

Kimball Design Tip #52: Let's Improve Our Operating Procedures

By Joy Mundy

In my career I've been able to review a lot of data warehouses, in various stages of their lifecycles. I've observed that, broadly speaking, we are not very good about operating the data warehouse system with anything like the rigor that the transaction system guys expect of their systems. In all fairness, a data warehouse is not a transaction system, and few companies can justify a 24x7 service level agreement for data warehouse access. But come on guys, do we have to look like Keystone Kops in an emergency? As we all know, Bad Things Happen — especially as a data warehouse is downstream of every other system in your company.

Operating a data warehouse in a professional manner is not much different than any other systems operations: follow standard best practices, plan for disaster, and practice. Here are some basic suggestions, based on my observations from actual deployments.

Negotiate a service level agreement with the business users. The key here is to negotiate, and then to take that SLA seriously. The decision about service level must be made between the executive sponsor and the data warehouse team leader, based on a thoughtful analysis of the costs and benefits of ratcheting up the SLA towards high availability. The basic outlines of the SLA need to be negotiated early in the project, as a requirement for high availability may significantly change the details of your physical architecture.

Use service accounts for all data warehouse operations. You'd think it would go without saying that all production operations should use a specified service account with appropriate permissions. But I've long lost count of how many times I've seen production loads fail because the DBA left the company and her personal account becomes inactive.

Isolate development from testing from production. Again, it should go without saying. Again, apparently it doesn't. I've observed two main barriers against teams' being rigorous about Dev/Test/Prod procedures: cost and complexity.

The hardware and software costs can be significant, because the best practice is to configure a test system identically to its corresponding production system. You may be able to negotiate reduced software licensing costs for the test system, but the hardware vendors are seldom so accommodating. If you have to skimp on hardware, reduce storage first, testing with a subset of historical data. Next I'd reduce the number of processors. As a last resort, I'd reduce the memory on the test machine. I really hate to make these compromises, because processing and query performance might change in a discontinuous fashion. In other words, the test system might behave substantially differently with reduced data, processors, or memory. The development hardware systems are usually normal desktop machines, although their software should be virtually identical to Test and Prod. Coerce your software vendors to provide as many development licenses as you need at near-zero cost. I think all Dev licenses should cost less than \$100. (Good luck!)

Everything that you do to the production system should have been designed in Dev and the deployment script tested on Test. Every operation on the backend should go through rigorous scripting and testing, whether deploying a new data mart, adding a column, changing indexes, changing your aggregate design, modifying a database parameter, backing up, or restoring. Centrally managed front room operations like deploying new query and reporting tools, deploying new corporate reports, and changing security plans, should be equally rigorously tested, and scripted if your frontend tools allow it.

When you are scripting operations, parameterize connection information like ServerName into configuration files, if your tools permit. (If they don't permit it, excoriate your vendor until they add such a feature.) Script everything you possibly can, and then check those scripts into source control.

Data warehouse software vendors do not make it easy for you to do the right thing. The front-end tools and OLAP servers are particularly bad about helping – or even permitting – the development of scripts for incremental operations. It is very challenging to coordinate the rollout of a new subject area across RDBMS, ETL system, analysis, and reporting systems. Be very careful, and test, test, test!

Keep on top of service packs, hot fixes, and product upgrades. The appropriate people on the DW team should be responsible for monitoring upgrades, including fixes, for all products used in the data warehouse system. Set a recurring weekly appointment to browse the appropriate websites to see what's available, and evaluate how important each fix or release is to your project. You shouldn't install every fix as soon as it comes out, but you should be educated about them. If you're proactive, you can save your expensive calls to Tech Support for problems that haven't already been solved.

Develop playbooks for all operations. A playbook contains step-by-step instructions for performing an operation such as restoring a database or table, or deploying a new data mart, or adding a new column to a table. You should develop generic playbooks, and then customize that playbook for each operation you plan to perform in production. For example, if you are changing a database parameter, write down, in reasonable detail, the steps to follow. Then test the playbook on the Test system before applying to Prod. This is absolutely vital if you are performing an operation through a tool's GUI rather than via a script.

Operations is not the fun part of data warehousing. But with good planning and practice, you can meet the inevitable snafu with calm and deliberation, rather than hysteria.