Oracle Tablespaces, etc.: Managing the Disk Resource

CS634
Lecture 7, Feb 24, 2014

These slides are not based on “Database Management Systems” 3rd ed, Ramakrishnan and Gehrke

Look at disks we have to work with on dbs2

```
dbs2(24)% df –lk (local filesystems, subset)
Filesystem           kbytes    used   avail capacity  Mounted on
/dev/dsk/c1t0d0s3    8263373 5166817 3013923    64%    /disk/sd0d
/dev/dsk/c1t1d0s3    8263373 8180740       0   100%    /disk/sd1d
/dev/dsk/c1t0d0s4    8263373       9 8180731     1%    /disk/sd0e
/dev/dsk/c1t0d0s5    8263373       9 8180731     1%    /disk/sd0f
/dev/dsk/c1t1d0s4    8263373 1049116 7131624    13%    /disk/sd1e
/dev/dsk/c1t0d0s7    18415754      9 18231588    1%    /disk/sd0h
/dev/dsk/c1t0d0s6    16526762      9 16361486    1%    /disk/sd0g
/dev/dsk/c1t1d0s5    8263373 7343660  837080    90%    /disk/sd1f
/dev/dsk/c1t1d0s6    16526762      9 16361486    1%    /disk/sd1g
/dev/dsk/c1t1d0s7    18415754 2181679 16049918  12%    /disk/sd1h
```

- This shows two disks, /dev/dsk/clt0d0 (aka sd0) and /dev/dsk/clt1d0 (sd1), with 5 partitions each with fs's, of size 8GB, 8GB, 8GB, 16GB, and 18GB (total 50GB).
- Old disks, now would see bigger disks, but these are sufficient for our use.

Partitions of a Disk (or RAID)

- A disk can be split up into partitions, commonly only 2 or 3, but 5 each on dbs2's disks.
- A partition is a consecutive sequence of cylinders of the disk.
  - Thus it limits seek time for files within it.
- Partitions are created before file systems. Each partition may have its own filesystem.
- Under UNIX/Linux (including MacOS), file systems can be pasted together by "mounting" one filesystem on a directory of another already in use.
  - The first filesystem to be put in use has the root directory of the final filesystem.
  - You can tell what partition your current directory is part of by using the "df ." command.
- This describes local disks and partitions. It is also possible to mount a remote filesystem via NFS (network file system).
  - However, for database use, we want local disk.

Oracle Data Files: *.dbf

Disk sd0 has Oracle binaries and disk sd1 has Oracle data files, on 3 partitions:

```
dbs2(36)% sudo ls -l /disk/*/data/ora*/*
/disk/sd1d/data/oracle-10.1/dbs2:
-rw-r-----   1 oracle   104865792 Feb 23 12:06 caspar.dbf
… smaller files deleted from list…
-rw-rw----   1 oracle   1090527232 Feb 23 13:42 sysaux01.dbf
-rw-rw----   1 oracle   524296192 Feb 23 13:40 system01.dbf
-rw-rw----   1 oracle   1574969344 Feb 22 09:01 temp01.dbf
-rw-rw----   1 oracle   2123374592 Feb 23 13:42 undotbs01.dbf
-rw-rw----   1 oracle   2915049472 Feb 23 12:06 users01.dbf
/disk/sd1e/data/oracle-10.1/dbs2:
-rw-r-----   1 oracle   1073750016 Feb 23 13:40 system02.dbf
/disk/sd1f/data/oracle-10.1/dbs2:
-rw-r-----   1 oracle   3221233664 Feb 23 13:36 undotbs02.dbf
-rw-r-----   1 oracle   4294975488 Feb 23 12:06 users02.dbf
```

Tablespaces are created from OS files

- Oracle, simple case:
- CREATE TABLESPACE tblspname
- DATAFILE 'filename1' SIZE 300G, 'filename2' SIZE 300G, …; -- other files
- Don't need SIZE if file already exists
- These files need to be as contiguous on disk as possible for best performance
- Suggest reinitializing the filesystem before creating the file.
- Alternatively, use "raw partitions", but not for novices.
- Remember a hardware RAID

