Introduction to Oracle Database

An Oracle database is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a server reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.

Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need for peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.

The database has logical structures and physical structures. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures.

Overview of Oracle Grid Architecture

The Oracle grid architecture pools large numbers of servers, storage, and networks into a flexible, on-demand computing resource for enterprise computing needs. The grid computing infrastructure continually analyzes demand for resources and adjusts supply accordingly.

For example, you could run different applications on a grid of several linked database servers. When reports are due at the end of the month, the database administrator could automatically provision more servers to that application to handle the increased demand.

Grid computing uses sophisticated workload management that makes it possible for applications to share resources across many servers. Data processing capacity can be added or removed on demand, and resources within a location can be dynamically provisioned. Web services can quickly integrate applications to create new business processes.

Difference between a cluster and a grid

Clustering is one technology used to create a grid infrastructure. Simple clusters have static resources for specific applications by specific owners. Grids, which can consist of multiple clusters, are dynamic resource pools shareable among many different applications and users. A grid does not assume that all servers in the grid are running the same set of applications. Applications can be scheduled and migrated across servers in the grid. Grids share resources from and among independent system owners.

At the highest level, the idea of grid computing is computing as a utility. In other words, you should not care where your data resides, or what computer processes your request. You should be able to request information or computation and have it delivered - as much as you want, and whenever you want. This is analogous to the way electric utilities work, in that you don't know where the generator is, or how the electric grid is wired, you just ask for electricity, and you get it. The goal is to make computing a utility, a commodity, and ubiquitous. Hence the name, The Grid. This view of utility computing is, of course, a "client side" view.

From the "server side", or behind the scenes, the grid is about resource allocation, information sharing, and high availability. Resource allocation ensures that all those that need or request resources are getting what they need, that resources are not standing idle while requests are going unserviced. Information sharing makes sure that the information users and applications need is available where and when it is needed. High availability features guarantee all the data and computation is always there, just like a utility company always provides electric power.

Responsibilities of Database Administrators

Each database requires at least one database administrator (DBA). An Oracle Database system can be large and can have many users. Therefore, database administration is sometimes not a one-person job, but a job for a group of DBAs who share responsibility.

A database administrator's responsibilities can include the following tasks:

- Installing and upgrading the Oracle Database server and application tools
- Allocating system storage and planning future storage requirements for the database system
- Creating primary database storage structures (tablespaces) after application developers have designed an application
- Creating primary objects (tables, views, indexes) once application developers have designed an application
- Modifying the database structure, as necessary, from information given by application developers
- Enrolling users and maintaining system security
- Ensuring compliance with Oracle license agreements

- Controlling and monitoring user access to the database
- Monitoring and optimizing the performance of the database
- Planning for backup and recovery of database information
- Maintaining archived data on tape
- Backing up and restoring the database
- Contacting Oracle for technical support

Creating the Oracle Database through SQL Commands

This section presents the steps involved when you create a database manually. These steps should be followed in the order presented. Before you create the database make sure you have done the planning about the size of the database, number of tablespaces and redo log files you want in the database.

Regarding the size of the database you have to first find out how many tables are going to be created in the database and how much space they will be occupying for the next 1 year or 2. The best thing is to start with some specific size and later on adjust the size depending upon the requirement

Plan the layout of the underlying operating system files your database will comprise. Proper distribution of files can improve database performance dramatically by distributing the I/O during file access. You can distribute I/O in several ways when you install Oracle software and create your database. For example, you can place redo log files on separate disks or use striping. You can situate datafiles to reduce contention. And you can control data density (number of rows to a data block).

