Spring 2015 CS157B: Oracle Lab
Instructor: Dr. Kim

Contents

I. SGA and Granule

From Wiki:

A **granule** is a unit of contiguous (adjacent/bordering) virtual memory allocated to a process. In the Oracle DBMS, the Oracle server allocates the system global area (SGA) in granule units at the time of instance startup. The granule size depends on the database version and sometimes on the operating system. In Oracle 11g, it is typically 4 MB if the SGA size is less than 1 GB, and 16 MB otherwise. There must be at least 3 granules in the SGA: one for the Database Buffer Cache, one for the Shared Pool Area and one for the Redo Log Buffer.

Exercise I-1: It is possible to retrieve information about the current granule size at any time by querying the **dynamic view V$SGAINFO**.

```
SQL> select * from V$SGAINFO;

NAME                                  BYTES RES
-------------------------------- ---------- ---
Fixed SGA Size                      2217464 No
Redo Buffers                        6832128 No
Buffer Cache Size                 247463936 Yes
Shared Pool Size                  201326592 Yes
Large Pool Size                     4194304 Yes
Java Pool Size                      4194304 Yes
Streams Pool Size                         0 Yes
Shared IO Pool Size                       0 Yes
Granule Size                        4194304 No
Maximum SGA Size                  784998400 No
Startup overhead in Shared Pool    75497472 No


NAME                                  BYTES RES
-------------------------------- ---------- ---
Free SGA Memory Available         318767104
```

Exercise I-2: Granule is the unit of memory allocation in Oracle.

```
SQL> show parameter shared_pool_size;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
shared_pool_size                     big integer 0
```

Note: The current size of 0 due to the ASMM (Automatic Shared Memory Management) function (Oracle 10g new feature)

```
SQL> alter system set shared_pool_size=10M;

System altered.

SQL> show parameter shared_pool_size;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
shared_pool_size                     big integer 12M

SQL> alter system set shared_pool_size= 9M;

System altered.

SQL> show parameter shared_pool_size;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
shared_pool_size                     big integer 12M

SQL> alter system set shared_pool_size= 8M;

System altered.

SQL>  show parameter shared_pool_size;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
shared_pool_size                     big integer 8M
```

II. Oracle Startup and Shutdown

Exercise II-1: To startup and shutdown Oracle instance

```
[oracle@cs72 ~]$ sqlplus / as sysdba

SQL*Plus: Release 11.2.0.1.0 Production on Thu Apr 3 13:50:58 2014
Copyright (c) 1982, 2009, Oracle.  All rights reserved.
Connected to an idle instance.
```

```
SQL> startup
ORACLE instance started.

Total System Global Area  784998400 bytes
Fixed Size                  2217464 bytes
Variable Size             528484872 bytes
Database Buffers          247463936 bytes
Redo Buffers                6832128 bytes
Database mounted.
Database opened.

SQL> shutdown immediate
Database closed.
Database dismounted.
ORACLE instance shut down.
```

Note: When you are in the sqlplus command line but want to check something on the shell command line you can use !

```
SQL> !
```

Exercise II-2: Without initialization parameter file (pfile or spfile), instance startup fails.

Step 1: Move spfile from $ORACLE_HOME/dbs to /tmp so that it can't be found from the original location. Note: Since Oracle 9i, spfileSID.ora is created by default.

```
[oracle@cs72 ~]$ cd $ORACLE_HOME/dbs/
[oracle@cs72 dbs]$ ls
hc_ORCL.dat  init.ora  lkORCL  orapwORCL  spfileORCL.ora
[oracle@cs72 dbs]$ mv spfileORCL.ora /tmp/
[oracle@cs72 dbs]$ exit
exit

SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
```

Step 2: Start up the instance again. Instance startup will fail.

```
SQL> startup
ORA-01078: failure in processing system parameters
LRM-00109: could not open parameter file
'/apps/oracle/product/11.2.0/dbhome_1/dbs/initORCL.ora'
```

Step 3: Copy spfile back to its original location. Now, instance start up will succeed.

```
SQL> !
[oracle@cs72 ~]$ cp /tmp/spfileORCL.ora $ORACLE_HOME/dbs/
[oracle@cs72 ~]$ exit
exit
```

```
SQL> startup
ORACLE instance started.

Total System Global Area  784998400 bytes
Fixed Size                  2217464 bytes
Variable Size             528484872 bytes
Database Buffers          247463936 bytes
Redo Buffers                6832128 bytes
Database mounted.
Database opened.
```

Exercise II-3: To check if Oracle uses spfil or pfile.

```
SQL> show parameter spfile;

NAME         TYPE         VALUE
---------------------------------------------------------------------------------
spfile          string      /apps/oracle/product/11.2.0/dbhome_1/dbs/spfileORCL.ora

SQL> show parameter pfile;

NAME        TYPE         VALUE
---------------------------------------------------------------------------------
spfile          string      /apps/oracle/product/11.2.0/dbhome_1/dbs/spfileORCL.ora

spfile is being used.
```

Exercise II-4: Statup options

```
SQL> startup nomount;
ORACLE instance started.

Total System Global Area  784998400 bytes
Fixed Size                  2217464 bytes
Variable Size             528484872 bytes
Database Buffers          247463936 bytes
Redo Buffers                6832128 bytes
SQL> select status from v$instance;

STATUS
------------
STARTED ← no mount state

SQL> alter database mount;
Database altered.

SQL> select status from v$instance;
STATUS
------------
MOUNTED

SQL> alter database open;
```

```
Database altered.
```

Exercise II-5: Shutdown doesn't have mount or nomount option.

```
SQL> shutdown mount;
SP2-0717: illegal SHUTDOWN option
SQL> shutdown nomount;
SP2-0717: illegal SHUTDOWN option
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
```

III. Managing Control Files

Quoted from Oracle Database Concepts:

- Every Oracle Database has a control file, which is a small binary file that records the physical structure of the database. The control file includes:
  - The database name
  - Names and locations of associated data files and redo log files
  - The timestamp of the database creation
  - The current log sequence number
  - Checkpoint information
- The control file must be available for writing by the Oracle Database server whenever the database is open. Without the control file, the database cannot be mounted and recovery is difficult.
- The control file of an Oracle Database is created at the same time as the database. By default, at least one copy of the control file is created during database creation. On some operating systems the default is to create multiple copies. You should create two or more copies of the control file during database creation.
- Location of Control file is recorded in the parameter file (e.g. spfile) . A server process reads this location information from the parameter file and locates control files.

