Oracle 11g DBA Fundamentals Overview

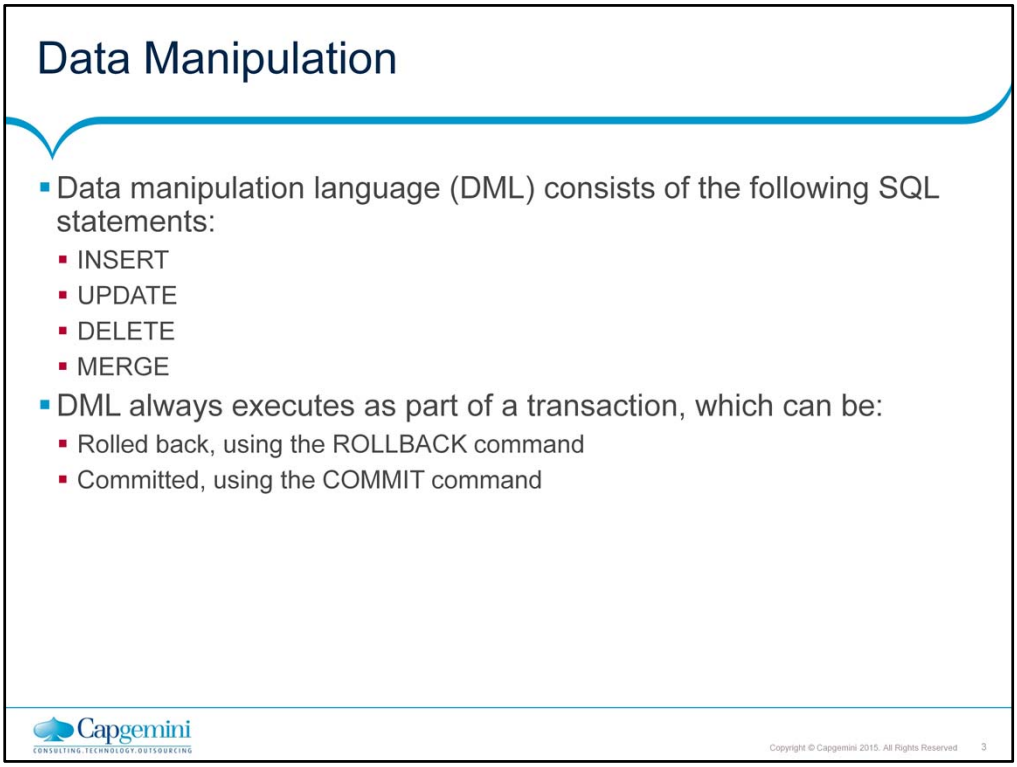
Lesson 07: Managing the Undo Tablespace

Objectives

- After completing this lesson, you should be able to do the following:
  - Explain DML and undo data generation
  - Monitor and administer undo data
  - Describe the difference between undo data and redo data
  - Configure undo retention
  - Guarantee undo retention
  - Use the Undo Advisor

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 2

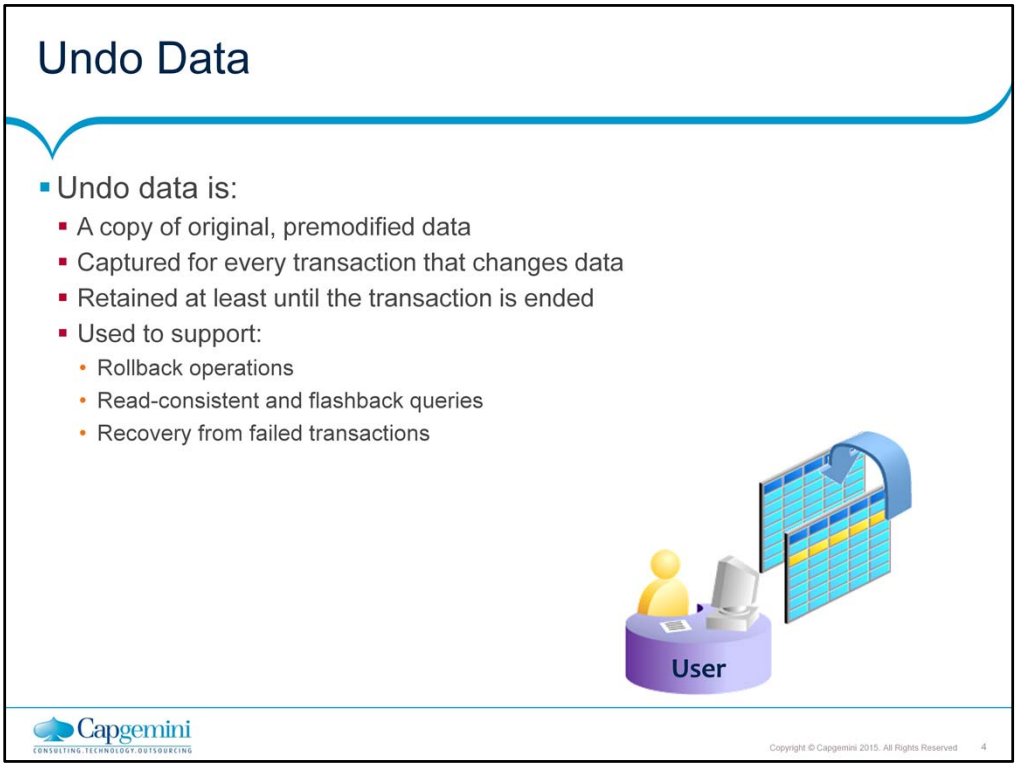


Data Manipulation

Data is manipulated, or modified, by the DML class of SQL statements: INSERT, UPDATE, DELETE, and MERGE. These statements execute as part of a transaction, which starts with the first successful DML statement and ends with either a COMMIT or ROLLBACK command. A transaction is either entirely committed or entirely rolled back.

