



REASONS TO ISOLATE SCHEMAS INTO THEIR OWN TABLESPACES

An article by Trevor Bezuidenhout of RDB Consulting
October 2008

This article provides Database Administrators with tried and tested methodologies when managing their database tablespaces. Both Craig and Trevor have developed this approach after many years experience with Oracle databases. They have had practical experience with the benefits that these guidelines afford databases and their DBA's.

GENERAL RULES TO FOLLOW

RULE 1

All schemas to reside in their own tablespace, no mixing of schema's within the same tablespace.

RULE 2

Tables and indexes should be further separated into their own tablespaces.

RULE 3

Schema AUDITING should be to a separate schema and tablespace.

RULE 4

Schema ARCHIVING should be to a separate schema and tablespace.

RULE 5

Partitioned tables should be placed in separate tablespaces.

SIMPLE EXAMPLE WITH SUGGESTED NAMING STANDARDS

Three schemas, ABS, EBD and ESP reside in a database. Six tablespaces will be needed:

1. One tablespace for ABS tables.
2. One tablespace for ABS indexes.
3. One tablespace for EBD tables.
4. One tablespace for EBD indexes.
5. One tablespace for ESP tables.
6. One tablespace for ESP indexes.

Suggested tablespace naming standards can be as follow:

1. ABS_DATA and ABS_INDEX
2. EBD_DATA and EBD_INDEX
3. ESP_DATA and ESP_INDEX

HERE ARE THE REASONS WHY YOU WOULD WANT TO DO THIS

1. In the event of a loss or a corruption of a tablespace or data file, only a single schema would be affected for any one tablespace or datafile, making restoring and recovering of the tablespace much less of an impact to an application as the other schemas would still be available for use.
2. In the event of the loss or corruption of an index tablespace, the schema tables would still be available for access. This reduces the risk by half of losing both tables and indexes at the same time.
3. Schemas can be easily dropped along with their associated tablespaces and datafiles.
4. Schemas can be dropped without leaving behind potentially large fragmented free space within a tablespaces that is shared with other schemas, as this space cannot easily be recovered.
5. Transportable tablespaces can be performed per schema.
6. Transportable tablespaces can be parallelized on a schema basis for performance reasons i.e. three schemas in a single tablespace is a single stream for transporting, while three schemas in three tablespaces means three streams for transporting.
7. RMAN backups can be done per tablespace and hence per schema. In Data Warehouse applications, each new schema can be backed up individually and as often as required. If schemas shared tablespaces, the potential of backing up unchanged schemas is greatly increased.
8. The nature of a schema may change for example to 'read only'. In this case a tablespace could not be made 'read only' as it would potentially conflict with the characteristics of other schemas sharing the same tablespace.
9. Only a single backup of a 'read only' schema residing in its 'read only' tablespace is required. If the 'read only' schema shared a tablespace with 'read write' schemas, then the potential of unnecessarily backing up a 'read only' schema over and over again is quite real.
10. Different tablespace block sizes can be applied to individual schemas and to their tables and indexes.
11. Table and index BLOCK de-fragmentation and repacking can be easily achieved without fragmenting free space within a shared tablespace. To do this one can create a new tablespace and slowly but surely move the table objects to

the new tablespace as time allows or all at once if possible. This is in conjunction with creating a new index tablespace to rebuild the relevant indexes. Once the MOVE / REBUILD / REDEFINITION is complete the old data and index tablespace can be dropped, and the new ones renamed back to the original names. Thus releasing fragmented space within blocks, increasing block density whilst improving I/O performance and best of all, not leaving behind a highly fragmented shared tablespace.