Exercise III-1: To multiplex control files (when ORACLE uses Spfile)

```
SQL> startup;
SQL> select status from v$instance;

STATUS
------------
OPEN ← current state


SQL> show parameter spfile;

NAME        TYPE        VALUE
------------------------------------------------------------------------------
spfile      string      /apps/oracle/product/11.2.0/dbhome_1/dbs/spfileORCL.ora
```

Step 1: Check the location of control files

```
SQL> select name from v$controlfile;

NAME
--------------------------------------------------------------------------------
/apps/oracle/oradata/ORCL/control01.ctl
/apps/oracle/flash_recovery_area/ORCL/control02.ctl
```

There are control files. now: control files do exist

Step 2: Change the location of control files (`control_files`) in spfile content and shutdown the instance

```
SQL> alter system set control_files ='/apps/oracle/disk1/control01.ctl',
  2  '/apps/oracle/disk2/control02.ctl' scope = spfile;

System altered.

SQL > shutdown immediate;
ORACLE instance shut down.
```

Step 3: Create directories disk1 and disk2 under /apps/oracle as you promised in spfile as a new location of control files. Copy control files to the directories you created.

```
SQL> !
[oracle@cs72 ~]$ cd /apps/oracle
[oracle@cs72 ~]$ mkdir disk1 disk2

[oracle@cs72 ~]$ ~~cp /apps/oracle/~~oradata/ORCL/control01.ctl
   /apps/oracle/disk1/control01.ctl
[oracle@cs72 ~]$ cp /apps/~~oracle/~~oradata/ORCL/control01.ctl /apps/oracle/disk2/c
ontrol02.ctl
[oracle@cs72 ~]$ exit
exit
```

cp /apps/oracle/app/oracle/oradata/ORCL/control01.ctl /apps/oracle/disk1/

```
SQL> startup
ORACLE instance started.

Total System Global Area  784998400 bytes
Fixed Size                  2217464 bytes
Variable Size             528484872 bytes
Database Buffers          247463936 bytes
Redo Buffers                6832128 bytes
Database mounted.
Database opened.
SQL>

SQL> select name from v$controlfile;

NAME
--------------------------------------------------------------------------------
/apps/oracle/disk1/control01.ctl
/apps/oracle/disk2/control02.ctl
```

Step 4: Make sure to change control_files back to the original value after this exercise.

```
SQL> !cp /apps/oracle/disk1/control01.ctl /apps/oracle/oradata/ORCL/control01.ctl
SQL> !cp /apps/oracle/disk2/control02.ctl/~~apps/oracle~~/flash_recovery_area/ORCL/control02.ctl
                                       ↑  /apps/oracle/app/oracle/

SQL> alter system set control_files ='/apps/oracle/oradata/ORCL/control01.ctl',
  2  '/apps/oracle/flash_recovery_area/ORCL/control02.ctl' scope=spfile;

     System altered.
```

Summary of this exercise: Steps to multiplex control files

- Modify Spfil (control_files)
- Shutdown Instance
- Copy control files
- Start Instances

IV. Managing Redo log

We learned redo records, redo log buffer, and on line redo files. In this exercise, we will practice how to create groups and members of redo log files.

- Data modification is done through DDL, DML(Insert/Delete/Update), or TCL (Transaction Control Language, e.g, COMMIT, ROLLBACK, SAVE POINT, SET TRANSACTION)
- Two important mechanisms: Write Log Ahead and Log Force at Commit

Main Commands to practice in this part of lab are:

- Creating a new group
- Adding member to a group
- Deleting member
- Deleting group
- Forcing log switching
- Forcing checkpoint

Three states of Red Log File

- CURRENT: LGWR is currently writing on it.
- ACTIVE: active means "we need it for crash recovery" if the checkpoint signaled by the switch log file has completed we no longer need that log file for crash recovery.
- Inactive: we don't need it for crash recovery

Note:

- When a group has only one member, the member can't be dropped. You need to drop the group that contains the member.
- If Oracle installation wasn't done in ASM (in our case), dropping group and member will not physically remove the corresponding file from the disk. A DBM has to manually delete it.

Exercise IV-1

Step 1: Prepare a sql script we are going to frequently use to check the log files.

```
SQL> !vi log.sql
set line 200
col group# for 999
col mb for 999
col member for a45
col seq# for 999
col status for a8
col arc for a5

SELECT a.group#,a.member,b.bytes/1024/1024 MB,b.sequence# "SEQ#",b.status,b.arch
ived "ARC"
FROM v$logfile a, v$log b
WHERE a.group#=b.group#
ORDER BY 1,2;
/
:wq!
```

Check the status of log files. SEQ# and Status may vary. ARC indicates the log file was archived or not.

```
SQL> @log

GROUP# MEMBER                                          MB SEQ# STATUS   ARC
------ --------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log            50    4 INACTIVE NO
     2 /apps/oracle/oradata/ORCL/redo02.log            50    5 CURRENT  NO
     3 /apps/oracle/oradata/ORCL/redo03.log            50    3 INACTIVE NO
```

Step 2: To add a new group (group number 4) /member

The following will create a new group and one member inside of the group and add the group to the database.

```
SQL> alter database add logfile group 4
  2  '/apps/oracle/oradata/ORCL/redo04_a.log' size 5M;

Database altered.
SQL> @log

GROUP# MEMBER                                          MB SEQ# STATUS   ARC
------ --------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log            50    4 INACTIVE NO
     2 /apps/oracle/oradata/ORCL/redo02.log            50    5 CURRENT  NO
     3 /apps/oracle/oradata/ORCL/redo03.log            50    3 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log           5    0 UNUSED   YES
```