Select the standard database block size. This is specified at database creation by the DB_BLOCK_SIZE initialization parameter and cannot be changed after the database is created. For databases, block size of 4K or 8K is widely used

Before you start creating the Database it is best to write down the specification and then proceed

The examples shown in these steps create an example database my_ica_db

Let us create a database my_ica_db with the following specification

Database name and System Identifier

SID=myicadb
DB_NAME=myicadb

TABLESPACES
(we will have 6 tablespaces in this database. With 1 datafile in each tablespace)

Tablespace Name	**Datafile Location**	**Size**
SYSTEM	/u01/oracle/oradata/myica/sys.dbf	500M
USERS	/u01/oracle/oradata/myica/usr.dbf	100M
UNDOTBS	/u01/oracle/oradata/myica/undo.dbf	100M
TEMP	/u01/oracle/oradata/myica/temp.dbf	100M
INDEX_DATA	/u01/oracle/oradata/myica/indx.dbf	100M
SYSAUX	/u01/oracle/oradata/myica/sysaux.dbf	100M

LOGFILES
(we will have 2 log groups in the database)

Logfile Group	**Member Location**	**Size**
GROUP 1	/u01/oracle/oradata/myica/log1.ora	10M
GROUP 2	/u01/oracle/oradata/myica/log2.ora	10M

CONTROL FILE
(We will have 1 Control File in the following location)

/u01/oracle/oradata/myica/control.ora

PARAMETER FILE
(we will use normal parameter file for now, later on we can switch to SPFile)

/u01/oracle/dbs/initmyicadb.ora

(remember the parameter file name should of the format init<sid>.ora and it should be
in ORACLE_HOME/dbs directory in Unix o/s and ORACLE_HOME/database directory in windows o/s)

Now let us start creating the database.

Step 1: Login to oracle account and make directories for your database.

```
$ mkdir /u01/oracle/oradata/myica
$ mkdir /u01/oracle/oradata/myica/bdump
```

$ mkdir /u01/oracle/oradata/myica/udump
$ mkdir /u01/oracle/oradata/myica/recovery

Step 2: Create the parameter file by copying the default template (init.ora) and set the required parameters

$ cd /u01/oracle/dbs
$ cp init.ora initmyicadb.ora

Now open the parameter file and set the following parameters

$ vi initmyicadb.ora

DB_NAME=myicadb
DB_BLOCK_SIZE=8192
CONTROL_FILES=/u01/oracle/oradata/myica/control.ora
UNDO_TABLESPACE=undotbs
UNDO_MANAGEMENT=AUTO
SGA_TARGET=500M
PGA_AGGREGATE_TARGET=100M
LOG_BUFFER=5242880
DB_RECOVERY_FILE_DEST=/u01/oracle/oradata/myica/recovery
DB_RECOVERY_FILE_DEST_SIZE=2G

The following parameters are required only in 10g or earlier versions

BACKGROUND_DUMP_DEST=/u01/oracle/oradata/myica/bdump
USER_DUMP_DEST=/u01/oracle/oradata/myica/udump

After entering the above parameters save the file by pressing "Esc :wq"

Step 3: Now set ORACLE_SID environment variable and start the instance.

$ export ORACLE_SID=myicadb

$ sqlplus
Enter User: / as sysdba
SQL>startup nomount

Step 4: Give the create database command

Here I am not specifying optional setting such as language, characterset etc. For these settings oracle will use
the default values. I am giving the bare minimum command to create the database to keep it simple.

The command to create the database is

```
SQL> create database myicadb
    datafile '/u01/oracle/oradata/myica/sys.dbf' size 500M
    sysaux datafile '/u01/oracle/oradata/myica/sysaux.dbf' size 100m
    undo tablespace undotbs
    datafile '/u01/oracle/oradata/myica/undo.dbf' size 100m
    default temporary tablespace temp
    tempfile '/u01/oracle/oradata/myica/tmp.dbf' size 100m
    logfile
        group 1 '/u01/oracle/oradata/myica/log1.ora' size 50m,
        group 2 '/u01/oracle/oradata/myica/log2.ora' size 50m;
```

After the command finishes you will get the following message

Database created.

