April / May 2005

zData Perspectives

by Craig S. Mullins

Changing Table Space Partitioning Specifications

Historically, one of the biggest impediments to managing DB2 database systems has been administering partitioned table spaces. Prior to DB2 V8, it was either difficult or impossible to modify the structure and many of the parameters of a partitioned table space. Exacerbating this problem is the fact that most partitioned table spaces are the largest, most critical, table spaces in the system with the highest availability requirements. Fortunately,

DB2 V8 removes many of the barriers to managing partitioned table spaces.

As of DB2 V8 you can add partitions, rotate partitions, and change the partitioning key values. In order to gain this flexibility, though, you will need to change from index-controlled partitioning to table-controlled partitioning. Then, you can use the ALTER TABLE statement to modify most of the partitioning specifications.

To **add a partition** to an existing table space the ALTER TABLE statement has been augmented with the ADD PART parameter. For example, consider a table space that is partitioned having one fiscal quarter worth of data per partition. Eventually, you might run out of partitions and need to add one. Assume that the last partition holds data up to the third quarter of 2004, but now you need to add data past this date. Assuming that your fourth quarter ends in December, the following SQL uses ALTER to add the new partition to the table:

```
ALTER TABLE CREATOR.TBNAME

 ADD PART VALUES('12-31-2004');
```

You won't specify a partition number when you add a partition. DB2 will determine the next partition number to be used by examining information in the DB2 Catalog. You can add partitions up to the maximum limit; the maximum number of partitions depends on the DSSIZE parameter and page size of the table space.

You cannot specify attributes like PRIQTY and SECQTY; instead, DB2 uses the values in use for the previous logical partition. Before you begin to use the new partition, you might want to alter the tablespace to provide accurate space parameters for the new partition.

Each newly added partition will be immediately available for use, but you must stop the table space and partitioned index before adding the partition.

If the requirement to add a partition can be satisfied by allowing an existing partition to be reused, you might be able to ***rotate the partition***. Rotating partitions allows old data to “roll off,” but the partition is kept for new data. This is a good option any time old data is periodically archived and only a limited number of partitions need to be active.

Partition rotation is implemented using ALTER TABLE with the ALTER PART ROTATE FIRST TO LAST parameter. When rotating, if you specify the RESET parameter, the data rows in the oldest (or logically first) partition are deleted and a new table space high boundary is set so that partition becomes the last logical partition in sequence. This partition will then be ready to hold the new data as it is added. The partition that was rolled off is immediately available after the ALTER succeeds; a REORG is not required.

The aftermath of rotating a partition can be confusing. This is especially the case if you are trying to match partitions to physical data sets. The .A001 data set is now the last logical partition, not the first. You will need to use the new LOGICAL_PART column in the SYSTABLEPART table to match partitions to data sets. Additionally, you can use the DISPLAY command to list the status of table space partitions by logical partition.

Also, steps need to be taken if you need to keep the rolled off data for archival purposes. Be sure to unload the data immediately before rotating the

partition using either an UNLOAD utility or a user-written program.

You can change partition boundaries relatively easily, too. DB2 V6 introduced the ability to modify limit keys for partitions. DB2 V8 adds the same capability for table-based partitioning with the ALTER PART VALUES parameter of ALTER TABLE . The affected data partitions are placed into the REORP pending state until they have been reorganized.

Finally, you can rebalance partitions when running DB2 V8. Unlike the schema changes previously discussed, partition rebalancing is accomplished using the REORG utility with the new REBALANCE parameter. During the REORG, DB2 will rebalance the data such that it is evenly distributed across the partitions. Rebalancing is most practical when the data isn't greatly skewed.

The REORG utility will set new partition boundaries so that all the rows participating in the reorganization are evenly distributed across the partitions being reorganized. DB2 will update the SYSTABLEPART and SYSINDEXPART tables to record the new limit key

values. The values previsously set in the DDL are no longer valid -- so don't try to reference them there.

DB2 V8 greatly simplifies the maintenance of partitioned tablespaces. Be sure you understand the options at your disposal when you move to V8 to reduce the administrative burden of partitioning.

From zJournal, April / May 2005

.

Home.