Let's add one more member to the group 4.

```
SQL> alter database add logfile member
```

```
  2  '/apps/oracle/oradata/ORCL/redo04_b.log' to group 4;

Database altered.

SQL> @log

GROUP# MEMBER                                              MB SEQ# STATUS   ARC
------ -------------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log                50    4 INACTIVE NO
     2 /apps/oracle/oradata/ORCL/redo02.log                50    5 CURRENT  NO
     3 /apps/oracle/oradata/ORCL/redo03.log                50    3 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log               5    0 UNUSED   YES
     4 /apps/oracle/oradata/ORCL/redo04_b.log               5    0 UNUSED   YES


SQL> alter system switch logfile;

System altered.

SQL> @log

GROUP# MEMBER                                              MB SEQ# STATUS   ARC
------ -------------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log                50    4 INACTIVE NO
     2 /apps/oracle/oradata/ORCL/redo02.log                50    5 ACTIVE   NO
     3 /apps/oracle/oradata/ORCL/redo03.log                50    3 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log               5    6 CURRENT  NO
     4 /apps/oracle/oradata/ORCL/redo04_b.log               5    6 CURRENT  NO
```

The following command will change the status of ACTIVE of group 2 to INACTIVE.

```
SQL> alter system checkpoint;

System altered.

SQL> @log

GROUP# MEMBER                                              MB SEQ# STATUS   ARC
------ -------------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log                50    4 INACTIVE NO
     2 /apps/oracle/oradata/ORCL/redo02.log                50    5 INACTIVE NO
     3 /apps/oracle/oradata/ORCL/redo03.log                50    3 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log               5    6 CURRENT  NO
     4 /apps/oracle/oradata/ORCL/redo04_b.log               5    6 CURRENT  NO
```

Step 3: To drop existing member/group

```
SQL> alter system switch logfile;

System altered.

SQL> @log

GROUP# MEMBER                                              MB SEQ# STATUS   ARC
------ -------------------------------------------------- ---- ---- -------- -----
```

```
     1 /apps/oracle/oradata/ORCL/redo01.log           50     8 INACTIVE NO
     2 /apps/oracle/oradata/ORCL/redo02.log           50     5 INACTIVE NO
     3 /apps/oracle/oradata/ORCL/redo03.log           50     7 CURRENT  NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log          5     6 ACTIVE   NO
     4 /apps/oracle/oradata/ORCL/redo04_b.log          5     6 ACTIVE   NO

SQL> alter system checkpoint;

System altered.

SQL> @log

GROUP# MEMBER                                          MB SEQ# STATUS   ARC
------ --------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log           50     8 CURRENT  NO
     2 /apps/oracle/oradata/ORCL/redo02.log           50     5 INACTIVE NO
     3 /apps/oracle/oradata/ORCL/redo03.log           50     7 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log          5     6 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_b.log          5     6 INACTIVE NO

SQL> alter database drop logfile member '/apps/oracle/oradata/ORCL/redo04_b.log';

Database altered.

SQL> @log

GROUP# MEMBER                                          MB SEQ# STATUS   ARC
------ --------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log           50     8 CURRENT  NO
     2 /apps/oracle/oradata/ORCL/redo02.log           50     5 INACTIVE NO
     3 /apps/oracle/oradata/ORCL/redo03.log           50     7 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log          5     6 INACTIVE NO

SQL> !ls /apps/oracle/oradata/ORCL/redo04_b.log
/apps/oracle/oradata/ORCL/redo04_b.log ← physical copy still exist after dropping it!

SQL> !rm /apps/oracle/oradata/ORCL/redo04_b.log ← have to manually remove it.


SQL> !ls /apps/oracle/oradata/ORCL/redo04_b.log
ls: cannot access /apps/oracle/oradata/ORCL/redo04_b.log: No such file or directory

SQL> @log

GROUP# MEMBER                                          MB SEQ# STATUS   ARC
------ --------------------------------------------- ---- ---- -------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log           50     8 CURRENT  NO
     2 /apps/oracle/oradata/ORCL/redo02.log           50     5 INACTIVE NO
     3 /apps/oracle/oradata/ORCL/redo03.log           50     7 INACTIVE NO
     4 /apps/oracle/oradata/ORCL/redo04_a.log          5     6 INACTIVE NO ← only
  one member in the group 4

SQL> alter database drop logfile member '/apps/oracle/oradata/ORCL/redo04_a.log'
  2  ;
alter database drop logfile member '/apps/oracle/oradata/ORCL/redo04_a.log'
*
ERROR at line 1:
```

```
ORA-00361: cannot remove last log member /apps/oracle/oradata/ORCL/redo04_a.log for
   group
```

Note that you can't remove the last member. Have to remove the group itself.

```
SQL> alter database drop logfile group 4;

Database altered.

SQL> @log

GROUP# MEMBER                                             MB  SEQ# STATUS    ARC
------ ---------------------------------------------- ---- ---- --------- -----
     1 /apps/oracle/oradata/ORCL/redo01.log             50     8 CURRENT   NO
     2 /apps/oracle/oradata/ORCL/redo02.log             50     5 INACTIVE  NO
     3 /apps/oracle/oradata/ORCL/redo03.log             50     7 INACTIVE  NO


SQL> !rm -fr /apps/oracle/oradata/ORCL/redo04_a.log ← manually remove the physical
   file.

SQL> !ls /apps/oracle/oradata/ORCL/redo04_a.log
ls: cannot access /apps/oracle/oradata/ORCL/redo04_a.log: No such file or directory
```

