

Kimball Design Tip #27: Being Offline As Little As Possible

By Ralph Kimball

If you update your data warehouse each day, you have a characteristic scramble when you take yesterday's data offline and bring today's data online. During that scramble, your data warehouse is probably unavailable. If all your end users are in the same time zone, you may not be feeling much pressure, as long as you can run the update between 3 and 5 am. But, more likely, if your end users are dispersed across the country or around the world, you want to be offline absolutely as little as possible, because in your case, the sun never sets on the data warehouse. So, how can you reduce this downtime to the bare minimum?

In this design tip, we'll describe a set of techniques that will work for all of the major relational DBMSs that support partitioning. The exact details of administering partitions will vary quite a bit across the DBMSs but you will know what questions to ask.

A partition in a DBMS is a physical segment of a DBMS table. Although the table has a single name as far as applications are concerned, a partitioned table can be managed as if it is made up of separate physical files. In this design tip, I assume your partitioning allows

- * moving a partition, but not the whole table, to a new storage device
- * taking a partition, but not the whole table, offline
- * dropping and rebuilding any index on the partition, but not the whole table
- * adding, deleting, and modifying records within a designated partition
- * renaming a partition
- * replacing a partition with an alternate copy of that partition

Your DBMS lets you partition a table based on a sort order that you specify. If you add data on a daily basis, you need to partition your fact tables based on the main date key in the fact table. In other design tips I have mentioned that if you are using surrogate (integer) keys then you should make sure that the surrogate keys for your date dimension table are assigned in order of the true underlying dates. That way, when you sort your fact table on the date surrogate key, all the most current records cluster in one partition.

If your fact table is named FACT, you also need an unindexed copy called LOADFACT. Actually in all the following steps we're only talking about the most current partition, not the whole table! Here are the steps to keep you offline as little as possible. We go offline at step 4 and come back online at step 7.

1. Load yesterday's data into the LOADFACT partition. Complete quality assurance pass. 2. When done loading, make a copy of LOADFACT named COPYLOADFACT. 3. Build indexes on LOADFACT.
4. Take FACT offline (actually just the most current partition). 5. Rename most current FACT partition to be SAVEFACT. 6. Rename LOADFACT (partition) to be most current FACT partition. 7. Bring FACT online.
8. Now clean up by renaming COPYLOADFACT to be the new LOADFACT. You can resume

dribbling incoming data into this new LOADFACT. 9. If all is well, you can delete SAVEFACT.

So, we have reduced the offline interval to just two renaming operations in steps 5 and 6. Almost certainly, these renaming operations will be faster if the physical size of the most current partition is as small as possible.

There is no question that this scenario is an idealized goal. Your data warehouse will have additional complexities. The main complexities you will have to think through include

- * limitations to the partitioning capability of your DBMS.
- * need to load old, stale data into your fact table that would disrupt the "most current" assumption
- * handling associated aggregate fact tables

In the next design tip, I'll extend the ideal case to cover some of these messy situations. But for a couple of weeks, we can all pretend that life is simple... I would be especially interested in hearing about how your DBMS handles partitioning and whether you have been able to get this scheme to work. I'll publish your comments. Write to me at ralph@ralphkimball.com before the next design tip.