Rollback may also occur if there is a process or system failure.

Note: The MERGE command performs a combination of inserts and updates to merge data from one table into another. It is covered in the lesson titled “Managing Data and Concurrency.”



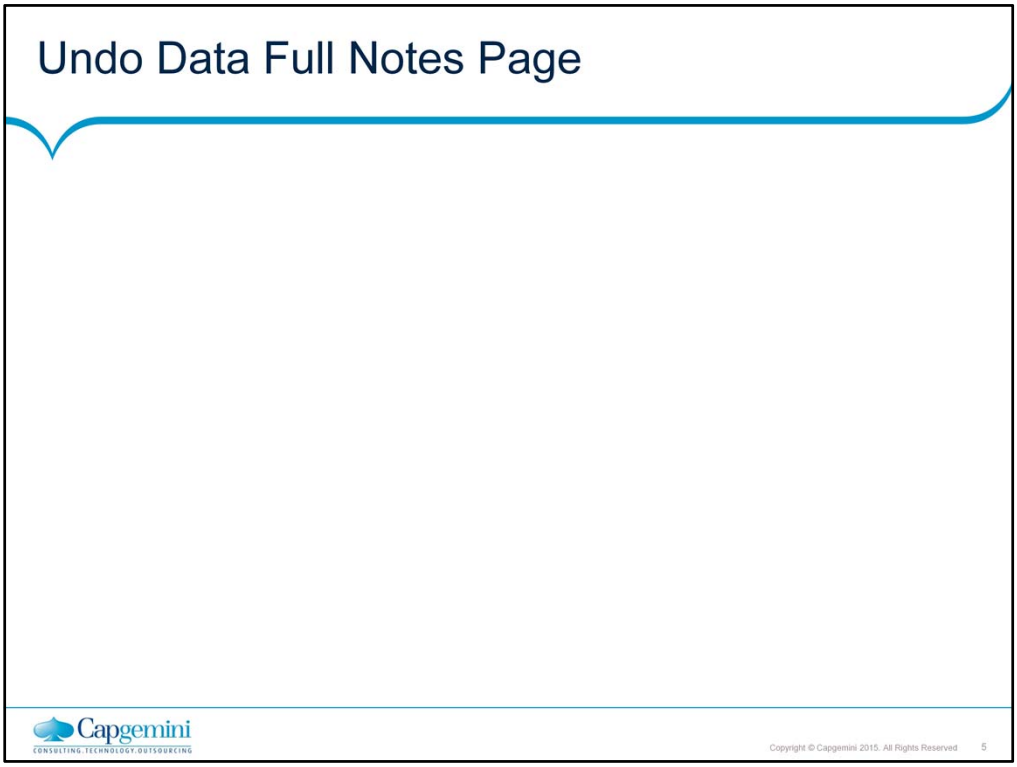
Undo Data

The Oracle database saves the old value (undo data) when a process changes data in a database. It stores the data as it existed before being modified. Capturing undo data enables you to roll back your uncommitted data. Undo also supports read-consistent and flashback queries.

Read-consistent queries provide results that are consistent with the data as of the time a query started. For a read-consistent query to succeed, the original information must still exist as undo information. As long as the undo information is retained, the Oracle database can reconstruct data to satisfy read-consistent queries.

Flashback queries are queries that purposely ask for a version of the data as it existed at some time in the past. As long as undo information for that past time still exists, flashback queries can complete successfully.

Undo data is also used to recover from failed transactions. A failed transaction occurs when a user session ends abnormally (possibly because of network errors or a failure on the client computer) before the user decides to commit or roll back the transaction. Failed transactions may also occur when the instance crashes.



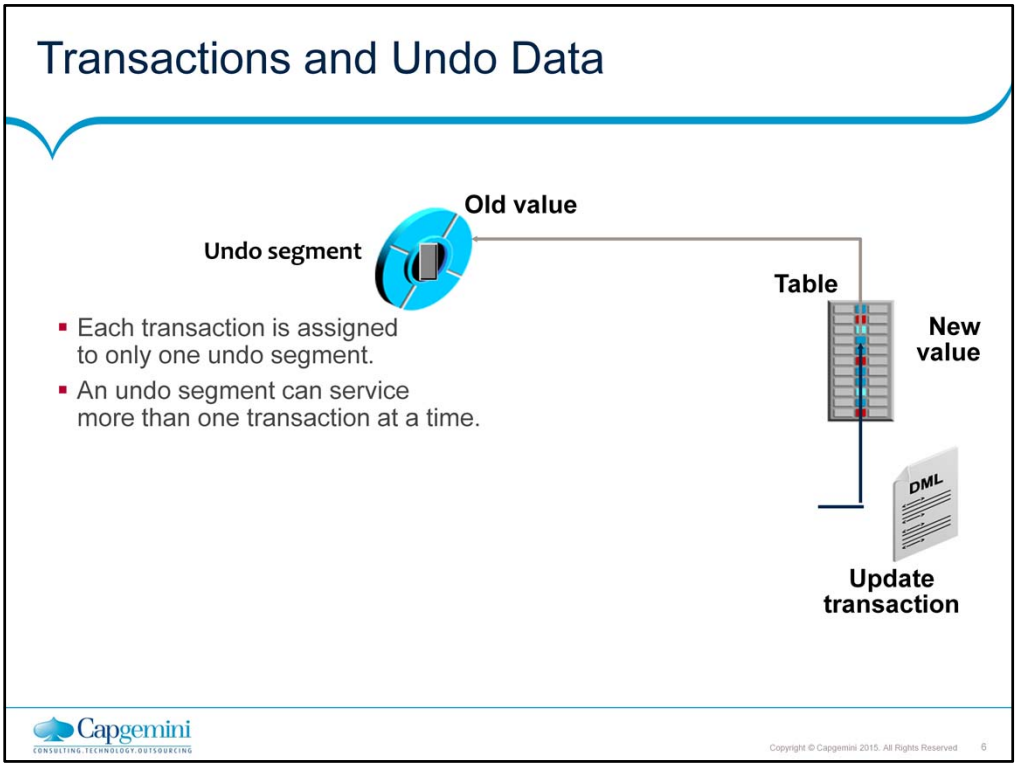
Undo Data (continued)