If you are getting any errors then see accompanying messages. If no accompanying messages are shown then
you have to see the alert_myicadb.log file located in BACKGROUND_DUMP_DEST directory, which will show the
exact reason why the command has failed. After you have rectified the error please delete all created files in
u01/oracle/oradata/myica directory and again give the above command.

Step 5: After the above command finishes, the database will get mounted and opened. Now create additional
tablespaces

To create USERS tablespace

```
SQL> create tablespace users
datafile '/u01/oracle/oradata/myica/usr.dbf' size 100M;
```

To create INDEX_DATA tablespace

```
SQL>create tablespace index_data
 datafile '/u01/oracle/oradata/myica/indx.dbf' size 100M
```

Step 6: To populate the database with data dictionaries and to install
procedural options execute the following scripts

First execute the CATALOG.SQL script to install data dictionaries

```
SQL>@/u01/oracle/rdbms/admin/catalog.sql
```

The above script will take several minutes. After the above script is finished run the CATPROC.SQL script to install procedural option.

SQL>@/u01/oracle/rdbms/admin/catproc.sql

This script will also take several minutes to complete.

Step 7: Now change the passwords for SYS and SYSTEM account, since the default passwords change_on_install and manager are known by everybody.

SQL>alter user sys identified by myica;
SQL>alter user system identified by myica;

Step 8: Create Additional user accounts. You can create as many user account as you like. Let us create the popular account SCOTT.

SQL>create user scott default tablespace users identified by tiger quota 10M on users;
SQL>grant connect to scott;

Step 9: Add this database SID in listener.ora file and restart the listener process.

$ cd /u01/oracle/network/admin

$ vi listener.ora

(This file will already contain sample entries. Copy and paste one sample entry and edit the SID setting)

```
LISTENER =
  (DESCRIPTION_LIST =
    (DESCRIPTION =
     (ADDRESS =(PROTOCOL = TCP)(HOST=200.200.100.1)(PORT = 1521))
    )
  )
 SID_LIST_LISTENER =
  (SID_LIST =
    (SID_DESC =
      (SID_NAME = PLSExtProc)
      (ORACLE_HOME =/u01/oracle)
      (PROGRAM = extproc)
    )
            (SID_DESC =
     (SID_NAME=orcl)
     (ORACLE_HOME=/u01/oracle)
    )
   )
```

#Add these lines in SID_LIST_LISTENER at the bottom of file

```
        (SID_DESC =
     (SID_NAME=myicadb)
    (ORACLE_HOME=/u01/oracle)
   )
```

Save the file by pressing Esc :wq

Now restart the listener process.

```
$ lsnrctl stop
$ lsnrctl start
```

Step 10: It is recommended to take a full database backup after you just created the database.

How to take backup is deal in the Backup and Recovery Section.

Congratulations you have just created an oracle database.

Managing Tablespaces and Datafiles

Using multiple tablespaces provides several Advantages

- Separate user data from data dictionary data to reduce contention among dictionary objects and schema objects for the same datafiles.
- Separate data of one application from the data of another to prevent multiple applications from being affected if a tablespace must be taken offline.
- Store different the datafiles of different tablespaces on different disk drives to reduce I/O contention.
- Take individual tablespaces offline while others remain online, providing better overall availability.

Creating New Tablespaces

You can create Locally Managed or Dictionary Managed Tablespaces. In prior versions of Oracle only Dictionary managed Tablespaces were available but from Oracle ver. 8i you can also create Locally Managed tablespaces. The advantages of locally managed tablespaces are

Locally managed tablespaces track all extent information in the tablespace itself by using bitmaps, resulting in the following benefits:

- Concurrency and speed of space operations is improved, because space allocations and deallocations modify locally managed resources (bitmaps stored in header files) rather than requiring centrally managed resources such as enqueues
- Performance is improved, because recursive operations that are sometimes required during dictionary-managed space allocation are eliminated

To create a locally managed tablespace give the following command

SQL> CREATE TABLESPACE ica_lmts DATAFILE '/u02/oracle/ica/ica01.dbf' SIZE 50M EXTENT MANAGEMENT LOCAL AUTOALLOCATE;

AUTOALLOCATE causes the tablespace to be system managed with a minimum extent size of 64K.

