Data and Database Integrity Testing

Mike Kelly

First Published on SQAtester.com

What is Data and Database Integrity Testing?

Data integrity and database integrity test techniques verify that data is being stored by the system in a manner where the data is not compromised by updating, restoration, or retrieval processing. This type of testing is intended to uncover design flaws that may result in data corruption, unauthorized data access, lack of data integrity across multiple tables, and lack of adequate transaction performance (Dustin 252). The databases, data files, and the database or data file processes should be tested as a subsystem within the application.

Determine What To Test

The first step in determining what need to be tested is to identify the types of database activity (insert, delete, update) that will be invoked, and to identify when these transactions will occur within the application-under-test. It can also be helpful to work with the application designers and architects to determine the calculations or processing rules used by the application, as well as the time-critical transactions / functions / conditions, and possible causes of poor performance (Dustin 136). At this time the dataflow for the application should also be examined and tests that mimic it's path(s) should be earmarked for planning and execution. This high level approach will set the baseline for your test plan. It will identify the areas that you will need to investigate further; and it will allow you to start planning for what types of testing you will need (manual, automated, etc.).

The next step is to look in depth at the design, code, databases, and file structures. You will want to check data fields for alphabetic and numeric characters. You will need to look at files and fields to ensure proper spacing and length. You will want to develop some method to verify correct data format(s). Your data consistency checks should include both internal and external validations of essential data fields. Internal checks involve data type checking and ensure that columns are of the correct types; external checks involve the validation or relational integrity to determine whether duplicate data are being loaded from different files. Also at this time you will want to be sure to plan tests for all data files; including clip art, templates, tutorials, samples, etc.

Finally, you will want to take a look at conditions of the system that can affect the performance or stability of the data file(s) or database(s). This will include testing conditions like low memory, low disk space, and other limited resources. It will also include backup integrity and recovery testing at multiple levels; including, but not limited to, restoring individual files, restoring application data, restoring a database or other information stores, and full system recovery.

When planning and designing your data and database tests it will be helpful if you keep the following in mind. It is a good practice to keep backup copies of test files and test databases. Before reporting an error, check your working copies of the input and comparison files against the backups. Further, all data and database sub-systems should be tested, at some point, without the target-of-test's user interface. This removes the possibility of errors being introduced by the user interface.

Automating Data and Database Testing

There are some simple tools that can be used to automate some of the mundane tasks needed to perform data and database testing. These tools include files comparison utilities, files viewers, and file format converters. These tools can be used to show what changes must be made to one comparison file in order to produce the other, compare object code, graphics, and compressed data files, look at the data in disk files in many different formats, and convert data files, text files, or graphic files, from one format into another.

More complicated tools are available that can aid you in testing database connectivity, prevent unplanned downtime and slow performance, create valid test data, improve data movement and debugging messages, and thoroughly test proprietary database calls, syntax and structures. These are numerous and information on them can readily be found via Google.

Just keep in mind that automated tests should be designed to operate as stand-alone tests. This means that the output for one test does not serve as the input to the next test. What this allows you to do is to create a database-reset script, which is a script that is designed to restore the database to a known baseline allowing the next test to start at that known baseline with little to no setup required (Dustin 328).

Example of Good Data Integrity Testing

The following is a vendor alert posted by Nasdaq. This alert illustrates the great pains Nasdaq takes to ensure data integrity and stability.

"The Nasdaq Stock Market® frequently disseminates test data over the Nasdaq market data feeds during non-market hours. To prevent database corruption, market data vendors should treat all messages received after the "End of Transmission" control message on a Nasdaq® data feed as test data. In addition, vendors should treat any data disseminated during the weekends or market holidays as test data.

For quality assurance purposes, The Nasdaq Stock Market conducts testing of its computer systems and telecommunications network during non-market hours. Frequently, Nasdaq disseminates test data over the data feed circuits in order to verify the integrity of the MCI WorldCom Intelligent Shared Network (ISN). As a result, vendors also receive the test data over the Nasdaq data feed lines during non-market hours.

During market hours, Nasdaq disseminates test data with a special retransmission requester code in the message header. After-hours test data, however, does not support this special code. Due to the testing methods utilized, after-hours test data typically is disseminated in the same manner as an original transmission.

To insure database integrity, Nasdaq recommends that vendors treat any data received after the "End of Transmission" control message or during non-market days as test data. The "End of Transmission" control message is disseminated at approximately 6:37 p.m. eastern time across the Level 1 and NQDS feeds, and at approximately 5:36 p.m. for the NTDS feed. Non-market days include weekends and holidays.

In advance of a data format change or major processing change, Nasdaq will schedule special vendor testing opportunities. Notification of special vendor testing opportunities and holiday will be posted to the Release Schedule on the Nasdaq TraderSM Web site. During the designated times, vendors will be encouraged to use test data to check their system processing. Unless otherwise instructed, however, vendors should disregard and/or dispose of test data…."

-Vendor Alert #1998-7 - February 24, 1998

References

For more information on data and database testing I would recommend the following books. Not only do they contain useful information on data and database testing, they both contain a wealth of information on testing techniques and theory.

Dustin, Rashka, and John Paul. Automated Software Testing. Reading: Addison-Wesley. 1999.

Kaner, Falk, and Hung Quoc Nguyen. Testing Computer Software. Second Edition. New York: Wiley. 1999.