# **Tablespace Management in oracle**

* Tablespace is the primary logic structure of the database. It consists of one or more physical datafiles. Datafiles stores database data in it.(DISKS) . So when we create a datafile of 30G in database, it will eat 30G of space from defined storage.
* System and undo are critical tablespaces we cannot offline these tablespaces.
* maximum 1024 number of datafiles’s can be added under a tablespace.

### **Types of Tablespace:**

1. **system**

* metadata tables of the database is stored in system tablespace.

1. **sysaux**

* it stores the snap id of the database.this was interduces from 10g.
* Performance related like awr report will be stored.

1. **users**

* when ever user is creating objects it stores that objects.

1. **undo**

* when the operations like dml are happening and the transaction has be rolled back the previous information will be stored under undo tablespace.

1. **Temp**

* when the user retrieving data from multiple tables that data need some storage point.Temp will hold the data for temporarily.

**Physical definition of tablespace**

* Tablespace contains one or more datafiles it is a physical file.

**Logical defination of tablespace**

* tablespace is a collection of segments.
* Segments is a collection of extents.
* Extents is collection of contiguous oracle blocks

**There are two type's of tablespace's**

**SMALL FILE & BIG FILE TABLESPACE**

1. **Small file**

* Small file tablespace size in GB
* It contain one or more datafiles.By default all the tablespace are created with small file .
* We can add upto 1024 datafile’s when the existing datafile filled.

(32k block size 128gb)

(16k block size 64gb)

(8k block size 32gb)

(4K block size 16gb)

(2k block size 8gb)

1. **Big file**

* Big file tablespace size in TB
* It contain only one datafile.
* We cannot add datafile when this tablespace filled.
* we can also created big file tablespace
* Create bigfile tablespace palikila datafile ‘/prod/hyd/oradata/palikila01.dbf’ size 30 m;

(32k block size 128T)

(16k block size 64T)

(8k block size 32T)

(4K block size 16T)

(2k block size 8T)

**Main view of tablespace**

* desc dba\_tablespaces
* select contents from dba\_tablespaces;
* Select distinct contents from dba\_tablespaces;

**To check tablespace permanent or Temp:**

* select tablespace\_name,contents from dba\_tablespaces;

**To check users default\_tablespace**

* select username,default\_tablespace from dba\_users where username=‘Palikila’;

**To check tablespace size:**

column "tablespace" format a30

column "used MB" format 99,999,999

column "free MB" format 99,999,999

column "total MB" format 99,999,999

select fs.tablespace\_name "tablespace",(df.totalspace - fs.freespace)

"used MB", fs.freespace "free MB",df.totalspace "total MB",

round(100\*(fs.freespace / df.totalspace)) "pct.free" from (select tablespace\_name,round(sum(bytes)/1048576)

totalspace from dba\_data\_files group by tablespace\_name) df,(select tablespace\_name,round(sum(bytes)/1048576)

FreeSpace from dba\_free\_space group by tablespace\_name) fs where df.tablespace\_name=fs.tablespace\_name;

**To change user default tablespace**

* alter user mouli default tablespace mouli;

**To check how many users linked with tablespace(Palikila):**

* select username from dba\_users where default\_tablespace=‘palikila’;

**To allocate tablespace to user**

* alter user u1 default tablespace Palikila;

**To check tablespace’s**

* select name from v$tablespace;

**To create tablespace (mouli)**

* create tablespace mouli datafile ‘/path/ tablespace\_name01.dbf’. Size 100m;

**To know datafile location**

* select name from v$ datafile;

**To check datafiles under tablespace**

* select file\_name from dba\_data\_files where tablespace\_name=‘palikila’;

**To drop tablespace**

* drop tablespace palikila including contents and datafiles;
* droping Tablespace will drop datafiles and data.

**To Rename tablespace**

* alter tablespace mouli rename to mouli\_palikila;

**Autoextend ON & OFF**

* By default autoextend is disabled for datafile’s.
* When the datafile is created with 100m and autoextend ON it automatically extends the size up to 32 gb.

