Managing UNDO

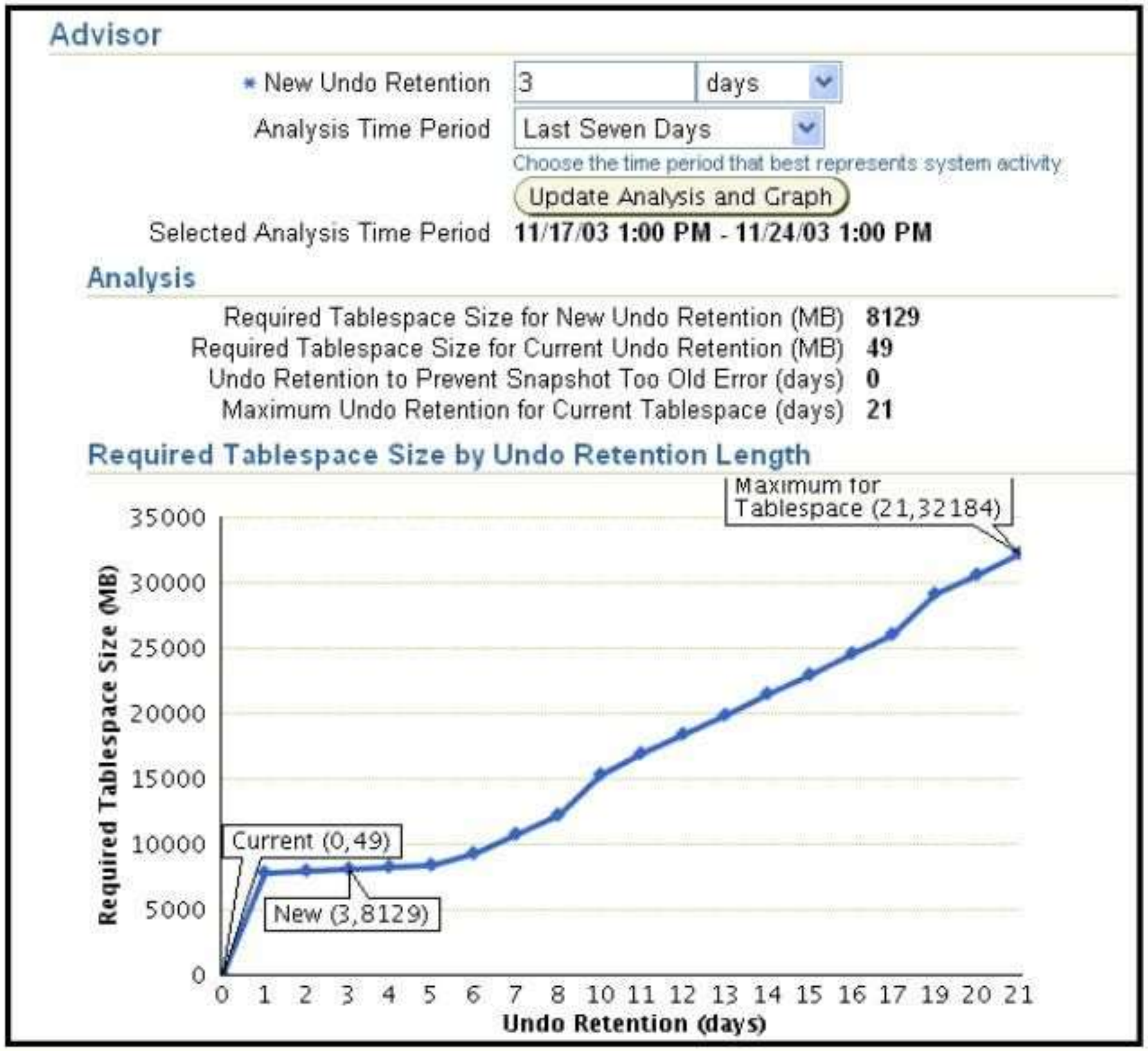
Automatic Undo Management Concepts

- Rollback data is managed by means of an undo tablespace.
- Allocate enough disk space for the workload of each instance in the undo tablespace, versus allocating rollback segments in different sizes.
- The notion of a single `SYSTEM` rollback segment is retained:
  - Created automatically within the `SYSTEM` tablespace
  - Automatically managed
  - Cannot be taken offline

Data Dictionary Views to Support Automatic Undo Management

- `V$UNDOSTAT` contains information about how rollback segments are used by the current instance. It is available for both `MANUAL` and `AUTO` mode.
- `DBA_UNDO_EXTENTS` shows the commit time for each extent in the undo tablespace.
- You can still use the `V$ROLLSTAT` and `V$TRANSACTION` views in AUM mode.

