. Design/Create/Test Data Scripts**
For the volume of data required during the load testing, typically scripts to generate data are written. Small amounts of data can be user generated, but for thousands of transactions worth of data, PL/SQL scripts can generate large amounts of PkMS data.

Once a data script is completed, the data should be tested in isolation by running the testing script to consume the data.

6. Test the Scenario(s)

After all the testing scripts and data scripts are ready, then full blown scenario testing can start. This is testing the all the transactions running together at once with transaction pacing. It is important to note that several test runs are usually required to fine tune the data, the test scripts, and the scenario itself. Once the test runs are acceptable (the data and the test scripts work at an acceptable level), a full pre-test backup of the database can be performed to provide a "Gold" database to restore before each test. This avoids having to run the data generation scripts before every test.

7. Perform Load Testing

Now that everything is ready to go, load testing can proceed. Several different scenarios are usually tested dependent on the user's requirements. Examples are 100% load for 2 hours, 100% load for 6 hours, 75% load for 24 hours, 150% load for 2 hours.

8. Analyze Results

After each load test, an analysis of the results is required. Analysing the results requires checking several sources. Of primary importance is to check the transaction timings from the testing tool to see what level of performance each transaction is providing. Operating system tools can provide results showing CPU usage, memory usage or concerns during testing. Database monitoring tools can show database locking problems, paging problems, or under performing SQL queries. PkMS logs are checked to find application errors or warnings generated during testing.

9. System Performance Improvements

Once problem transactions are identified, and analysis indicates the type of problem, improvements can be made to the system. Some examples of improvements would include:

- Different configuration of PkMS

- Reworking of SQL

- Source code performance improvements

- Operating system tweaks

- Database tweaks

- Hardware upgrades

In conclusion, load testing can have a very positive on system performance by allowing the identification and improvement of system performance problems and transaction bottle necks. But it is important to have realistic expectations and conduct the load testing with sound methodology.