V. Table spaces and Data Files management

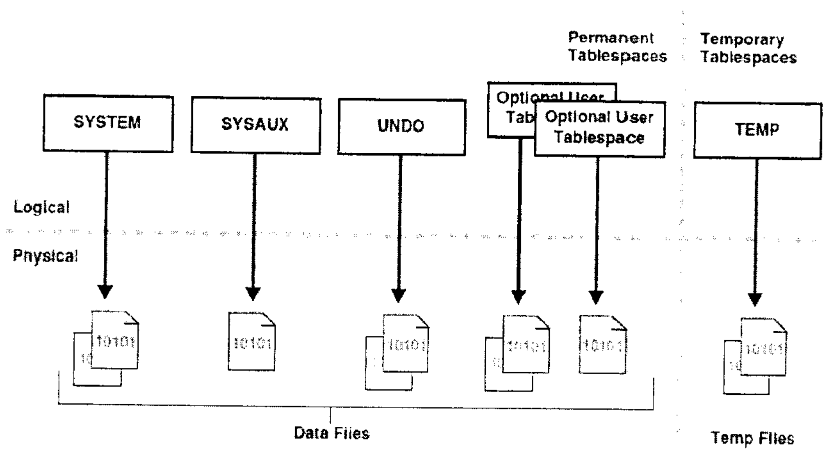
- Reminder: Oracle separate the logical view of storage (oracle data blocks, extends, segments, and tablespaces, segments) from the physical view of storage (operating system blocks and data files).

- Why Tablespaces? (Quoted from Expert Oracle Database 11g Administration pp. 171)

  - Table spaces make it easier to allocate space quotas to various users in the database.
  - Table spaces enable you to perform partial backups and recoveries based on the table space as a unit.
  - Because a large object like a data warehouse portioned table can be spread over several table spaces, you can increase performance by spanning the table space over several disks and controllers
  - You can take a table space offline without having to bring down the entire database.
  - Table spaces are an easy way to allocate database space.
  - You can import or export specific application data by using the import and export utilities at the table space level.

- Types of tablespace

Figure 12-27 Tablespaces



A permanent tablespace groups persistent schema objects. The segments for objects in the tablespace are stored physically in data files. You will learn about SYSTEM table space and UNDO table space before starting exercises.

(1) SYSTEM table space

- It stores
    - Data dictionary
    - Tables and views that contain administrative information about the database
    - Compiled stored objects such as triggers, procedures, and packages
- Caution: With damaged System table space, the Oracle Instance cannot start!
- SYSTEM table space is owned by the SYS account. Notice that the SYS user can look up the dictionary but cannot modify the table content.
- To check the number of dictionaries stored in the SYSTEM table space.

```
SQL> select count(*) from dictionary;

  COUNT(*)
----------
      2553
```

- Category of Dictionaries
    - Base Table (can't access it, even DBA can't)
        - These underlying tables store information about the database.
        - Only Oracle Database should write to and read these tables.
        - Users, even DBA, rarely access the base tables directly because they are normalized and most data is stored in a cryptic format.
    - Data Dictionary View – (a) Static Data Dictionary Views and (b) Dynamic Performance Views: These views decode the base table data into useful information.

- Data Dictionary View Details

(a) Static Data Dictionary Views: can be accessed in INSTANCE OPEN mode.

These tables and views are called static, because they change only when a change is made to the data dictionary (for example, when a new table is created or when a user is granted new privileges).

Table 6–1 Data Dictionary View Sets

Prefix	User Access	Contents	Notes
DBA_	Database administrators	All objects	Some DBA_ views have additional columns containing information useful to the administrator.
ALL_	All users	Objects to which user has privileges	Includes objects owned by user. These views obey the current set of enabled roles.
USER_	All users	Objects owned by user	Views with the prefix USER_ usually exclude the column OWNER. This column is implied in the USER_ views to be the user issuing the query.

Examples:
DBA_TABLES, DBA_INDEXES
ALL_TABLES, ALL_INDEXES
USER_TABLES, USER_INDEXES

(b) Dynamic Performance Views: can be accessed from INSTANCE MOUNT mode.

These views are called dynamic performance views because they are continuously updated while a database is open and in use, and their contents relate primarily to performance.

Example: v$XXXX

Caution: Altering or manipulating the data in data dictionary tables can permanently and detrimentally affect database operation.

(2) UNDO table space

Terminology:

- Undo data: old image of data
- Undo segment: specialized segment that contains undo data
- Undo table space: table space that stores undo segments

Usage of undo table space

1. Transaction rollback
2. Instance Recovery
3. Read Consistency: Uncommitted update by session A will not be seen by session B. (even in different sessions opened by the same user).

We learned the first and second usage of undo table space in class. The 3rd usage is explained with an example. You don't have to submit this example exercise.

Step 1: Let's open a session A

```
[oracle@cs72 ~]$ sqlplus scott/tiger
SQL> create table test(name varchar(20));
Table created.

SQL> insert into test(name) values ('Jane');
1 row created.

SQL> select * from test;

NAME
--------------------
Jane
```

Note that this update is not committed.

Step 2: Let's open another sessionB by scott/tiger in a different window.

```
[oracle@cs72 ~]$ sqlplus scott/tiger
```

```
SQL> select * from test;

no rows selected
```

Session B reads from undo segment which stores before image of the update. The before image is nothing but an empty table.

Step 3: Now, in session A, commit the update.

```
SQL> commit;
Commit complete.
```

Step 4: In session B,

```
SQL> select * from test;

NAME
--------------------
Jane
```

Exercise V-1: This exercise is to create a table and see if the table is listed in static data dictionary views (e.g. user_tables and all_tables). In this exercise, we are going to login in as 'scott'. Notice that the scott user is already comes with Oracle.