In case of a failed transaction, the safest behavior is chosen, and the Oracle database reverses all changes made by a user, restoring the original data.

Undo information is retained for all transactions, at least until the transaction is ended by:

- Users undoing the transaction (rolls back)
- Users ending a transaction (commits)
- User session abnormally terminating (rolls back)
- User session normally terminating with an exit (commits)

The amount of undo data that is retained and the time for which it is retained depend on the amount of database activity and the database configuration.



Transactions and Undo Data

When a transaction starts, it is assigned to an undo segment. Throughout the life of the transaction, when data is changed, the original (before the change) values are copied into the undo segment. You can see which transactions are assigned to which undo segments by checking the v$transaction dynamic performance view.

Undo segments are specialized segments that are automatically created by the instance as needed to support transactions. Like all segments, undo segments are made up of extents, which, in turn, consist of data blocks. Undo segments automatically grow and shrink as needed, acting as a circular storage buffer for their assigned transactions.

Transactions fill extents in their undo segments until a transaction is completed or all space is consumed. If an extent fills up and more space is needed, the transaction acquires that space from the next extent in the segment. After all extents have been consumed, the transaction either wraps around back into the first extent or requests a new extent to be allocated to the undo segment.

Note: Parallel DML operations can actually cause a transaction to use more than one undo segment. To learn more about parallel DML execution, see the Oracle Database Administrator's Guide 11g.

Storing Undo Information

- Undo information is stored in undo segments, which are, in turn, stored in an undo tablespace. Undo tablespaces:
  - Are used only for undo segments
  - Have special recovery considerations
  - May be associated with only a single instance
  - Require that only one of them be the current writable undo tablespace for a given instance at any given time

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

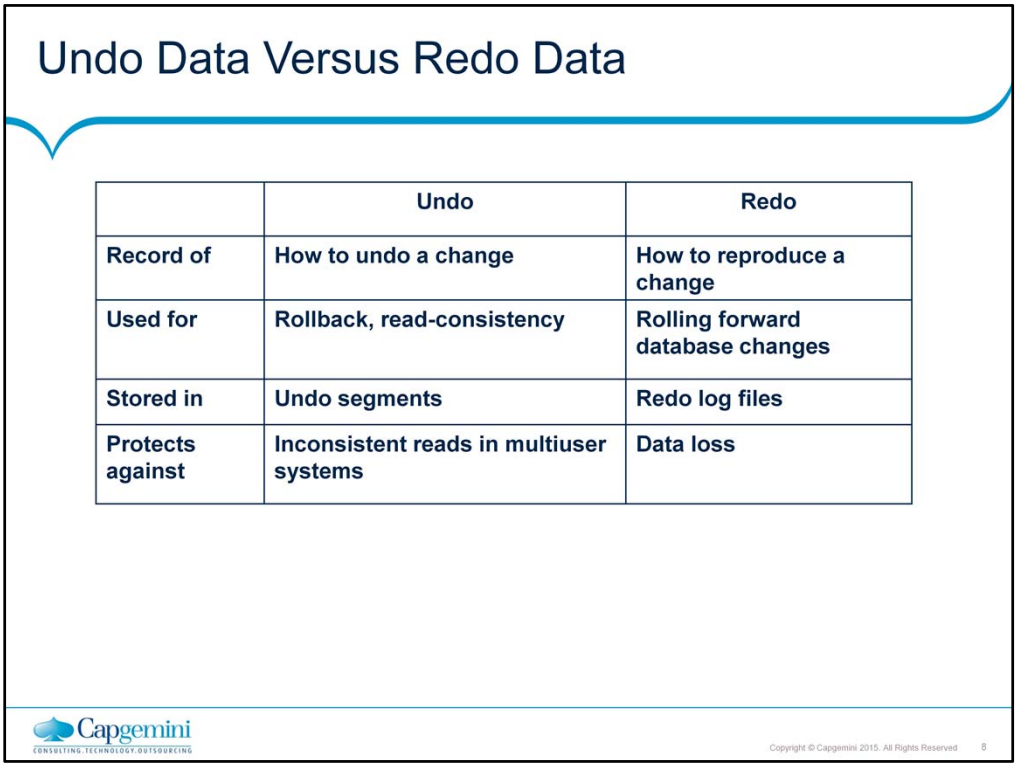
 7

Storing Undo Information

Undo segments can exist only in a specialized form of tablespace called an undo tablespace. Although a database may have many undo tablespaces, only one of them at a time can be designated as the current one to which undo data is written.

Undo segments are always owned by SYS. Because the segments act as a circular buffer, each segment has a minimum of two extents. The default maximum number of extents depends on the database block size, but is very high (32,765 for an 8-KB block size).