Tablespaces in other products

- Create tablespace command exists in mysql 5.7, but not our v 5.6.
- For mysql v 5.1-5.6, can only set up the one and only all-inclusive system tablespace at initialization. You can add a file to it later under some conditions.
- DB2 has tablespaces much like Oracle.
- MS Sql Server has "file groups"

Files to Tablespaces on dbs2

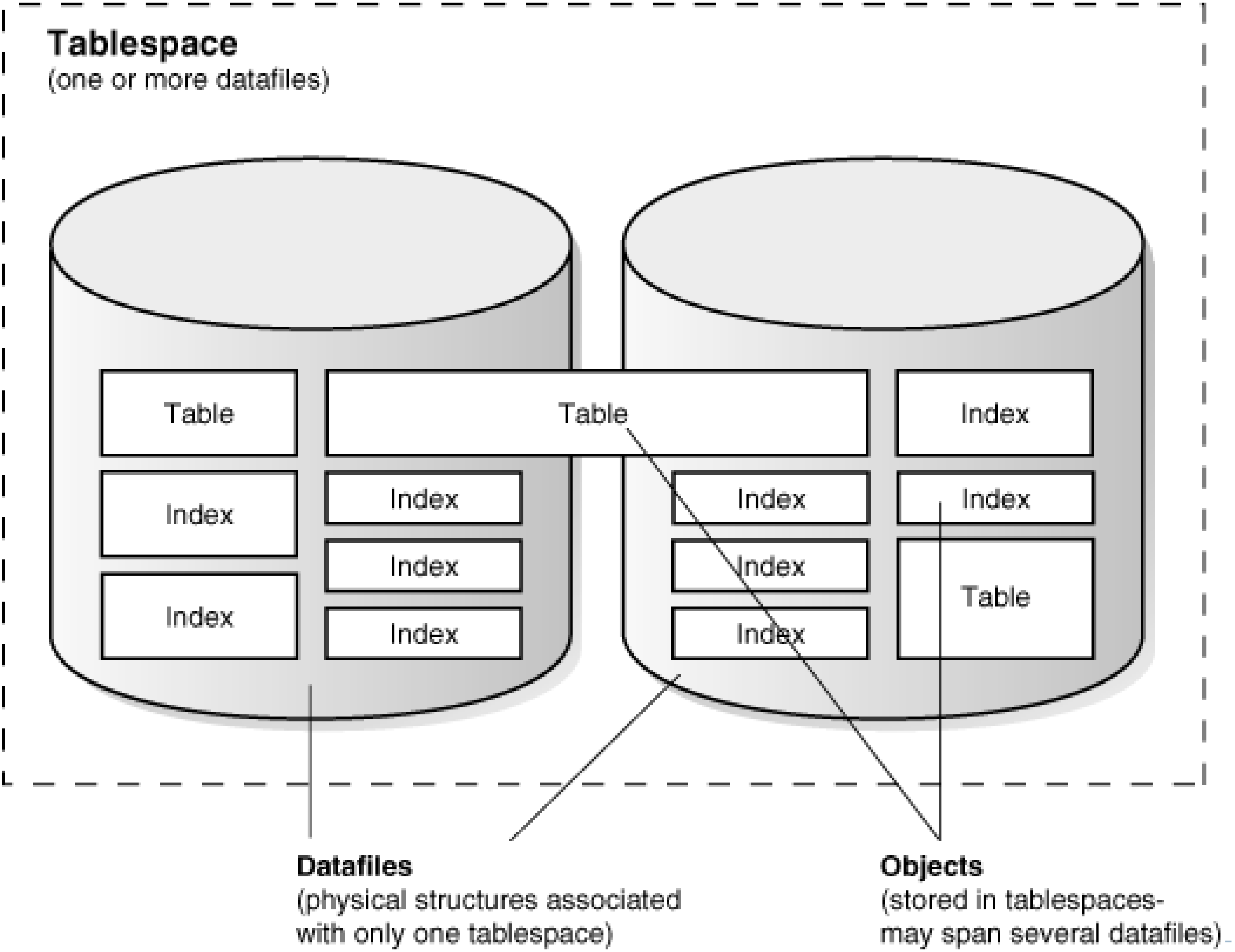
```
SQL> select file_name, tablespace_name, blocks from dba_data_files
FILE_NAME                                         TABLESPACE_NAME      BLOCKS
------------------------------------------------  ---------------      ----------
/disk/sd1d/data/oracle-10.1/dbs2/users01.dbf      USERS                355840
/disk/sd1d/data/oracle-10.1/dbs2/sysaux01.dbf     SYSAUX               133120
/disk/sd1d/data/oracle-10.1/dbs2/undotbs01.dbf    UNDOTBS1             259200
/disk/sd1d/data/oracle-10.1/dbs2/system01.dbf     SYSTEM               64000
/disk/sd1d/data/oracle-10.1/dbs2/caspar.dbf       CASPAR               12800
/disk/sd1e/data/oracle-10.1/dbs2/system02.dbf     S YSTEM              131072
/disk/sd1f/data/oracle-10.1/dbs2/users02.dbf      USERS                524288
/disk/sd1f/data/oracle-10.1/dbs2/undotbs02.dbf    UNDOTBS1             93216
```