Step 1: Login as `sysdba` and set the password of `scott` to `tiger`

```
[oracle@cs72 ~]$ sqlplus / as sysdba

SQL> select * from all_users;
```

Users are displayed including scott.

```
SQL> alter user scott identified by tiger
User altered.
```

Step 2: Login as scott using the password tiger. If the system indicates the account is locked, unlock it as sysdba.

```
    [oracle@cs72 ~]$ sqlplus scott/tiger
    The account is locked.
    SQL> exit

    [oracle@cs72 ~]$ sqlplus / as sysdba

    SQL> alter user scott account unlock;
    User altered.
    SQL> exit
```

Step 3: Now connect to Oracle again as scott.

```
[oracle@cs72 ~]$ sqlplus scott/tiger
```

Step 4: Create a table named stest and populate the table using a loop in PL/SQL.

```
SQL> create table stest (no number);

Table created.

SQL> begin
  2  for i in 1..100 loop
  3  insert into stest values (i);
  4  end loop;
  5  commit;
  6  end;
  7  /

PL/SQL procedure successfully completed.

SQL>  select count(*) from stest;

  COUNT(*)
----------
       100
```

Step 5: Look up ALL_TABLES and USER_TABLES

- ALL_TABLES and USER_TABLES are examples of static data dictionary views. You can find more information about Oracle metadata from http://en.wikipedia.org/wiki/Oracle_metadata.
- ALL_TABLE shows a list of all tables in the current database that are accessible to the current user
- USER_TABLES shows only the tables owned by the current user, in our case, scott.

```
SQL> select table_name, tablespace_name from user_tables where table_name ='STEST';

TABLE_NAME                     TABLESPACE_NAME
------------------------------ ------------------------------
STEST                          USERS


SQL> select table_name, tablespace_name from all_tables where owner ='SCOTT';

TABLE_NAME                     TABLESPACE_NAME
------------------------------ ------------------------------
DEPT                           USERS
EMP                            USERS
SALGRADE                       USERS
STEST                          USERS
ADMISSION                      COLLEGE
TEST                           USERS
BONUS                          USERS
```

Note: In the above select statements, SCOTT should be in capital letter.

Exercise V-2: In this exercise, we will create our own table space as a DBA. DBA can create and drop a table space as needed. Through this exercise, you will also learn that a table space (logical storage structure) may contain multiple physical storage structures (data files).

Step 1: Login as sysdba and create a table space called college. In the tablespace college, create a datafile named college01.dbf with size 1M.

```
[oracle@cs72 ~]$ sqlplus / as sysdba
SQL> create tablespace college
  2  datafile '/apps/oracle/oradata/ORCL/college01.dbf' size 1M;


Tablespace created.
```

Step 2: Lookup the college table space.

```
SQL>  select tablespace_name, status, contents, extent_management,
segment_space_management
  2  from dba_tablespaces;

TABLESPACE_NAME                 STATUS    CONTENTS  EXTENT_MAN SEGMEN
------------------------------ --------- --------- ---------- ------
SYSTEM                          ONLINE    PERMANENT LOCAL      MANUAL
SYSAUX                          ONLINE    PERMANENT LOCAL      AUTO
UNDOTBS1                        ONLINE    UNDO      LOCAL      MANUAL
TEMP                            ONLINE    TEMPORARY LOCAL      MANUAL
USERS                           ONLINE    PERMANENT LOCAL      AUTO
EXAMPLE                         ONLINE    PERMANENT LOCAL      AUTO
COLLEGE                         ONLINE    PERMANENT LOCAL      AUTO

7 rows selected.
```

Step 3: Check database file information.

```
SQL> select tablespace_name, bytes/1024/1024 MB, file_name
  2  from dba_data_files;

TABLESPACE_NAME         MB     FILE_NAME
--------------------------------------------------------------------------------
USERS                   5     /apps/oracle/oradata/ORCL/users01.dbf
UNDOTBS1                50    /apps/oracle/oradata/ORCL/undotbs01.dbf
SYSAUX                  500   /apps/oracle/oradata/ORCL/sysaux01.dbf
SYSTEM                  680   /apps/oracle/oradata/ORCL/system01.dbf
EXAMPLE                 100   /apps/oracle/oradata/ORCL/example01.dbf
COLLEGE                 1     /apps/oracle/oradata/ORCL/college01.dbf

6 rows selected.
```

Step 4: Check the actual usage of each data file. Here I would like to show you how to write a sql script and reuse it as needed. The soft copy of `usage.sql` is available on the course material site under oracle lab manual.

```
SQL> !vi usage.sql
set line 200;
col file# for 999;
col ts_name for a10;
col total_blocks for 9999999;
col used_blocks for 999999;
col pct_used for a10
select distinct d.file_id       file#, ← must be a tab
d.tablespace_name       ts_name,
d.bytes /1024 / 1024    MB,
d.bytes /8192   total_blocks,
sum(e.blocks)   used_blocks,
to_char(nvl(round(sum(e.blocks)/(d.bytes/8192),4),0)*100,'09.99') || ' %' pct_used
from dba_extents e, dba_data_files d
where d.file_id = e.file_id(+)
group by d.file_id, d.tablespace_name, d.bytes
order by 1, 2;


SQL> @usage ←This will take some time to run

FILE# TS_NAME             MB TOTAL_BLOCKS USED_BLOCKS PCT_USED
----- ---------- ---------- ------------ ----------- ----------
    1 SYSTEM            680        87040       86008  98.81 %
    2 SYSAUX            500        64000       60264  94.16 %
    3 UNDOTBS1           50         6400        1512  23.63 %
    4 USERS               5          640         400  62.50 %
    5 EXAMPLE           100        12800        9912  77.44 %
    6 COLLEGE             1          128              00.00 %

6 rows selected.
```

Exercise V-2: In this exercise, we will learn how to manage a table space. A table will be created in a table space. After generating more data than a data file can accommodate, we will learn how to solve the problem through various approach.

Step 1: Create a table called admission which belongs to scott user in tablespace college

```
[oracle@cs72 ~]$ sqlplus / as sysdba
SQL> create table scott.admission (studno number) tablespace college;
Table created.
```

Step 2: Populate the table with 50000 values.

```
SQL> begin
  2 for i in 1..50000 loop
  3 insert into scott.admission values(i);
```

```
4 end loop;
5 commit;
6 end;
7 /
```

PL/SQL procedure successfully completed.

Step 3: Run the above code one more time so that the number of data exceeds the capacity of datafile holding data.

```
SQL> /
begin
*
ERROR at line 1:
ORA-01653: unable to extend table SCOTT.ADMISSION by 8 in tablespace COLLEGE
ORA-06512: at line 3

/// Error due to not enough storage
```

Now we will examine three alternative solutions for this problem.