Undo tablespaces are permanent, locally managed tablespaces with automatic extent allocation. They are managed like any other tablespace with the exception of recovery. Because undo data is required to recover from failed transactions (such as those that may occur when an instance crashes), undo tablespaces can be recovered only while the instance is in the MOUNT state. Recovery considerations for undo tablespaces are covered in the lesson titled “Performing Database Recovery.”

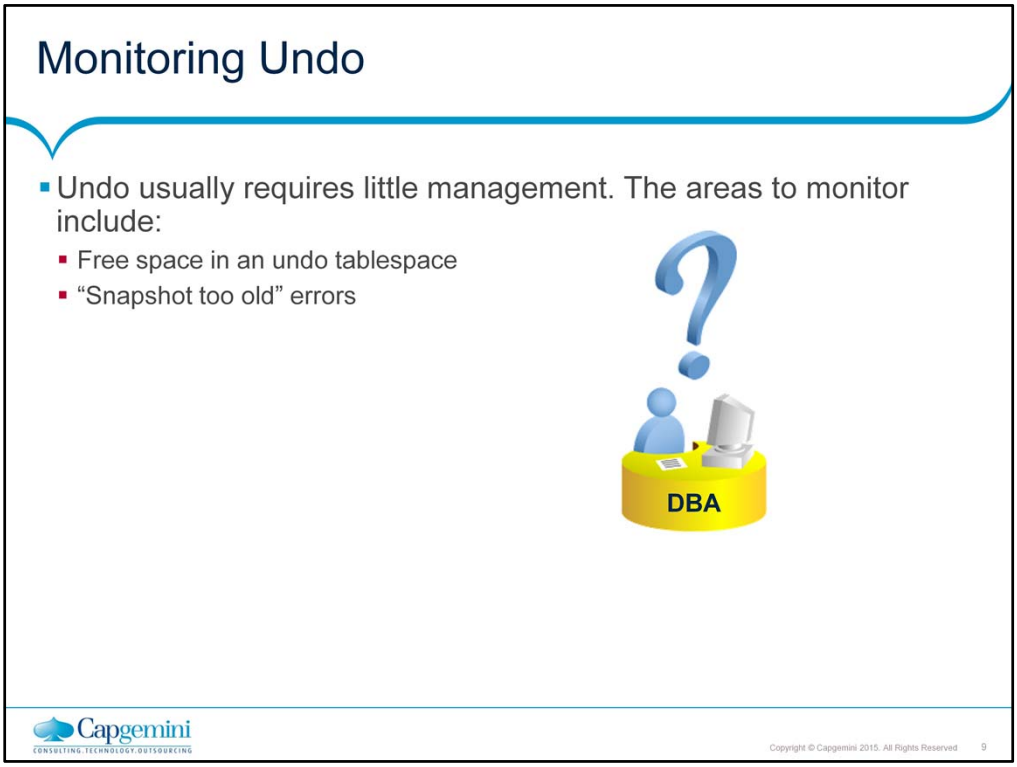


Undo Data Versus Redo Data

Undo data and redo data seem similar at first, but they serve different purposes. Undo data is needed in case there is the need to undo a change, and this occurs for read-consistency and rollback. Redo data is needed in case there is the need to perform the changes again, in case they are lost for some reason.

The process of committing entails a verification that the changes in the transaction have been written to the redo log file, which is persistent storage on the disk, as opposed to memory. In addition, it is typically multiplexed. So, there are multiple copies of the redo data on the disk. Even though the changes may not have yet been written to the data files where the table's blocks are actually stored, guaranteeing that the changes have been written to the redo log file is enough.

A power outage that occurs just before committed changes have been reflected into the redo files does not cause a problem because the transaction has been committed. So, when the system starts up again, it is able to roll forward any redo records that are not yet reflected in data files at the time of the outage.



Monitoring Undo

Most of the time, undo is managed automatically by the instance with little need for database administrator (DBA) intervention. A few things that may require administrator involvement include:

Insufficient space for undo

Users receiving the ORA-01555 snapshot too old error messages

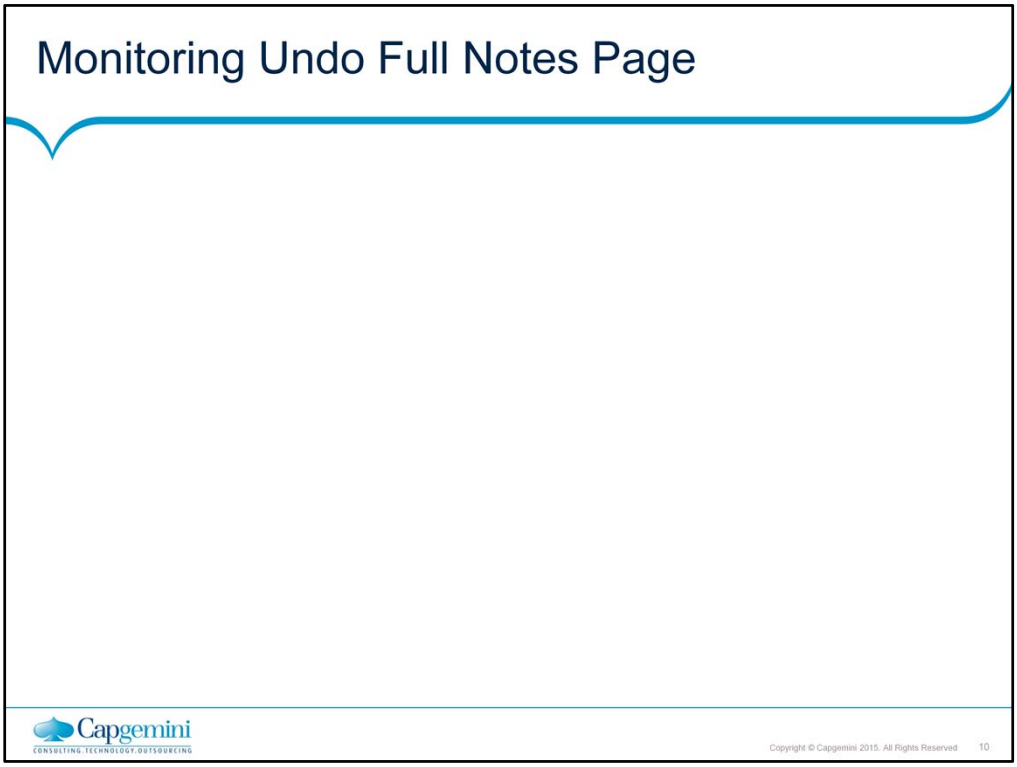
Undo information is always retained until a transaction ends. This means that if extremely large amounts of data are deleted or updated (Insert operations consume very little undo space because the original image of inserted data is a null value.) without being committed, the undo tablespace must be equally large to contain the original data. Imagine a case where a 50-GB table had all rows deleted with the following command:

SQL> DELETE FROM reallybigtable;

The undo tablespace would be required to make room for 50 GB of original information just in case the user who issued this statement changed his or her mind and wanted to roll back the change. When the undo tablespace runs out of room for undo data, users receive an error message such as the following:

ORA-01650: unable to extend rollback segment

Proactive monitoring detects space problems in an undo tablespace before they affect users.

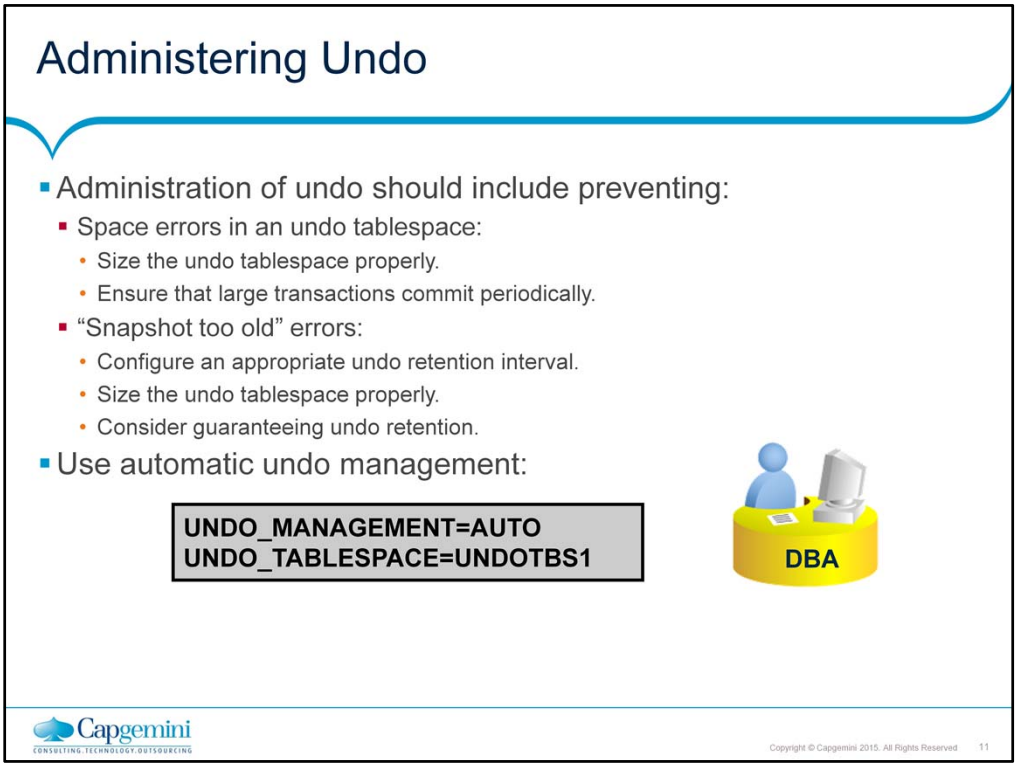


Monitoring Undo (continued)

Another problem that the administrator may encounter with undo information is when a query needs to access undo information that has already been overwritten. This may happen in a long-running or flashback query. When a query needs a “snapshot” of data as of some time in the past, and reconstructing that snapshot requires undo data that no longer exists, the query returns the following error:

ORA-01555: snapshot too old

This can happen because the Oracle database presents the user with a consistent view of the data as it exists at the time the query starts running. If there are uncommitted changes to the table being queried, the Oracle database reads the undo data to get the committed version of data. This is read consistency. If the query runs so long that in the meantime those modifications are indeed committed, and subsequently their undo data is released and overwritten, then the long-running query no longer can see a consistent view of the data as of when it first began to run. For this reason, undo retention should be configured to accommodate the longest-running query.