The alternative to AUTOALLOCATE is UNIFORM. which specifies that the tablespace is managed with extents of uniform size. You can specify that size in the SIZE clause of UNIFORM. If you omit SIZE, then the default size is 1M. The following example creates a Locally managed tablespace with uniform extent size of 256K

SQL> CREATE TABLESPACE ica_lmt DATAFILE '/u02/oracle/ica/ica01.dbf' SIZE 50M EXTENT MANAGEMENT LOCAL UNIFORM SIZE 256K;

To Create Dictionary Managed Tablespace

SQL> CREATE TABLESPACE ica_lmt DATAFILE '/u02/oracle/ica/ica01.dbf' SIZE 50M EXTENT MANAGEMENT DICTIONARY;

Bigfile Tablespaces (Introduced in Oracle Ver. 10g)

A bigfile tablespace is a tablespace with a single, but very large (up to 4G blocks) datafile. Traditional smallfile tablespaces, in contrast, can contain multiple datafiles, but the files cannot be as large. Bigfile tablespaces can reduce the number of datafiles needed for a database.

To create a bigfile tablespace give the following command

SQL> CREATE BIGFILE TABLESPACE ica_bigtbs DATAFILE '/u02/oracle/ica/bigtbs01.dbf' SIZE 50G;

Extending the Size of Tablespace and Datafiles

To Extend the Size of a tablespace

Option 1

You can extend the size of a tablespace by increasing the size of an existing datafile by typing the following command

SQL> alter database ica datafile '/u01/oracle/data/icatbs01.dbf' resize 100M;

This will increase the size from 50M to 100M

Option 2

You can also extend the size of a tablespace by adding a new datafile to a tablespace. This is useful if the size of existing datafile is reached o/s file size limit or the drive where the file is existing does not have free space. To add a new datafile to an existing tablespace give the following command.

SQL> alter tablespace add datafile '/u02/oracle/ica/icatbs02.dbf' size 50M;

Option 3

You can also use auto extend feature of datafile. In this, Oracle will automatically increase the size of a datafile whenever space is required. You can specify by how much size the file should increase and Maximum size to which it should extend.

To make a existing datafile auto extendable give the following command

SQL> alter database datafile '/u01/oracle/ica/icatbs01.dbf' auto extend ON next 5M maxsize 500M;

You can also make a datafile auto extendable while creating a new tablespace itself by giving the following command.

SQL> create tablespace ica datafile '/u01/oracle/ica/icatbs01.dbf' size 50M auto extend ON next 5M maxsize 500M;

To decrease the size of a tablespace

You can decrease the size of tablespace by decreasing the datafile associated with it. You decrease a datafile only up to size of empty space in it. To decrease the size of a datafile give the following command

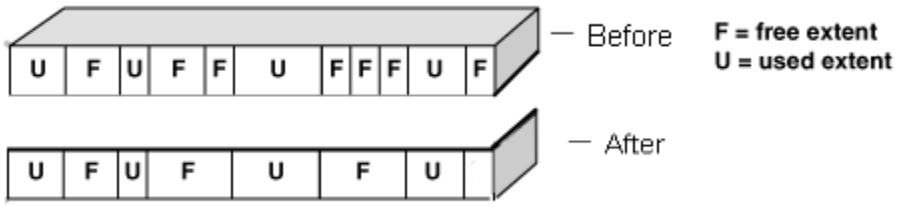
SQL> alter database datafile '/u01/oracle/ica/icatbs01.dbf' resize 30M;

Coalescing Tablespaces

A free extent in a dictionary-managed tablespace is made up of a collection of contiguous free blocks. When allocating new extents to a tablespace segment, the database uses the free extent closest in size to the required extent. In some cases, when segments are dropped, their extents are deallocated and marked as free, but adjacent free extents are not immediately recombined into larger free extents. The result is fragmentation that makes allocation of larger extents more difficult.