Solution 1: Add more data file to the table space

```
SQL> alter tablespace college
  2  add datafile '/apps/oracle/oradata/ORCL/college02.dbf' size 20M;

Tablespace altered.

SQL>!vi check_files.sql

select tablespace_name, bytes/1024/1024 MB, file_name from dba_data_files

SQL> @check_files

TABLESPACE_NAME        MB     FILE_NAME
-------------------------------------------------------------------------------
USERS                  5      /apps/oracle/oradata/ORCL/users01.dbf
UNDOTBS1               50     /apps/oracle/oradata/ORCL/undotbs01.dbf
SYSAUX                 500    /apps/oracle/oradata/ORCL/sysaux01.dbf
SYSTEM                 680    /apps/oracle/oradata/ORCL/system01.dbf
EXAMPLE                100    /apps/oracle/oradata/ORCL/example01.dbf
COLLEGE                1      /apps/oracle/oradata/ORCL/college01.dbf
COLLEGE                20     /apps/oracle/oradata/ORCL/college02.dbf
```

Solution 2: Increase the size of the existing data file

```
SQL> alter database datafile '/apps/oracle/oradata/ORCL/college01.dbf' resize 20M;

Database altered.

SQL>@check_files
COLLEGE                              20 /apps/oracle/oradata/ORCL/college01.dbf
```

Solution 3: Set auto extend on

Step 1: If you created college02.dbf from the solution 2, in order to see the effect of solution 3, you need to drop it from the table space and physically removes it from the disk as shown below:

```
SQL> alter database datafile '/apps/oracle/oradata/ORCL/college02.dbf' offline drop;

Database altered.

SQL> !rm /apps/oracle/oradata/ORCL/college02.dbf
SQL> shutdown immediate;
SQL> startup;
```

Step 2: Resize college01.dbf back to 1 M and set auto extend on.

```
SQL> alter database datafile '/apps/oracle/oradata/ORCL/college01.dbf' resize 1M;
SQL> alter database datafile '/apps/oracle/oradata/ORCL/college01.dbf' autoextend on;

Database altered.

SQL> @usage

FILE# TS_NAME             MB TOTAL_BLOCKS USED_BLOCKS PCT_USED
----- ---------- ---------- ------------ ----------- ----------
    1 SYSTEM             680        87040       86024  98.83 %
    2 SYSAUX             510        65280       61200  93.75 %
    3 UNDOTBS1           115        14720        3496  23.75 %
    4 USERS                5          640         400  62.50 %
    5 EXAMPLE            100        12800        9912  77.44 %
    6 COLLEGE              1          128         120  93.75 %
    7 COLLEGE             20         2560              00.00 %

SQL> select tablespace_name, bytes/1024/1024 MB, file_name, autoextensible "Auto",
online_status from dba_data_files;

...
/apps/oracle/oradata/ORCL/college01.dbf YES ONLINE
...
```

Step 3: Populate the table with 500000 data and see if the table space can accommodate them.

```
SQL> begin
  2  for i in 1..500000 loop
  3  insert into scott.admission values(i);
  4  end loop;
  5  commit;
  6  end;
  7  /

PL/SQL procedure successfully completed. (takes some time)
```

```
SQL> /

PL/SQL procedure successfully completed.

SQL>@check_files

TABLESPACE_NAME                             MB
------------------------------ ----------
COLLEGE                              8.0625 ← previous 1 MB.

FILE_NAME                                     Aut ONLINE_
-----------------------------------------------------------
/apps/oracle/oradata/ORCL/college01.dbf YES ONLINE
```

Exercise V-3: To take a tablespace off line

- You may wish to take a tablespace offline for any of the following reasons: (http://www.riddle.ru/mirrors/oracledocs/server/sad73/ch803.html#taketaboff)
    - To make a portion of the database unavailable while allowing normal access to the remainder of the database.
    - To perform an offline tablespace backup (even though a tablespace can be backed up while online and in use).
    - To make an application and its group of tables temporarily unavailable while updating or maintaining the application.

- You can specify any of the following priorities when taking a tablespace offline:
    - Normal offline: A tablespace can be taken offline *normally* if no error conditions exist for any of the datafiles of the tablespace.
    - Temporary offline: A tablespace can be taken offline *temporarily*, even if there are error conditions for one or more files of the tablespace. Oracle7 takes offline the datafiles that are not already offline, checkpointing them as it does so.
    - Immediate offline: A tablespace can be taken offline *immediately*, without Oracle's taking a checkpoint on any of the datafiles. With immediate offline priority, media recovery for the tablespace is required before the tablespace can be brought online. You cannot take a tablespace offline immediately if the database is running in NOARCHIVELOG mode.

- To take a **table space** offline **(do not try it yet)**

```
SQL> alter tablespace college offline;
Tablespace altered.
```

- To take a **data file** offline **(do not try it yet.)**

```
SQL> alter database datafile '/apps/oracle/oradata/ORCL/college01.dbf'  offline drop;
```

Step1: Login as sysdba. If you dropped college02.dbf in the previous exercise, create it as shown below.

```
SQL> alter tablespace college
2 add datafile '/apps/oracle/oradata/ORCL/college02.dbf' size 20M;
```

Let's check the current status of data files.

```
SQL> col name for a50
SQL> select file#, name, status from v$datafile;

     FILE# NAME                                               STATUS
---------- -------------------------------------------------- -------
         1 /apps/oracle/oradata/ORCL/system01.dbf             SYSTEM
         2 /apps/oracle/oradata/ORCL/sysaux01.dbf             ONLINE
         3 /apps/oracle/oradata/ORCL/undotbs01.dbf            ONLINE
         4 /apps/oracle/oradata/ORCL/users01.dbf              ONLINE
         5 /apps/oracle/oradata/ORCL/example01.dbf            ONLINE
         6 /apps/oracle/oradata/ORCL/college01.dbf            ONLINE
         7 /apps/oracle/oradata/ORCL/college02.dbf            ONLINE

7 rows selected.
```