**To check datafile is autoextend enable or not:**

* Desc dba\_data\_files
* select FILE\_NAME,BYTES/1024/1024,MAXBYTES/1024/1024,AUTOEXTEND ON from dba\_data\_files;

**If tablespace is filled**

* Check space in os level

**We can do two things**

1. resize the existing datafile

* resize datafile under tablespace Palikila
* alter database datafile ‘/prod/hyd/oradata/palikila01.dbf’ resize 60m;

1. Add new datafile in tablespace palikila

* alter tablespace palikila add datafile ‘/prod/hyd/oradata/palikila02.dbf’ size 30m;

**To restrict max size of datafile Palikila**

* alter database datafile ‘/prod/hyd/oradata/palikila01.dbf’ autoextend on maxsize 300 m;

**To know Block size**

* desc dba\_tablespaces;
* select TABLESPACE\_NAME,BLOCK\_SIZE from dba\_tablespaces;
* Shows tablespaces block size

**To change Block\_size**

* we have to change the parameter
* Show parameter cache
* Shows different parameters like this 2k,4K,8k,16k,32k
* Alter system set db\_32k\_Cache\_Size=100m scope=both;
* Now we can create datafile with 32k block\_size.
* create tablespace kota datafile ‘/prod/hyd/oradata/kota01.dbf’ size 30m blocksize 32768;

**To check tablespace size and Auto extensible**

* desc dba\_data\_files;
* select FILE\_NAME,BYTES/1024/1024,MAXBYTES/1024/1024,AUTOEXTENSIBLE from dba\_data\_files;

**To AUTO EXTEND ON/OFF datafile**

* Alter database datafile ‘/prod/hyd/oradata/ palikila01.dbf’ autoextend on;
* Alter database datafile ‘/prod/hyd/oradata/ palikila01.dbf’ autoextend off;

**TABLESAPCE LOGGING & NO LOGGING:**

* desc dba\_tablespaces
* if tablespace is in nologging the data in tablespace will not convert as archives.
* If nologging datafile goes to recover state we cannot recover it.

**We can create nologging tablespace and change to logging :**

* create tablespace mouli datafile ‘ /prod/hyd/oradata/mouli01.dbf’ size 30m nologging;

**To change from no logging to loggin**

* alter tablespace mouli loggin;

**To check Tablespace status and online / offline**

* we can’t offline system and undo tablespace and datafile.
* online - read write.
* read only - we cannot load any data.we. Can perform only select operations
* select tablespace \_name,status from dba\_tablespaces;

**To offline**

* alter tablespace mouli offline;

**To online**

* alter tablespace mouli online;

**Datafile online / offline**

* we can’t offline system and undo tablespace and datafile
* desc dba\_data\_files
* desc v$datafile;
* select file\_name, status from dba\_data\_files;
* select file#,name,status from v$datafile;
* to offline the datafile db be must be enable with archive log mode.

**To offline**

* alter database datafile ‘/prod/hyd/oradata/mouli01.dbf’ offline;
* IT goes to recover state
* recover datafile ‘/prod/hyd/oradata/mouli01.dbf’;

(OR)

* recover datafile 1;
* Now datafile goes to offline state :

**To online**

* alter database datafile ‘/prod/hyd/oradata/mouli01.dbf’ online;

**Datafile rename**

\* we can rename datafile by two methods.

1) by offline tablespace method

2) Datafile offline method

* better to do datafile offline method.
* to offline the datafile db be must be enable with archive log mode.