- Shows tablespaces SYSTEM (2 files), USERS (2 files), UNDOTBS1 (2 files), SYSAUX (1 file), CASPAR (1 file)

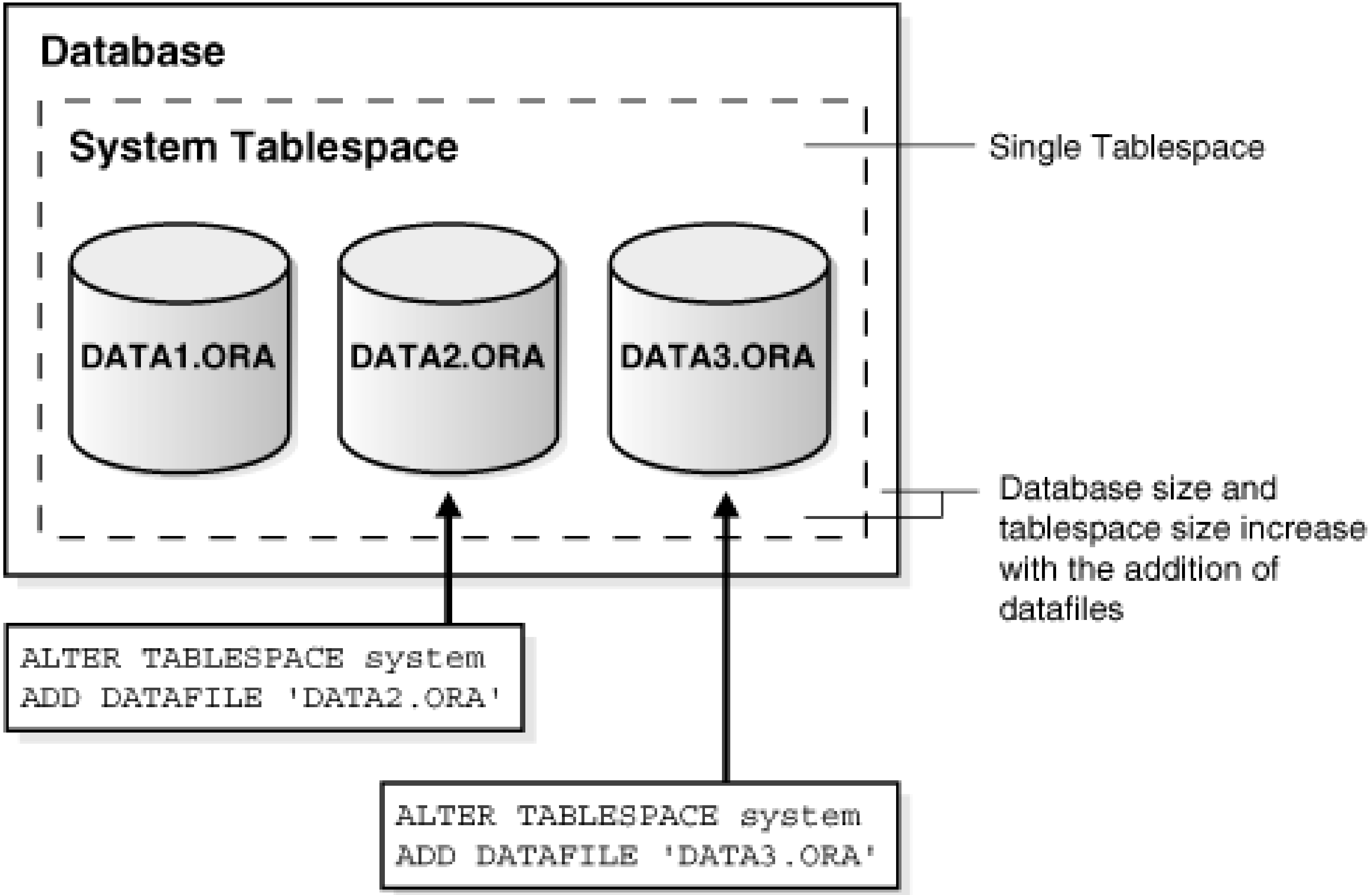
The SYSTEM tablespace

- Every Oracle database contains a tablespace named SYSTEM, which Oracle creates automatically when the database is created.
- The SYSTEM tablespace is always online when the database is open.