Step2: Take the datafile college02.dbf offline. If you take a data file offline in no archive mode, the status of the datafile becomes RECOVER.

```
SQL> alter database datafile '/apps/oracle/oradata/ORCL/college02.dbf' offline drop;

Database altered.

SQL> select file#, name, status from v$datafile;

     FILE# NAME                                               STATUS
---------- -------------------------------------------------- -------
         1 /apps/oracle/oradata/ORCL/system01.dbf             SYSTEM
         2 /apps/oracle/oradata/ORCL/sysaux01.dbf             ONLINE
         3 /apps/oracle/oradata/ORCL/undotbs01.dbf            ONLINE
         4 /apps/oracle/oradata/ORCL/users01.dbf              ONLINE
         5 /apps/oracle/oradata/ORCL/example01.dbf            ONLINE
         6 /apps/oracle/oradata/ORCL/college01.dbf            ONLINE
         7 /apps/oracle/oradata/ORCL/college02.dbf            RECOVER
```

Step 3: Take a table space called EXAMPLE offline and take it online back. The EXAMPLE tablespace contains one data file called example01.dbf.

Notice that after taking a table space offline and online it again, you need to make sure to generate checkpoint to synchronize SCN (Checkpoint Change Number).

```
SQL> alter tablespace example offline;

Tablespace altered.

SQL> select file#, name, status from v$datafile;

     FILE# NAME                                               STATUS
```

```
---------- ---------------------------------------------------- -------
         1 /apps/oracle/oradata/ORCL/system01.dbf               SYSTEM
         2 /apps/oracle/oradata/ORCL/sysaux01.dbf               ONLINE
         3 /apps/oracle/oradata/ORCL/undotbs01.dbf              ONLINE
         4 /apps/oracle/oradata/ORCL/users01.dbf                ONLINE
         5 /apps/oracle/oradata/ORCL/example01.dbf              OFFLINE
         6 /apps/oracle/oradata/ORCL/college01.dbf              ONLINE
         7 /apps/oracle/oradata/ORCL/college02.dbf              RECOVER
```

Step 4: Check the SCNs of table spaces. You will find that the SCNs of table spaces EXAMPLE and COLLEGE (that contains college02.dbf) not synchronized with the rest.

```
SQL> !vi show_scn.sql
select a.file#, a.ts#, b.name, a.status, a.checkpoint_change#
from v$datafile a, v$tablespace b
where a.ts#=b.ts#;


SQL> @show_scn

FILE#        TS# NAME     STATUS  CHECKPOINT_CHANGE#
----- ---------- ---------------------------------------
    1          0 SYSTEM   SYSTEM             1564342
    2          1 SYSAUX   ONLINE             1564342
    3          2 UNDOTBS1 ONLINE             1564342
    4          4 USERS    ONLINE             1564342
    5          6 EXAMPLE  OFFLINE            1565603
    6          7 COLLEGE  ONLINE             1564342
    7          7 COLLEGE  RECOVER            1565186

7 rows selected.
```

Step 5: Let's take the example table space back online. You will find that SCNs are not still synchronized

```
SQL>  alter tablespace example online;

Tablespace altered.

SQL> @show_scn

FILE#        TS# NAME      STATUS  CHECKPOINT_CHANGE#
-----------------------------------------------------
    1          0 SYSTEM    SYSTEM             1564342
    2          1 SYSAUX    ONLINE             1564342
    3          2 UNDOTBS1  ONLINE             1564342
    4          4 USERS     ONLINE             1564342
    5          6 EXAMPLE   ONLINE             1565729
    6          7 COLLEGE   ONLINE             1564342
    7          7 COLLEGE   RECOVER            1565186
```

Step 6: Generate a checkpoint.

```
SQL> alter system checkpoint;

System altered.
```

```
Now ALL SCNs are synchronized.

SQL> @show_scn

FILE#        TS# NAME              STATUS   CHECKPOINT_CHANGE#
----
    1          0 SYSTEM            SYSTEM              1565803
    2          1 SYSAUX            ONLINE              1565803
    3          2 UNDOTBS1          ONLINE              1565803
    4          4 USERS             ONLINE              1565803
    5          6 EXAMPLE           ONLINE              1565803
    6          7 COLLEGE           ONLINE              1565803
    7          7 COLLEGE          RECOVER             1565186

7 rows selected.
```

Step 7: Let's try to take COLLEGE table space offline. Since college02.dbf is already offline, college can't be offline normal. (If a table space contains an offline data file, you cannot take it offline normal.) You need to take it offline temporary.

```
SQL> alter tablespace college offline;
alter tablespace college offline
*
ERROR at line 1:
ORA-01191: file 7 is already offline - cannot do a normal offline
ORA-01110: data file 7: '/apps/oracle/oradata/ORCL/college02.dbf'


SQL> alter tablespace college offline temporary;

Tablespace altered.

SQL> recover tablespace college;
Media recovery complete.

SQL> alter tablespace college online;

Tablespace altered.

SQL> @show_scn

FILE#        TS# NAME           STATUS   CHECKPOINT_CHANGE#
----- ---------- ---------------------------- ------------------
    1          0 SYSTEM         SYSTEM             1565803
    2          1 SYSAUX         ONLINE             1565803
    3          2 UNDOTBS1       ONLINE             1565803
    4          4 USERS          ONLINE             1565803
    5          6 EXAMPLE        ONLINE             1565803
    6          7 COLLEGE        ONLINE             1565888
    7          7 COLLEGE        ONLINE             1565888

7 rows selected.
```

```
SQL> alter system checkpoint;

System altered.


SQL> @show_scn

FILE#        TS# NAME        STATUS  CHECKPOINT_CHANGE#
----- ---------- ----------- ------- ------------------
    1          0 SYSTEM      SYSTEM             1565926
    2          1 SYSAUX      ONLINE             1565926
    3          2 UNDOTBS1    ONLINE             1565926
    4          4 USERS       ONLINE             1565926
    5          6 EXAMPLE     ONLINE             1565926
    6          7 COLLEGE     ONLINE             1565926
    7          7 COLLEGE     ONLINE             1565926

7 rows selected.
```

Exercise V-4: In this exercise, we will learn how to move data files in database. A rule of thumb is that you should not move a data file when it is being used. Take it off line or shutdown the instance before moving a file.

