The Resource for Users of IBM zSeries & S/390 Systems
zJOURNAL

December 2010 / January 2011

zData Perspectives

Rebinding for Optimal DB2 Access Paths

by Craig S. Mullins

The access paths formulated by the DB2 optimizer during the BIND and REBIND processes are critical to your application performance. These access paths determine not only how DB2 data is accessed by your program, but how efficiently it's accessed. Whether you're preparing a new program, implementing changes into your existing DB2 applications, upgrading to a new version of DB2, or simply trying to achieve optimum performance for existing applications, an exhaustive and thorough REBIND management policy should be of paramount importance.

However, many organizations aren't doing everything possible to keep access paths up-to-date with the current state of their data. So, what's the best practice approach for rebinding your DB2 programs? The answer is "The Five R's," a methodology of regular rebinding followed by a review of your access paths:

Of course, there are products that can be used to implement a proactive approach to rebinding. These products preview the new access paths and then run them through a rules system to determine if the new access will be improved, unchanged, or degraded. With this information, we can rebind everything that would improve and avoid rebinding anything else until we can analyze the cause of the degradation. Using such an approach should prevent degraded access paths from sneaking into your production environment. You will be using REBIND to take advantage of new and improved performance for your SQL statements but also avoiding unpleasant performance "surprises." Being able to avoid a single instance of degraded SQL can improve overall performance, prevent untold struggles, and deliver the ROI that management expects.

In mainframe environments, change is typically tightly controlled and managed. Every change we make to application programs must be thoroughly tested before it's promoted to production. Every change is tested to minimize the risk of unintended consequences. We perform the same type of due diligence with

1. Inspect the Real Time Statistics (RTS) to determine which objects need to be reorganized
2. Run a REORG on table spaces and indexes as appropriate based on the statistics
3. Run RUNSTATS (to ensure the DB2 Catalog is up-to-date)
4. REBIND your programs (that access any of the objects that were reorganized)
5. Review the access paths generated by the REBIND.

For shops that have avoided rebinding for years, this approach represents a significant change. So, what new DB2 features are available to help? Well, DB2 9 provides plan stability. This feature enables you to save a backup version of your access paths as a precautionary measure. If any of the new access paths are less efficient after rebinding, the DBA can switch back to the backed up access paths. To implement this level of stability, you can use the PLANMGMT parameter of the REBIND command. There are three options: NONE, BASIC, and EXTENDED. BASIC saves the previous access paths, and EXTENDED saves the previous and an original.

You can use REBIND and the SWITCH parameter to revert to the saved access paths when the new access paths cause degraded performance. Of course, this is a reactive approach, but it's a nice precautionary measure when you're actively rebinding DB2 programs.

most other mainframe changes, including database changes, system software (z/OS, CICS, IMS, etc.) changes, PTFs, etc. Almost every type of change to the mainframe environment is subject to strict change control procedures. We do this to maximize availability because minor problems can disrupt production work.

But one exception to the tightly controlled change management environment of the mainframe has been DB2 access paths. We run BINDs and REBINDs in the production environments without the benefit of oversight or prior testing. Oh, we may try to mimic the production environment in test, but we can't preview production access paths in the production environment. This lack of change control results in unpredictable performance impacts. Because programs are moved to production and bound there, we're at the mercy of the DB2 optimizer, which generates access paths on the fly for our programs. Any issues with inefficient access paths are then dealt with in a reactive mode; that is, problems are addressed after the fact.

Using a software package such as previously described can help alleviate this problem by enforcing change control on your DB2 access paths. Implementing such a standard can help improve the stability and performance of your DB2 applications.

From zJournal, Dec 2010 / January 2011.