You should often use the ALTER TABLESPACE ... COALESCE statement to manually coalesce any adjacent free extents. To Coalesce a tablespace give the following command

SQL> alter tablespace ica coalesce;



Managing Availibility of Tablespaces in Oracle

Taking tablespaces Offline or Online

You can take an online tablespace offline so that it is temporarily unavailable for general use. The rest of the database remains open and available for users to access data. Conversely, you can bring an offline tablespace online to make the schema objects within the tablespace available to database users. The database must be open to alter the availability of a tablespace.

We usually take tablespaces offline for maintenance purposes.

To alter the availability of a tablespace, use the ALTER TABLESPACE statement. You must have the ALTER TABLESPACE or MANAGE TABLESPACE system privilege.

To Take a Tablespace Offline give the following command

SQL>alter tablespace ica offline;

To again bring it back online give the following command.

SQL>alter tablespace ica online;

To take individual datafile offline type the following command

SQL>alter database datafile '/u01/oracle/ica/ica_tbs01.dbf' offline;

Again to bring it back online give the following command

SQL> alter database datafile '/u01/oracle/ica/ica_tbs01.dbf' online;

Note: You can't take individual datafiles offline it the database is running in NOARCHIVELOG mode. If the datafile has become corrupt or missing when the database is running in NOARCHIVELOG mode then you can only drop it by giving the following command

SQL>alter database datafile '/u01/oracle/ica/ica_tbs01.dbf' offline for drop;

Making a Tablespace Read only.

Making a tablespace read-only prevents write operations on the datafiles in the tablespace. The primary purpose of read-only tablespaces is to eliminate the need to perform backup and recovery of large, static portions of a database. Read-only tablespaces also provide a way to protecting historical data so that users cannot modify it. Making a tablespace read-only prevents updates on all tables in the tablespace, regardless of a user's update privilege level.

To make a tablespace read only

SQL>alter tablespace ica read only

Again to make it read write

SQL>alter tablespace ica read write;

Renaming Tablespaces

Using the RENAME TO clause of the ALTER TABLESPACE, you can rename a permanent or temporary tablespace. For example, the following statement renames the users tablespace:

ALTER TABLESPACE users RENAME TO usersts;

The following affect the operation of this statement:

- The COMPATIBLE parameter must be set to 10.0 or higher.
- If the tablespace being renamed is the SYSTEM tablespace or the SYSAUX tablespace, then it will not be renamed and an error is raised.
- If any datafile in the tablespace is offline, or if the tablespace is offline, then the tablespace is not renamed and an error is raised.

Dropping Tablespaces

You can drop a tablespace and its contents (the segments contained in the tablespace) from the database if the tablespace and its contents are no longer required. You must have the DROP TABLESPACE system privilege to drop a tablespace.

Caution: Once a tablespace has been dropped, the data in the tablespace is not recoverable. Therefore, make sure that all data contained in a tablespace to be dropped will not be required in the future. Also, immediately before and after dropping a tablespace from a database, back up the database completely

To drop a tablespace give the following command.

SQL> drop tablespace ica;

This will drop the tablespace only if it is empty. If it is not empty and if you want to drop it anyhow then add the following keyword

SQL>drop tablespace ica including contents;

This will drop the tablespace even if it is not empty. But the datafiles will not be deleted you have to use operating system command to delete the files.

But If you include datafiles keyword then, the associated datafiles will also be deleted from the disk.