- The SYSTEM tablespace always contains the data dictionary tables for the entire database.

From Oracle Docs



Enlarging a Database by Adding a Datafile to a Tablespace



Single Tablespace

Database size and tablespace size increase with the addition of datafiles

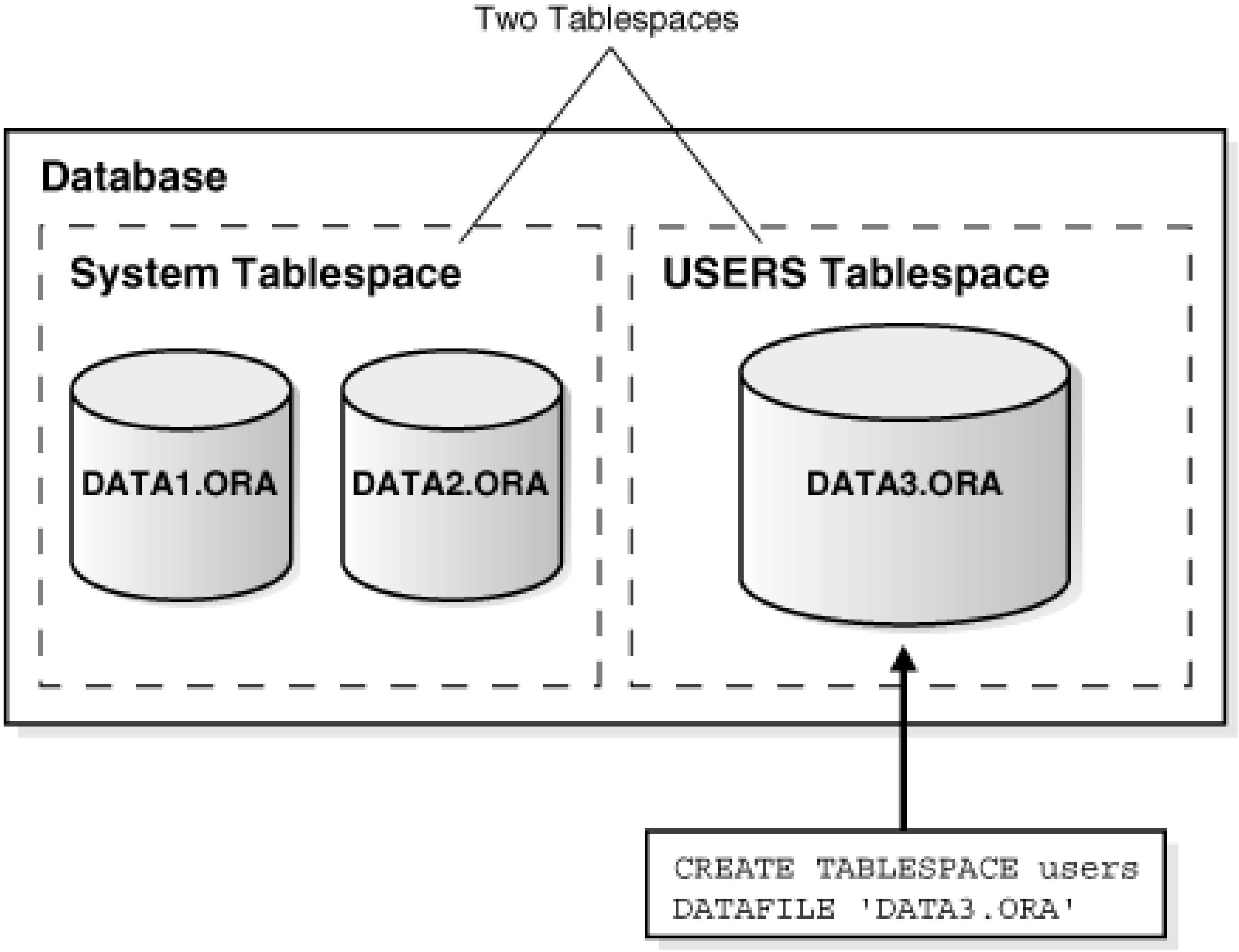
```
ALTER TABLESPACE system
ADD DATAFILE 'DATA2.ORA'
```

```
ALTER TABLESPACE system
ADD DATAFILE 'DATA3.ORA'
```