Case 1: If you can take the table space containing target data files offline, do the following procedure.

1) Offline the table space that contains the target files.
2) Copy the data files to the destination directory.
3) Modify the location of the data file in the control file.
4) Online the table space back.

The above procedure will make only corresponding table space unavailable.Let's practice this procedure. We are going to move college01.dbf and college02.dbf files to /apps/oracle/disk1/college01.dbf and /apps/oracle/disk2/college02.dbf, respectively.

Step 1: Create directories /apps/oracle/disk1 and /apps/oracle/disk2

```
SQL> !mkdir /apps/oracle/disk1
SQL> !mkdir /apps/oracle/disk2
```

Step 2: Take COLLEGE tablespace offline.

```
SQL> alter tablespace college offline;
Tablespace altered.
```

Step 3: Copy files to the destination

```
SQL> !cp /apps/oracle/oradata/ORCL/college01.dbf /apps/oracle/disk1/
SQL> !cp /apps/oracle/oradata/ORCL/college02.dbf /apps/oracle/disk2/
```

Step 4: Check the current locations of the files in the control file and update their locations.

```
SQL> select name from v$datafile; ← current content of control file

NAME
--------------------------------------------------
/apps/oracle/oradata/ORCL/system01.dbf
/apps/oracle/oradata/ORCL/sysaux01.dbf
/apps/oracle/oradata/ORCL/undotbs01.dbf
/apps/oracle/oradata/ORCL/users01.dbf
/apps/oracle/oradata/ORCL/example01.dbf
/apps/oracle/oradata/ORCL/college01.dbf
/apps/oracle/oradata/ORCL/college02.dbf


SQL> alter tablespace COLLEGE rename
  2  datafile '/apps/oracle/oradata/ORCL/college01.dbf'
  3  to '/apps/oracle/disk1/college01.dbf';


Tablespace altered.
SQL> alter tablespace COLLEGE rename
  2  datafile '/apps/oracle/oradata/ORCL/college02.dbf'
  3   to '/apps/oracle/disk2/college02.dbf';

Tablespace altered.

SQL> select name from v$datafile;

NAME
--------------------------------------------------
/apps/oracle/oradata/ORCL/system01.dbf
/apps/oracle/oradata/ORCL/sysaux01.dbf
/apps/oracle/oradata/ORCL/undotbs01.dbf
/apps/oracle/oradata/ORCL/users01.dbf
/apps/oracle/oradata/ORCL/example01.dbf
/apps/oracle/disk1/college01.dbf
/apps/oracle/disk2/college02.dbf

7 rows selected.
```

Step 5: Take back the COLLEGE tablespace online

```
SQL> alter tablespace college online;
Tablespace altered.
```

Case 2: If you cannot take a tablespace online, shutdown instance. Example of such table spaces include SYSTEM table space, UNDO table space in use, default temporary table space. When you move log files, this case also applies. Here is the procedure:

1) Shutdown the database
2) Start the database in mount mode.
3) Copy data files
4) Modify the location of the data file in the control file
5) Open database

Let's practice this procedure.

Step1: Shutdown instance

```
SQL> shutdown immediate;
Database closed.
Database dismounted.
ORACLE instance shut down.
```

Step 2: Start the database in mount mode.

```
SQL> startup mount;
ORACLE instance started.

Total System Global Area  784998400 bytes
Fixed Size                  2217464 bytes
Variable Size             574622216 bytes
Database Buffers          201326592 bytes
Redo Buffers                6832128 bytes
Database mounted.
```

Step 3: Copy data files to the destinations.

```
SQL> !cp  /apps/oracle/oradata/ORCL/system01.dbf /apps/oracle/disk3/

SQL> select name from v$datafile;

NAME
--------------------------------------------------------------------------------
/apps/oracle/oradata/ORCL/system01.dbf
/apps/oracle/oradata/ORCL/sysaux01.dbf
/apps/oracle/oradata/ORCL/undotbs01.dbf
/apps/oracle/oradata/ORCL/users01.dbf
/apps/oracle/oradata/ORCL/example01.dbf
/apps/oracle/disk1/college01.dbf
/apps/oracle/disk2/college02.dbf

7 rows selected.
```

Step 4: Modify the location of the data file in the control file

```
SQL> alter database rename
  2  file '/apps/oracle/oradata/ORCL/system01.dbf'
  3  to '/apps/oracle/disk3/system01.dbf';

Database altered.

SQL> select name from v$datafile;

NAME
```

```
------------------------------------------------
/apps/oracle/disk3/system01.dbf
/apps/oracle/oradata/ORCL/sysaux01.dbf
/apps/oracle/oradata/ORCL/undotbs01.dbf
/apps/oracle/oradata/ORCL/users01.dbf
/apps/oracle/oradata/ORCL/example01.dbf
/apps/oracle/disk1/college01.dbf
/apps/oracle/disk2/college02.dbf

7 rows selected.
```

Step 5: Open database

```
SQL> alter database open;
Database altered.
```

User Management

User management is briefly addressed here. For more details, you may refer to http://docs.oracle.com/cd/E27559_01/user.1112/e27151/usr_mangmnt.htm .

- Terminology: User vs. Schema
  - User: scott
  - Schema: All oracle objects a user created. E.g. table, index, view, constraint, trigger, dblinnk, synonym, sequence which scott created.
- Sample DDL to manage users.

```
CREATE USER jward
    IDENTIFIED BY aZ7bC2
    DEFAULT TABLESPACE data_ts
    QUOTA 100M ON test_ts
    QUOTA 500K ON data_ts
    TEMPORARY TABLESPACE temp_ts
    PROFILE clerk;
GRANT connect TO jward;

ALTER USER avyrros
    IDENTIFIED EXTERNALLY
    DEFAULT TABLESPACE data_ts
    TEMPORARY TABLESPACE temp_ts
    QUOTA 100M ON data_ts
    QUOTA 0 ON test_ts
    PROFILE clerk;

DROP USER jones CASCADE;
```