SQL>drop tablespace ica including contents and datafiles;

Viewing Information about Tablespaces and Datafiles

Oracle has provided many Data dictionaries to view information about tablespaces and datafiles. Some of them are:

To view information about Tablespaces in a database give the following query

SQL> select * from dba_tablespaces
SQL> select * from v$tablespace;

To view information about Datafiles

SQL> select * from dba_data_files;
SQL> select * from v$datafile;

To view information about Tempfiles

SQL> select * from dba_temp_files;
SQL> select * from v$tempfile;

To view information about free space in datafiles

SQL> select * from dba_free_space;

To view information about free space in tempfiles

SQL> select * from V$TEMP_SPACE_HEADER;

Renaming or Relocating Datafiles belonging to a Single Tablespace

You can rename datafiles to either change their names or relocate them.

To rename or relocate datafiles belonging to a Single Tablespace do the following.

1. Take the tablespace offline

2. Rename or Relocate the datafiles using operating system command

3. Give the ALTER TABLESPACE with RENAME DATAFILE option to change the filenames within the Database.

4. Bring the tablespace Online

For Example suppose you have a tablespace users with the following datafiles

/u01/oracle/ica/usr01.dbf'
/u01/oracle/ica/usr02.dbf'

Now you want to relocate /u01/oracle/ica/usr01.dbf' to '/u02/oracle/ica/usr01.dbf' and want to rename '/u01/oracle/ica/usr02.dbf' to '/u01/oracle/ica/users02.dbf' then follow the given the steps

1. Bring the tablespace offline

SQL> alter tablespace users offline;

2. Copy the file to new location using o/s command.

$ cp /u01/oracle/ica/usr01.dbf /u02/oracle/ica/usr01.dbf'

Rename the file '/u01/oracle/ica/usr02.dbf' to '/u01/oracle/ica/users02.dbf' using o/s command.

$ mv /u01/oracle/ica/usr02.dbf /u01/oracle/ica/users02.dbf

3. Now start SQLPLUS and type the following command to rename and relocate these files

```
SQL> alter tablespace users rename file '/u01/oracle/ica/usr01.dbf',
     '/u01/oracle/ica/usr02.dbf' to      '/u02/oracle/ica/usr01.dbf',
                         '/u01/oracle/ica/users02.dbf';
```

4. Now bring the tablespace Online

SQL> alter tablespace users online;

Procedure for Renaming and Relocating Datafiles in Multiple Tablespaces

You can rename and relocate datafiles in one or more tablespaces using the ALTER DATABASE RENAME FILE statement. This method is the only choice if you want to rename or relocate datafiles of several tablespaces in one operation. You must have the ALTER DATABASE system privilege

To rename datafiles in multiple tablespaces, follow these steps.

1. Ensure that the database is mounted but closed.

2. Copy the datafiles to be renamed to their new locations and new names, using the operating system..

3. Use ALTER DATABASE to rename the file pointers in the database control file.

For example, the following statement renames the datafiles/u02/oracle/rbdb1/sort01.dbf and /u02/oracle/rbdb1/user3.dbf to /u02/oracle/rbdb1/temp01.dbf and /u02/oracle/rbdb1/users03.dbf, respectively:

```
ALTER DATABASE
   RENAME FILE '/u02/oracle/rbdb1/sort01.dbf',
               '/u02/oracle/rbdb1/user3.dbf'
            TO '/u02/oracle/rbdb1/temp01.dbf',
               '/u02/oracle/rbdb1/users03.dbf;
```

Always provide complete filenames (including their paths) to properly identify the old and new datafiles. In particular, specify the old datafile names exactly as they appear in the DBA_DATA_FILES view.

4. Back up the database. After making any structural changes to a database, always perform an immediate and complete backup.

5. Start the Database

Managing Tablespaces and Datafiles

Using multiple tablespaces provides several Advantages

- Separate user data from data dictionary data to reduce contention among dictionary objects and schema objects for the same datafiles.
- Separate data of one application from the data of another to prevent multiple applications from being affected if a tablespace must be taken offline.
- Store different the datafiles of different tablespaces on different disk drives to reduce I/O contention.
- Take individual tablespaces offline while others remain online, providing better overall availability.