Example of use of Alter Tablespace

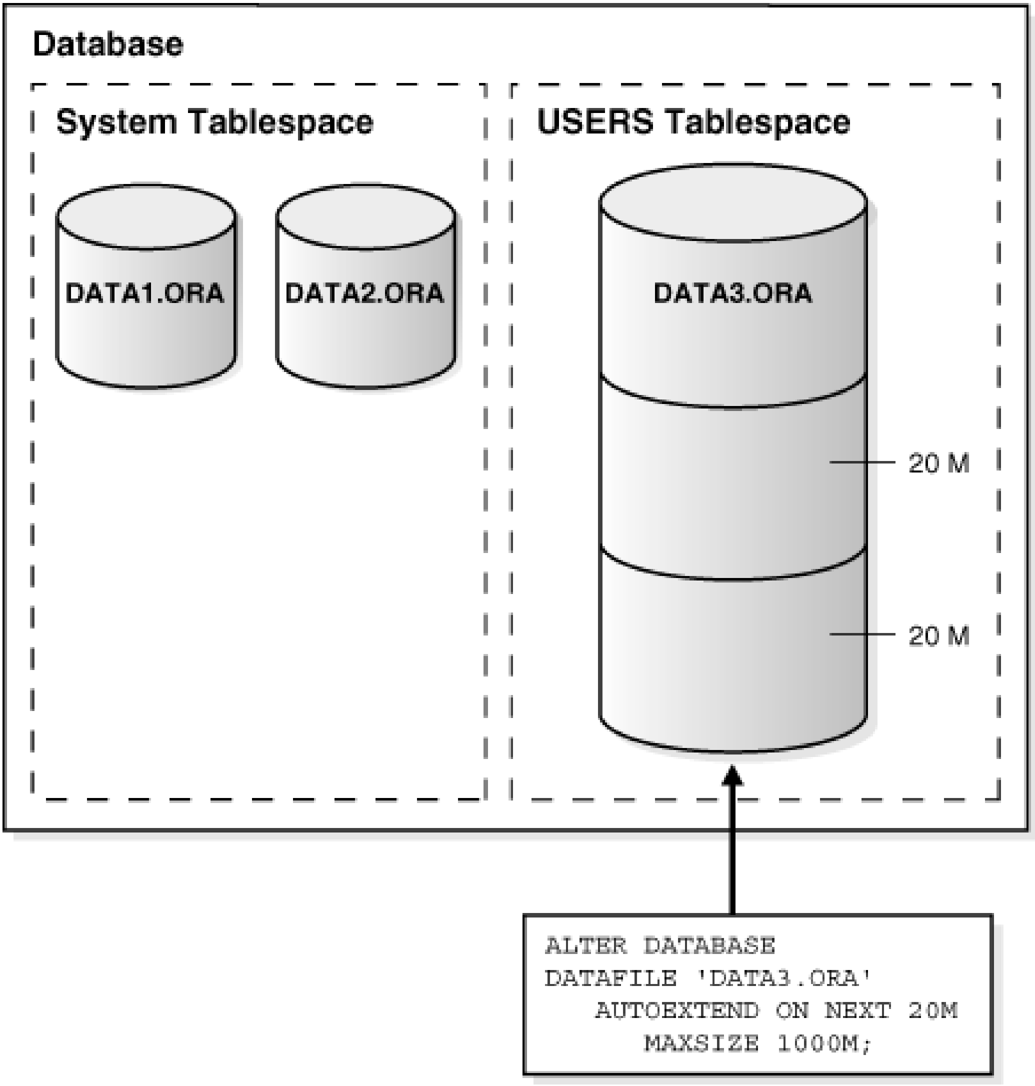
- Command used to expand our USERS tablespace:
- alter tablespace users
-    add datafile '/disk/sd1f/data/oracle-10.1/dbs2/users02.dbf'
-    size 4G;