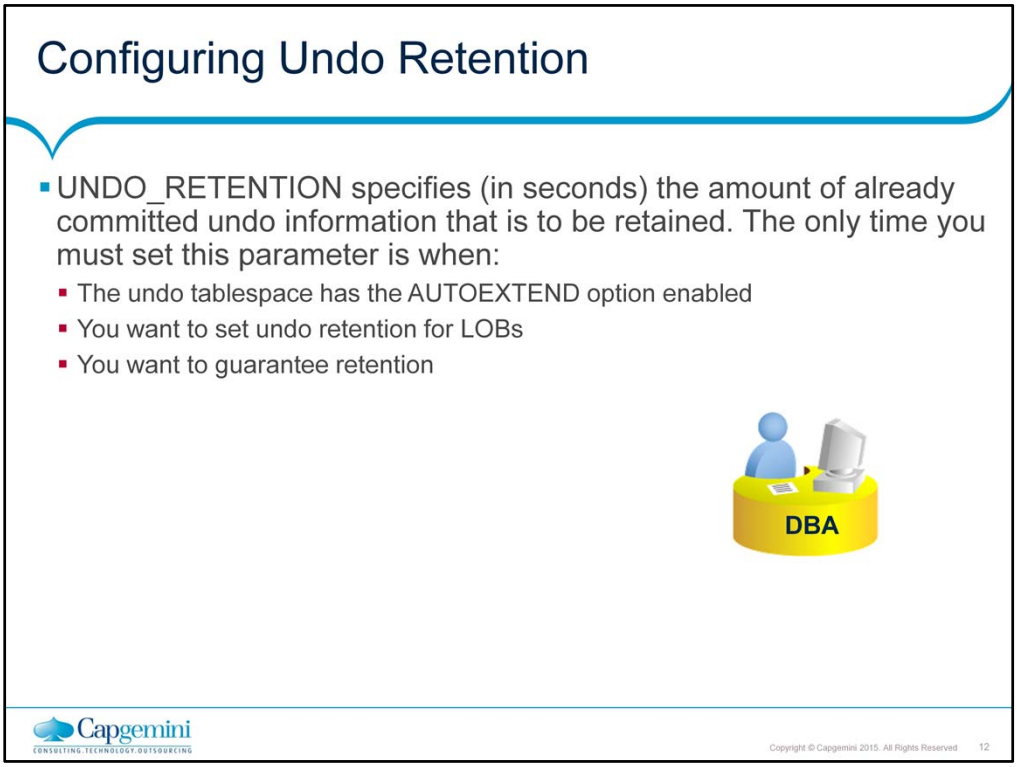
Administering Undo

It is recommended that you use automatic undo management, configured by setting the UNDO_MANAGEMENT initialization parameter to AUTO. Manual undo management is supported for backward compatibility with Oracle8i and earlier versions, but requires more DBA interaction.

With automatic undo management, the DBA manages undo at the tablespace level, controlling, with the UNDO_TABLESPACE initialization parameter, which undo tablespace an instance uses. After selecting the undo tablespace, the administrator need worry only about providing sufficient space and configuring an undo retention interval.

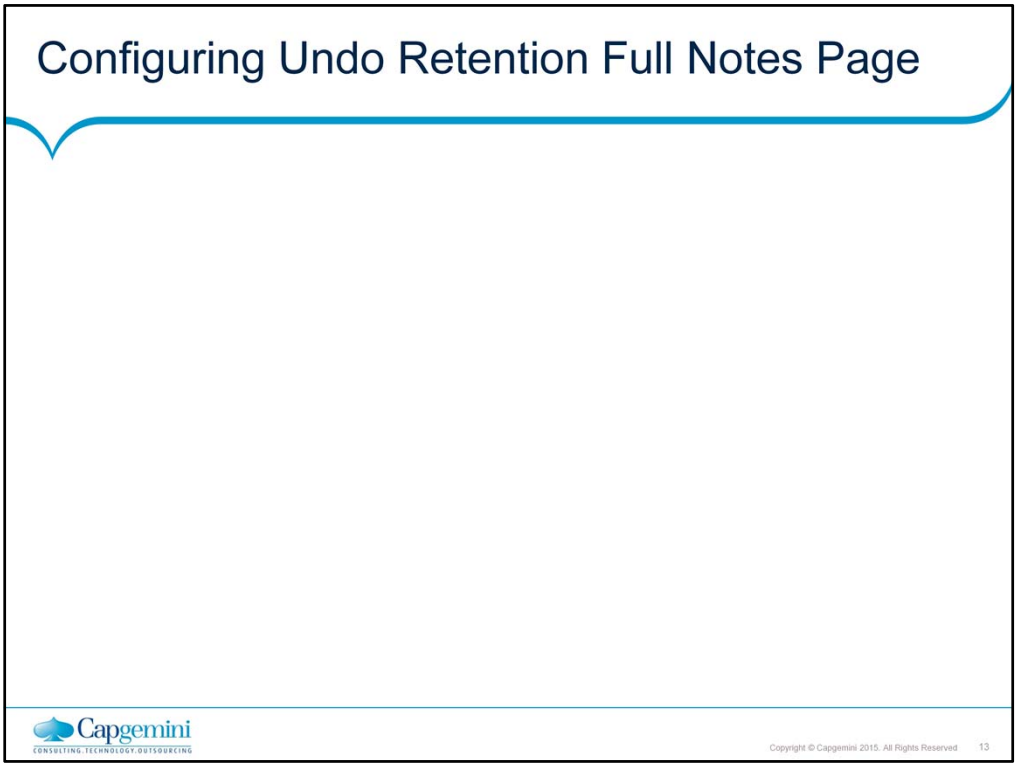
With manual management, the DBA must also consider:

Segment sizing, including maximum extents and extent sizing
Identifying and eliminating blocking transactions
Creating enough rollback segments to handle transactions (In manual mode, undo segments are known as rollback segments.)
Choosing a tablespace to contain the rollback segments (Undo tablespaces are used only with automatic undo management.)