Using the Undo Advisor



Advisor

* New Undo Retention	3 days
Analysis Time Period	Last Seven Days
	Choose the time period that best represents system activity
	Update Analysis and Graph
Selected Analysis Time Period	**11/17/03 1:00 PM - 11/24/03 1:00 PM**

Analysis

Required Tablespace Size for New Undo Retention (MB)	**8129**
Required Tablespace Size for Current Undo Retention (MB)	**49**
Undo Retention to Prevent Snapshot Too Old Error (days)	**0**
Maximum Undo Retention for Current Tablespace (days)	**21**

Required Tablespace Size by Undo Retention Length

Understanding Automatic Undo Management

Every database must have a method for handling undo information:

- Required for read consistency
- Required to undo uncommitted transactions
- Required for instance recovery
- Manual management uses rollback segments.
- Automatic management uses undo tablespace.

Automatic Undo Retention Tuning

- Proactive tuning
  - Undo retention is tuned for longest-running query.
  - Query duration information is collected every 30 seconds.
- Reactive tuning
  - Undo retention is gradually lowered under space pressure.
  - Oldest unexpired extents are used first.
  - Undo retention never goes below either `UNDO_RETENTION` or 15 minutes (whichever is less).
- Enabled by default

Objectives

After completing this lesson, you should be able to do the following:

- Configure the database to use automatic undo management
- Monitor undo usage
- Convert from manual rollback to automatic undo
- Use the Undo Advisor

Altering an Undo Tablespace

- The `ALTER TABLESPACE` command can be used to make changes to undo tablespaces.
- Most parameters are system managed.
- The following example adds another data file to the undo tablespace:

```
SQL> ALTER TABLESPACE undotbs_1
  2  ADD DATAFILE 'undotbs_2.dbf'
  3  AUTOEXTEND ON;
```

Configuring Automatic Undo Management

- Automatic Undo Management (AUM) simplifies and automates rollback segment management.
- You can manage rollback segments automatically or manually by choosing a rollback mode.
- Set the `UNDO_MANAGEMENT` parameter:
  - `AUTO`: The instance manages undo segments automatically.
  - `MANUAL`: You must create and manage rollback segments manually (this is the default).

Creating an Undo Tablespace at Database Creation Time

- An undo tablespace can be created if the instance is started in AUM mode.
- If you do not specify an `UNDO TABLESPACE` clause, an undo tablespace with the name `SYS_UNDOTBS` is created:
  - Default size: 10 MB, `AUTOEXTEND ON`
  - Default file name: `o1_mf_sys_undo_n_.dbf`

```
SQL> CREATE DATABASE
  2  UNDO TABLESPACE undotbs01
  3  DATAFILE SIZE 50M;
```

Summary

In this lesson, you should have learned how to:

- Configure your database to use automatic undo management
- Monitor undo usage
- Use the Undo Advisor
- Convert from manual rollback to automatic undo

Dropping an Undo Tablespace

```
SQL> DROP TABLESPACE undotbs_2;
```

- **This command has an implicit `INCLUDING CONTENTS` clause.**
- **You can drop an undo tablespace only if it is not currently used by any instance.**
- **Readers needing information from dropped undo tablespaces may get `ORA-1555` error messages.**

Specifying Guaranteed Undo Retention

```
SQL> CREATE UNDO TABLESPACE undotbs1
  2  DATAFILE 'undotbs01.dbf'
  3  SIZE 100M AUTOEXTEND ON
  4  RETENTION GUARANTEE;
```

```
SQL> SELECT tablespace_name, RETENTION
  2  FROM   dba_tablespaces;

TABLESPACE_NAME RETENTION
--------------- ---------
UNDOTBS1        GUARANTEE
...             ...
```

```
SQL> ALTER TABLESPACE undotbs1
  2> RETENTION NOGUARANTEE;
```

Specifying the Mode for Undo Space Management

- **Starting in AUM mode:**
  - `UNDO_MANAGEMENT = AUTO`
  - `UNDO_TABLESPACE`: Specifies a particular undo tablespace to be used; if it does not exist, an error is raised (dynamic parameter).
  - If AUM is chosen and no undo tablespace is specified, the Oracle server uses the first available one; if none are available, the `SYSTEM` rollback segment is used.
- **Starting in Rollback Segment Undo (RBU) mode:**
  - `UNDO_MANAGEMENT = MANUAL` (the default) or
  - Leave old initialization file unchanged

Converting from Rollback to Undo

To convert to automatic undo:

1. Create an undo tablespace.
2. Set `UNDO_TABLESPACE` and `UNDO_MANAGEMENT`.
3. Shut down and start up your instance.

Creating an Undo Tablespace

```
SQL> CREATE UNDO TABLESPACE undotbs1
  2  DATAFILE 'undotbs1.dbf' SIZE 50M;
```

An undo tablespace:

- **Can be specified at instance startup using the `UNDO_TABLESPACE` dynamic parameter**
- **Can only be used in the `AUTOMATIC` mode for storing undo information**
- **Is a permanent, locally managed tablespace, in read/write and logging mode**

Configuring Undo Retention

Undo retention specifies (in seconds) the amount of already-committed undo information to retain.