Enlarging a Database by Adding a New Tablespace



```
CREATE TABLESPACE users
DATAFILE 'DATA3.ORA'
```

Enlarging a Database by Dynamically Sizing Datafiles



```
ALTER DATABASE
DATAFILE 'DATA3.ORA'
   AUTOEXTEND ON NEXT 20M
      MAXSIZE 1000M;
```

Tables and indexes are in a particular tablespace

```
SQL> select table_name, tablespace_name from user_tables;
TABLE_NAME                     TABLESPACE_NAME
------------------------------ ------------------------------
ACCOUNT                        USERS
AGENTS                         USERS
APERF_RESULT                   USERS
…
SQL> select index_name, tablespace_name from user_indexes;
INDEX_NAME                     TABLESPACE_NAME
------------------------------ ------------------------------
BITS1                          USERS
BITS2                          USERS
K100X                          USERS
```

- …   not an accident: account eoneil has default tablespace USERS

Create table can specify tablespace

- CREATE TABLE [schema.]tablename
- (coldef | table_constraint}
- {, coldef | table_constraint, . . .}
- [TABLESPACE tblspname]
- [STORAGE…] ← will cover later today
- [PCTFREE n] [PCTUSED n] ← for pages of table
- [other clauses] ←partitioning support is in here
- [AS subquery]
- This tablespace will override the default for the user
- Create index is similar

PCTFREE and PCTUSED for table

- PCTFREE n, n goes from 0 to 99, default 10.
- PCTUSED n, n goes from 1 to 99, default 40.
- The PCTUSED n clause specifies a condition where if page gets empty enough, inserts will start again!
- Require PCTFREE + PCTUSED < 100, or invalid.
- Example, if PCTFREE 10 PCTUSED 80, then stop inserts when >90% full, start again when <80% full.

Uses of tablespaces: control over disk resources

- In a two-disk system, can use one disk for table, other for index to speed up range searches
- Put table in tablespace USERS, composed of files on one disk, create tablespace USERIND for indexes, composed of file(s) on other disk.
- In a shared system, put one project on high-end disks made into one tablespace using RAID, another project on cheap disks made into another tablespace, also using RAID.
- With RAID, can mix tables and indexes pretty freely.

Block Size (i.e., page size)

- “Oracle recommends smaller Oracle Database block sizes (2 KB or 4 KB) for online transaction processing (OLTP) or mixed workload environments and larger block sizes (8 KB, 16 KB, or 32 KB) for decision support system (DSS) workload environments” from Burleson
- How is this block size specified by the DBA?
- You might expect it to be specified by the tablespace, but it’s more central than that:
- The block size determines the page buffer size in the all-important database page buffer
- So most Oracle installations have a single page size

Multiple page sizes?

- From same page as previous quote
- **WARNING**: *Using multiple blocksizes effectively requires expert-level Oracle skills and an intimate knowledge of your I/O landscape. While deploying multiple blocksizes can greatly reduce I/O and improve response time, it can also wreak havoc in the hands of inexperienced DBA's. Using non-standard blocksizes is not recommended for beginners*
- So we'll assume a single block size
- What is it for dbs2's site?
- It is fixed for each tablespace, so we can find out from dba_tablespaces:

Finding the block size of an Oracle DB

```
SQL> select tablespace_name, block_size from
dba_tablespaces;

TABLESPACE_NAME                BLOCK_SIZE
------------------------------ ----------
SYSTEM                               8192
UNDOTBS1                             8192
SYSAUX                               8192
TEMP                                 8192
USERS                                8192
CASPAR                               8192
```

- So we see it’s 8KB, larger than recommended for OLTP,
- But small for DSS, i.e., a compromise.

Extents of disk in Oracle

An extent is a (hopefully contiguous) part of a file, composed of a whole number of blocks/pages.