Creating New Tablespaces

You can create Locally Managed or Dictionary Managed Tablespaces. In prior versions of Oracle only Dictionary managed Tablespaces were available but from Oracle ver. 8i you can also create Locally Managed tablespaces. The advantages of locally managed tablespaces are

Locally managed tablespaces track all extent information in the tablespace itself by using bitmaps, resulting in the following benefits:

- Concurrency and speed of space operations is improved, because space allocations and deallocations modify locally managed resources (bitmaps stored in header files) rather than requiring centrally managed resources such as enqueues

- Performance is improved, because recursive operations that are sometimes required during dictionary-managed space allocation are eliminated

To create a locally managed tablespace give the following command

SQL> CREATE TABLESPACE ica_lmts DATAFILE '/u02/oracle/ica/ica01.dbf' SIZE 50M EXTENT MANAGEMENT LOCAL AUTOALLOCATE;

AUTOALLOCATE causes the tablespace to be system managed with a minimum extent size of 64K.

The alternative to AUTOALLOCATE is UNIFORM. which specifies that the tablespace is managed with extents of uniform size. You can specify that size in the SIZE clause of UNIFORM. If you omit SIZE, then the default size is 1M. The following example creates a Locally managed tablespace with uniform extent size of 256K

SQL> CREATE TABLESPACE ica_lmt DATAFILE '/u02/oracle/ica/ica01.dbf' SIZE 50M EXTENT MANAGEMENT LOCAL UNIFORM SIZE 256K;

To Create Dictionary Managed Tablespace

SQL> CREATE TABLESPACE ica_lmt DATAFILE '/u02/oracle/ica/ica01.dbf' SIZE 50M EXTENT MANAGEMENT DICTIONARY;

Bigfile Tablespaces (Introduced in Oracle Ver. 10g)

A bigfile tablespace is a tablespace with a single, but very large (up to 4G blocks) datafile. Traditional smallfile tablespaces, in contrast, can contain multiple datafiles, but the files cannot be as large. Bigfile tablespaces can reduce the number of datafiles needed for a database.

To create a bigfile tablespace give the following command

SQL> CREATE BIGFILE TABLESPACE ica_bigtbs DATAFILE '/u02/oracle/ica/bigtbs01.dbf' SIZE 50G;

Procedure for Renaming and Relocating Datafiles in Multiple Tablespaces

You can rename and relocate datafiles in one or more tablespaces using the ALTER DATABASE RENAME FILE statement. This method is the only choice if you want to rename or relocate datafiles of several tablespaces in one operation. You must have the ALTER DATABASE system privilege

To rename datafiles in multiple tablespaces, follow these steps.

1. Ensure that the database is mounted but closed.

2. Copy the datafiles to be renamed to their new locations and new names, using the operating system..

3. Use ALTER DATABASE to rename the file pointers in the database control file.

For example, the following statement renames the datafiles/u02/oracle/rbdb1/sort01.dbf and /u02/oracle/rbdb1/user3.dbf to /u02/oracle/rbdb1/temp01.dbf and /u02/oracle/rbdb1/users03.dbf, respectively:

```
ALTER DATABASE
   RENAME FILE '/u02/oracle/rbdb1/sort01.dbf',
               '/u02/oracle/rbdb1/user3.dbf'
            TO '/u02/oracle/rbdb1/temp01.dbf',
               '/u02/oracle/rbdb1/users03.dbf;
```

Always provide complete filenames (including their paths) to properly identify the old and new datafiles. In particular, specify the old datafile names exactly as they appear in the DBA_DATA_FILES view.

4. Back up the database. After making any structural changes to a database, always perform an immediate and complete backup.

5. Start the Database