Configuring Undo Retention

UNDO_RETENTION specifies (in seconds) the low threshold value of undo retention. For the AUTOEXTEND undo tablespaces, the system retains undo for at least the time specified in this parameter, and automatically tunes the undo retention period to meet the undo requirements of the queries. For fixed-size undo tablespaces, the system automatically tunes for the maximum possible undo retention period on the basis of undo tablespace size and usage history; it ignores UNDO_RETENTION unless retention guarantee is enabled. So for automatic undo management, for the three cases listed, the UNDO_RETENTION setting is used. In cases other than these three, this parameter is ignored.



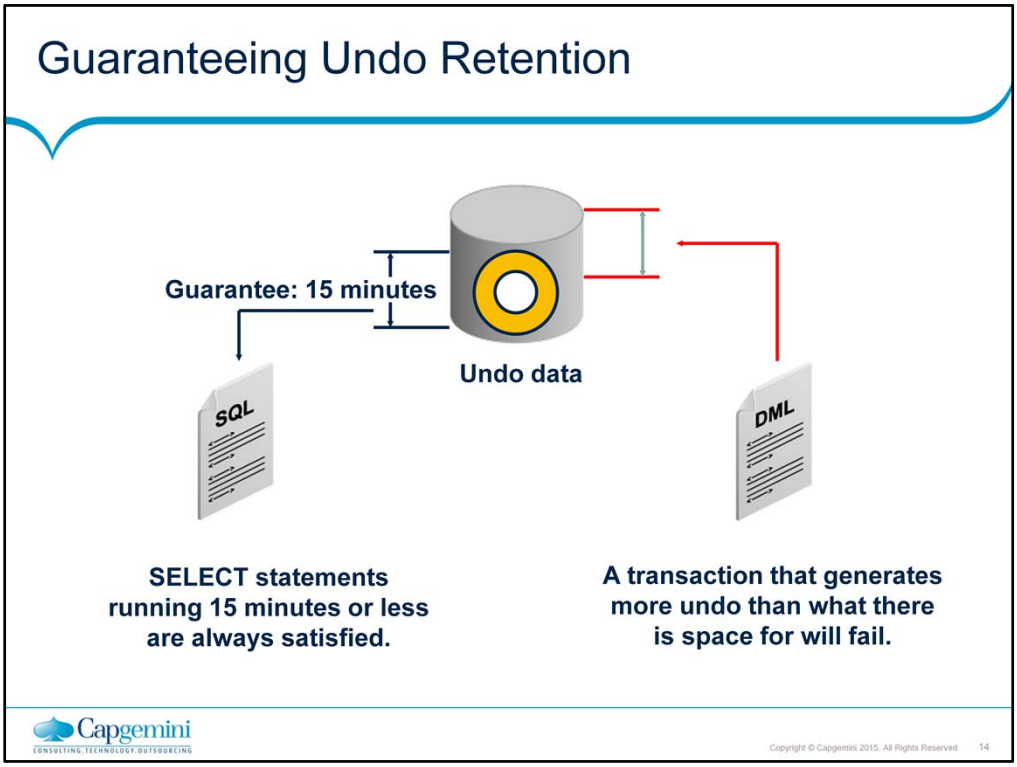
Configuring Undo Retention (continued)

Undo information is divided into three categories:

Uncommitted undo information: Supports a currently running transaction, and it is required if a user wants to roll back or if the transaction has failed. Uncommitted undo information is never overwritten.

Committed undo information: Is no longer needed to support a running transaction, but it is still needed to meet the undo retention interval. It is also known as “unexpired” undo information. Committed undo information is retained when possible without causing an active transaction to fail because of lack of space.

Expired undo information: Is no longer needed to support a running transaction. Expired undo information is overwritten when space is required by an active transaction.



Guaranteeing Undo Retention

The default undo behavior is to overwrite committed transactions that have not yet expired rather than to allow an active transaction to fail because of lack of undo space.

This behavior can be changed by guaranteeing retention. With guaranteed retention, undo retention settings are enforced even if they cause transactions to fail.

RETENTION GUARANTEE is a tablespace attribute rather than an initialization parameter. This attribute can be changed only with SQL command-line statements. The syntax to change an undo tablespace to guarantee retention is:

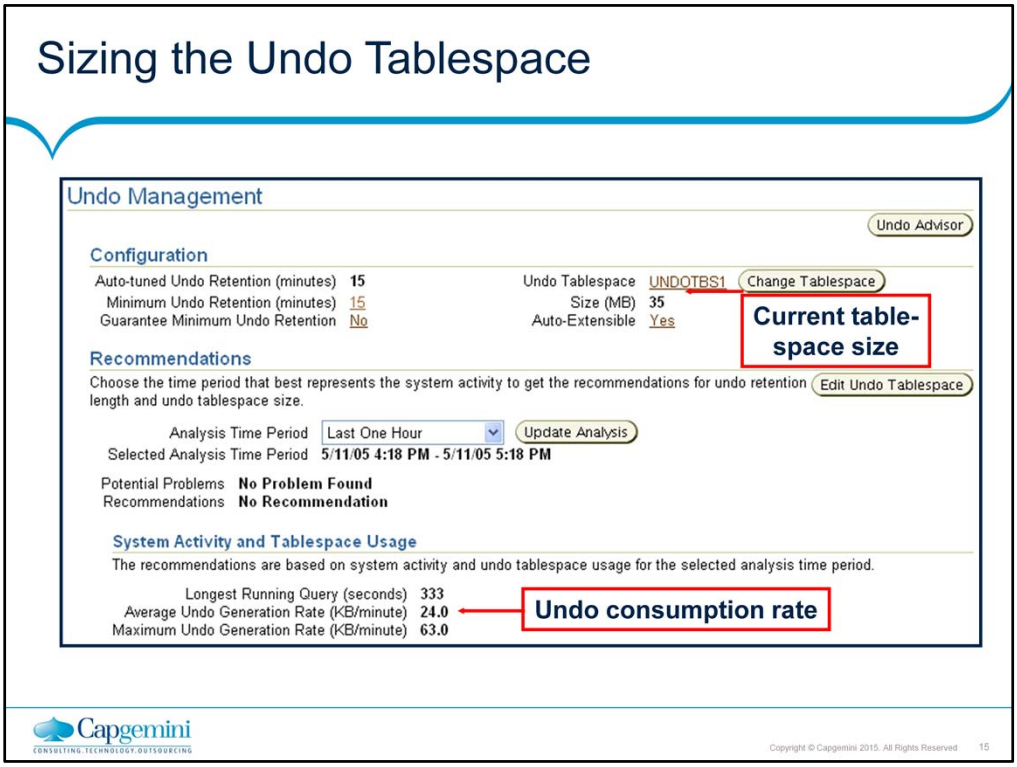
```
SQL> ALTER TABLESPACE undotbs1
RETENTION GUARANTEE;
```