One tablespace made of two files:

```
   File1                              File2
|-------------------------------|  |--------------------------|
  |----|        |--|                 |-----|                |---|
  extents for one table
```

Note extents can be of different sizes—by default they get bigger and bigger as the table grows.

Goal: less seeking because lots of related data is close by on disk

STORAGE clause of Create Table

[STORAGE ([INITIAL n [K|M|G]] [NEXT n [K|M|G]] [MINEXTENTS n] [PCTINCREASE n])]

INITIAL n: size in bytes of initial extent (default 5 pages)

NEXT n: size in bytes of next extent (default 5 pages)

PCTINCREASE n: increase from one extent to next, starting from third one. (default 50%)

- MINEXTENTS n: start at creation with this number of extents; used when know initial use will be very large

DEFAULT STORAGE clause of Create Tablespace

[DEFAULT STORAGE ([INITIAL n [K|M|G]] [NEXT n [K|M|G]] [MINEXTENTS n] [PCTINCREASE n])]

- Sets defaults for create table and create index in that tablespace
- Example: tablespace for warehouse tables should have larger extents by default
- DEFAULT STORAGE (INITIAL 10M NEXT 10M)
- Downside: a little side table takes 10M
- But 10M in a warehouse is trivial.

Other Database Files

- So far, considered the files holding pages of data for tables and indexes
- Other important files: saw redo*.dbf, undotbs01.dbf
- Redo log files: information that allows for crash recovery
  - The current such file is appended to constantly as the DB is changed, read only in crash recovery
  - The system cuts over to another of these files periodically
  - For a serious database, should be mirrored, since otherwise is a single point of failure
- Undo tablespace: information that allows for rollbacks and also snapshots for efficient reads
  - This data is written and read, more like the DB data, so held in a tablespace, unlike the redo log

RAID and Oracle, from Burleson

- RAID 5: slow for updates, but in wide use for safety
- Mirroring/shadowing: Great for redo log file

RAID	**Type of Raid**	**Control File**	**Database File**	**Redo Log File**	**Archive Log File**
0	Striping	Avoid	OK	Avoid	Avoid
1	Shadowing	Best	OK	Best	Best
1+0	Striping and Shadowing	OK	Best	Avoid	Avoid
3	Striping with static parity	OK	OK	Avoid	Avoid
5	Striping with rotating parity	OK	Best if RAID0-1 not available	Avoid	Avoid

Example:1TB Database with 2000 ops/s

- Burleson says: ***Size first for IO capacity, then for volume.***
- 2000 ops/sec means 20 7200 rpm disks or 10 15Krpm disks, roughly, not counting parity disks or mirrors or spares
- So say 12 15Krpm disks in a RAID1+0, plus 12 mirrors for data
- 2 disks for mirrored log, RAID 1, plus 5 spares.
  - Smart RAID controller with memory cache best here
- 1TB/12 = 83 GB, so 143GB disks are fine for data.

1TB example

- Build RAID for data
  - End up with new empty filesystem /disk/raida
- Build RAID for redo log
  - End up with new empty filesystem /disk/raidb
- Create tablespace DBDATA and let Oracle create one huge file /disk/raida/dbdata.dbf
- Change database to use redo logs on /disk/raidb:
  - alter database add logfile group 5 (‘/disk/raid/redo05a.log',
  - ‘/disk/raid/redo05b.log') size 500m;
- Create tables and indexes in tablespace DBDATA

Oracle Project Account

- Create an Oracle account for the project, and make its default tablespace be DBDATA

```
create user myproject identified by pw default tablespace dbdata
    temporary tablespace temp;
```

- This simplifies the createdb.sql, etc.
- Makes it less likely that someone accidentally makes a table in tablespace USERS for the project, off on wrong disks.
- Make a project rule that DBA actions are done as this user
- If user already exists:

```
alter user myproject default tablespace dbdata;
```

Summary

- Hierarchy of data containers:
- Files containing blocks/pages 8KB each on dbs2
- Tablespace: some number of files ganged together
- Extent: some number of blocks in a certain file and thus in a certain tablespace, by default, bigger and bigger as a table grows
- Table or Index: some number of extents all in the same tablespace
- Separately: redo log file, no page structure, just append records describing DB changes.