**In 11g:**

**datafile offline method**:

* to offline the datafile db be must be enable with archive log mode
* select name, status from v$datafile;
* Offline the datafile which we want to rename
* alter database datafile ‘/path/.dbf’ offline;
* datafile goes to recover state .
* recover it .
* recover datafile ‘ /path/.dbf’;
* now it goes to offline state .
* **shut down the database**.
* **s**hut immediate;

**move data files from source to required**

* mv /prod/hyd/oradata/palikila01.dbf /prod/hyd/oradata/palikila01.dbf
* start database in mount state
* Startup mount;

**Update in (CF)**

* alter database rename file ‘/prod/hyd/oradata/palikila01.dbf’ to ‘/prod/hyd/oradata/palikila\_01.dbf’;
* online the datafile.
* alter database datafile ‘/prod/hyd/oradata/mouli01.dbf’ online;

**In 12c & 19c**

**renaming the datafile**

* select name,status from v$ datafile;
* alter database move datafile ‘/prod/hyd/oradata/palikila01.dbf’ to ‘/prod/hyd/oradata/palikila\_01.dbf’;

**Extent management**:

* desc dba\_tablespace
* Select Tablespace\_name,Extent\_managemet from dba\_tablespaces;
* in 11g system tablespace is in DICTIONARY state.
* From 12c all the tablespaces are local.

**Based on extent management tablespace are of 2 types:**

1. Dictionary managed tablespace
2. Locally managed tablespace

* by default all the tablespaces are build locally.
* every tablespace has a header and some body.
* If it is locally managed tablespace the all extents allocation information is stored in the header.
* When the extents information is stored in system tablespace it is discriminate managed tablespace.
* Disctionary tablespace requires more I/O OPERATIONS

**Dictionary:**

* based on the extents allocation information.(logical definition)
* logical definition is collection of contents segments
* Segments is a collection of extents.
* Extents is collection of contiguous oracle blocks

**Create Dictionary managed Tablespace**:

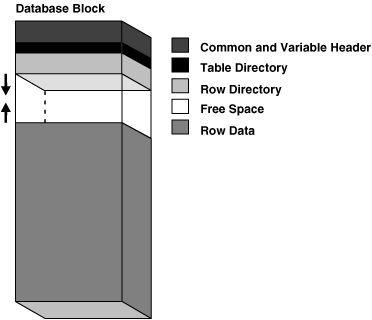
* create tablespace mouli datafile ‘/path/mouli01.dbf’ size 30m extent management dictionary;

**Convert Dictionary to locally**:

* exec dbms\_space\_admin.Tablespace\_migrate\_To\_local(‘MOULI’);

**Data blocks:**

* All the data in the Oracle database is stored in blocks.
* Oracle does not store a whole table in one block and return it from there directly.
* A block can have a size from 2KB to 32KB but generally, it is set to 8KB by default.
* A block is the smallest logical unit of the database to store the data.
* Data block size is usually multiple Operating System block size.
* This size is specified by the DBAs on the database installation.



#### **Header (Common and Variable)**

* The header contains general block information, such as the block address and the type of segment.

#### **Table Directory**

* This contains information about the tables having rows in this block.

#### **Row Directory**

* This contains row information about the actual rows in the block.

#### **Free Space**

* Available space in data block for additional row or update of row which require more space.

#### **Row Data (used data)**

* Contains table or index data. First three component of data block (Header, Table & Row directory) collectively known as **overhead**.

**PCT FREE :(block parameter)**

* **PCTFREE** is a block storage parameter used to specify how much space should be left in a database block for future updates.
* For example, **PCTFREE=10**, Oracle will keep on adding new rows to a block until it is 90% full. This leaves 10% for future updates.

**PCT USED :(block parameter)**

* **PCTUSED** is a storage parameter in oracle which specifies when a database block is empty enough for oracle to add it to the free list.
* When the percentage of the used space in a block is greater than the PCTUSED parameter, Oracle will not add new rows to the block. The default settings for all oracle tables are **PCTUSED=40**.

**Row Chaining:**

* When the data to too large to insert into a single block. Then oracle uses multiple blocks together holds the data.

**Row Migration:**

* When the block free space is not enough to hold the data. Then that data will migrate to another block where enough free space is available.