To return a guaranteed undo tablespace to its normal setting, use the following command:

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

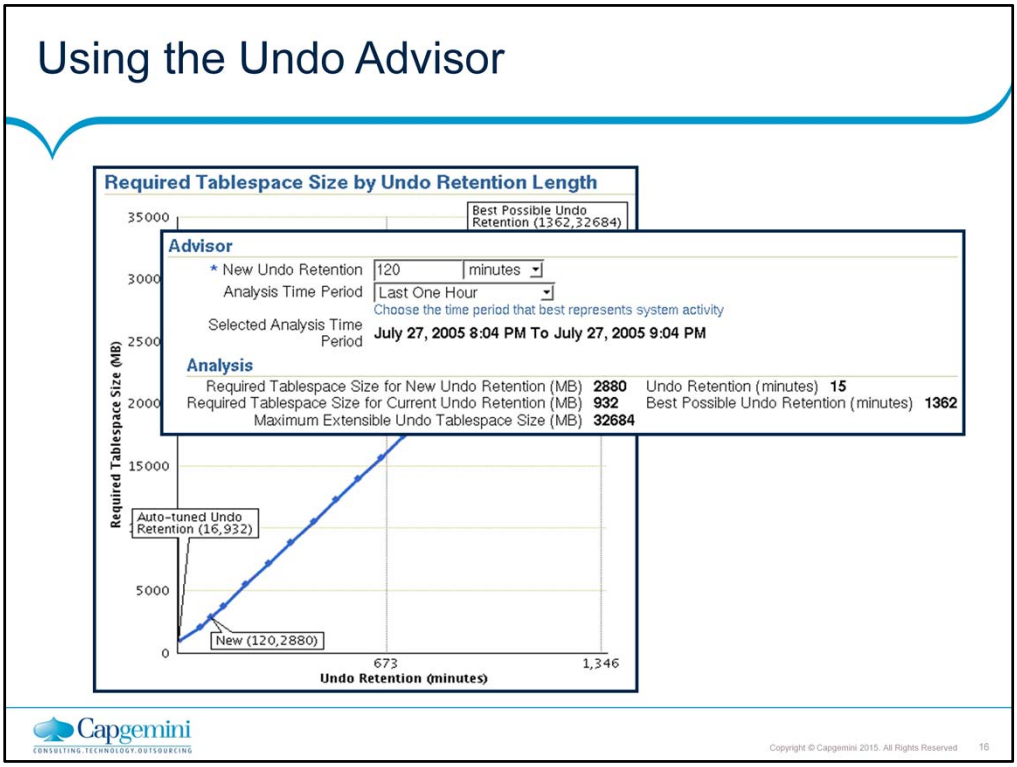
The retention guarantee applies only to undo tablespaces. Attempts to set it on a non-undo tablespace result in the following error:

```
SQL> ALTER TABLESPACE example
RETENTION GUARANTEE;
ERROR at line 1:
ORA-30044: 'Retention' can only specified for
undo tablespace
```



Sizing the Undo Tablespace

Undo tablespaces must be sized so that they can contain the original information for all transactions. Clicking the Undo Management link on the Enterprise Manager's Administration page reveals an overview of system undo, including current settings, undo consumption per minute, and the length of the longest-running query observed during a given time period. Data files belonging to an undo tablespace can automatically extend when they run out of free space. Oracle Corporation recommends that data files that are associated with undo tablespaces, unlike other tablespaces, should not have automatic extension enabled. When first determining undo space requirements, you may want to enable automatic extension of the data files, but after you have properly sized the tablespace, you must disable it. Disabling automatic extension in an undo tablespace's data files prevents a single user from inadvertently consuming large amounts of disk space by neglecting to commit transactions.



Using the Undo Advisor

The Undo Advisor is accessed through the Undo Management properties page. It provides an estimate of the undo tablespace size required to satisfy a given undo retention.

Enter the desired retention period, and the analysis region of the advisor displays the tablespace size required to support the retention period. You can also click a point on the graph to see the tablespace size required to support the selected period.

After you have selected an undo retention period, click OK to implement the new retention period.

Summary

- In this lesson, you should have learned how to:
  - Explain DML and undo data generation
  - Monitor and administer undo segments
  - Describe the difference between undo data and redo data
  - Configure undo retention
  - Guarantee undo retention
  - Use the Undo Advisor

Capgemini
CONSULTING.TECHNOLOGY.OUTSOURCING

 17