- Default value is 0 (automatic).
- Maximum value is 2^{32} seconds (more than 187 years).
- A setting of 0 indicates automatic undo retention mode.
- A setting greater than 0 is a minimum retention time.

```
UNDO_RETENTION=0
```

Sizing the Undo Tablespace

- The undo retention will be limited by the size of the undo tablespace.
- Estimate the minimum size the undo tablespace requires to honor an undo retention time by using this formula:

```
UndoSpace = UR * UPS + overhead
```

Retaining Undo Information

- The goal is to retain undo information until it is no longer needed.
- AUM does this by:
  - Adding a new state: Expired extents
  - Retaining inactive extents based on the value of the `UNDO_RETENTION` parameter
  - Adjusting the allocation algorithm to retain extents as long as possible

Using V$UNDOSTAT

This V$UNDOSTAT example shows undo space consumption for the previous week from time 16:07.

```
End-     Undo      Txn   Txn Query  Exten SSTooOld
Time   Blocks Concrcy Total   Len Stolen    Error
----- ------- -------- ----- ----- ------ --------
16:07     252       15   151    25      2        0
16:00     752       16  1467   150      0        0
15:50     873       21  1954    45      4        0
15:40    1187       45  3210   633     20        1
15:30    1120       28  2498  1202      5        0
15:20     882       22  2002    55      0        0
...
```

Switching Undo Tablespaces

- Only one undo tablespace can be used by an instance at the same time, except for a `PENDING OFFLINE UNDO` tablespace.
- Switching is performed by using the `ALTER SYSTEM` command.

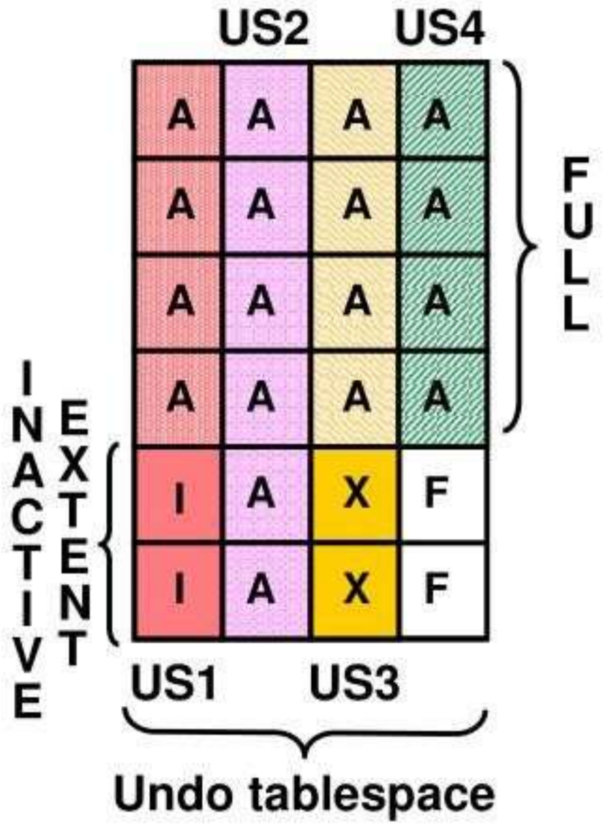
```
SQL> ALTER SYSTEM
  2  SET UNDO_TABLESPACE = undotbs02;
```

```
SQL> ALTER SYSTEM
  2  SET UNDO_TABLESPACE = '';
```

Dynamic Extents Transfer

Allocate extents by:

1. Using free extents
2. Using expired extents
3. Growing the tablespace
4. Using inactive extents
5. Give out of space error



A-Active I-Inactive X-Expired F-Free

USn stands for undo segment number n

Automatic Undo Management Concepts

- Rollback segments and undo segments are identical in purpose and very similar in behavior.
- With automatic undo management, you cannot create, alter, or drop undo segments.
- Undo segments have the same structure as rollback segments.
- Undo segments have the following characteristics:
  - Are automatically created
  - Use a modified allocation policy compared to Oracle8*i*
  - Support dynamic extents transfer
- SMON shrinks undo segments when necessary.