Trân trọng cảm ơn MyloveisThuy đã cung cấp cuốn sách này.

MỤC LỤC

Chương 1. CÁC ĐIỂM MỚI TRONG ORACLE 9i

Phiên bản Oracle9*i* Release 1 (9.0.1) được đưa ra thị trường vào đầu năm 2001 và được cải tiến, bổ sung thêm một số chức năng, đặc điểm mới. Các đặc điểm này đã làm cho việc quản lý database trở nên mềm dẻo, linh hoạt và hiệu quả hơn. Dưới đây, ta sẽ xem xét một số đặc điểm mới này:

Cho phép định nghĩa lại cấu trúc của tables đang online

Chức năng này được cung cấp trong gói package `DBMS_REDEFINITION` do Oracle cung cấp, cho phép người dùng có thể định nghĩa lại cấu trúc của một table thông quan câu lệnh DML ngay khi nó đang online. Với các phiên bản trước, Oracle 8i, ta cũng có thể định nghĩa lại cấu trúc của table nhưng trước đó cần phải đặt chế độ offline cho nó. Điều này không thuận tiện cho việc quản trị.

Cho phép thực hiện lệnh `ANALYZE VALIDATE STRUCTURE` tức thời

Có thể thực hiện lệnh `ANALYZE` để tối ưu table ngay cả khi đang có lệnh DML thực hiện trên table.

Điều khiển lưu trữ sau

Oracle cung cấp cơ chế điều khiển switching đối với các online redo log group dựa theo thời gian (`time-based`). Trong cấu hình `primary/standby`, tất cả các noncurrent logs tại primary site sẽ được lưu trữ rồi vận chuyển tới standby database. Việc này sẽ hiệu quả khi hạn chế số lượng các redo records.

Tạm treo database

Oracle9*i* cung cấp chức năng `suspend/resume`. Quản trị viên sử dụng lệnh `ALTER SYSTEM SUSPEND` để tạm treo database, dừng mọi thao tác truy xuất vào ra đối với các datafiles và control files. Khi database ở trạng thái tạm treo, các thao tác vào ra (`I/O operations`) đang thực hiện sẽ được kết thúc và những truy cập vào database mới phát sinh sẽ được đẩy vào queue. Thực hiện lệnh `ALTER SYSTEM RESUME` để khôi phục lại tình trạng bình thường của database.

Đặt chế độ hoạt động tĩnh cho database

Oracle9*i* cho phép đưa database vào chế độ hoạt động tĩnh(quiesced state). Theo đó chỉ có các DBA transactions, queries, và các lệnh PL/SQL là được phép thực hiện. Trạng thái này cho phép người dùng thực hiện các thao tác quản trị một cách an toàn. Sự dụng câu lệnh `ALTER SYSTEM QUIESCE RESTRICTED` để đưa database về chế độ hoạt động tĩnh.

Khả năng khôi phục và cấp phát lại không gian

Oracle sẽ tự động thực hiện tạm treo (suspending) và sau đó khôi phục (resuming) lại việc thực hiện các thao tác database tốn kém (large database operations) trong trường hợp có lỗi cấp phát không gian. Nhờ vậy mà Oracle database server sẽ có thể tự thực hiện các thao tác hợp lý thay vì việc trả về thông báo lỗi như ở các phiên bản trước. Sau khi các lỗi này được khắc phục database lại được tự động khôi phục bình thường.

Cho phép lưu trữ trên nhiều đích lưu trữ

Số lượng đích lưu trữ tối đa mà ta có thể sử dụng để lưu trữ các online redo log được tăng lên từ 5 tới 10.

Tự động quản lý vùng không gian

Oracle9*i* cho phép quản lý tự động việc giải phóng và sử dụng các vùng không gian có trong các segments được lưu trữ trong các locally managed tablespaces thông qua việc sử dụng mệnh đề `SEGMENT SPACE MANAGEMENT` có trong câu lệnh `CREATE TABLESPACE`. Quản trị viên có thể sử dụng chế độ `AUTO` hoặc `MANUAL` để chỉ rõ kiểu quản lý không gian mà Oracle sẽ sử dụng.

Cập nhật lại các global indexes mỗi khi thực hiện thao tác bảo trì partition

Theo mặc định, có thể có một vài phần của một bảng được phân khu (partitioned tables) ở trạng thái không sử dụng (đánh dấu `UNUSABLE`) sẽ được nạp vào trong global indexes. Và ta cần xây dựng lại (rebuild) toàn bộ global index. Oracle9*i* cho phép thực hiện tự động công việc rebuild này thông qua mệnh đề `UPDATE GLOBAL INDEX` có trong câu lệnh `ALTER TABLE` khi thực hiện bảo trì.

Cho phép sử dụng đồng thời nhiều kích cỡ block

Oracle cho phép sử dụng đồng thời nhiều kích cỡ blocks (multiple block sizes). Kích thước chuẩn (standard block size) được quy định trong tham số khởi tạo `DB_BLOCK_SIZE` nhưng cũng có thể mở rộng thêm 4 giá trị kích thước block phi chuẩn nữa (nonstandard block sizes). Các kích thước blocks phi chuẩn được chỉ rõ mỗi khi tạo tablespaces. Kích thước block chuẩn được sử dụng cho `SYSTEM` tablespace và hầu hết các tablespaces khác. Việc hỗ trợ sử dụng nhiều kích cỡ block sẽ cho phép thực hiện trao đổi các tablespaces của các database mà không có cùng một kích thước block.

Quản lý động buffer cache

Kích thước của buffer cache có trong vùng nhớ System Global Area được quản lý động. Điều này có nghĩa là giá trị của tham số `DB_BLOCK_BUFFERS` (trong file tham số khởi tạo) có thể được thay thế bởi giá trị có trong tham số khác, tham số `DB_CACHE_SIZE`. Trong Oracle 9i, buffer cache lại được phân chia thành nhiều bộ đệm con (subcaches) nếu có sử dụng chế độ multiple block sizes. Bốn giá trị kích cỡ block được chỉ ra trong 4 tham số `DB_nK_CACHE_SIZE` tương ứng .

Quản lý động vùng nhớ SGA

Các tham số khởi tạo có thể tác động tới kích cỡ của vùng nhớ SGA. Và ta có thể thay đổi kích cỡ của SGA dễ dàng thông qua câu lệnh `ALTER SYSTEM SET`.

Quản lý việc khôi phục (undo) tự động

Oracle sử dụng rollback segments để lưu trữ các thông tin cho khôi phục. Việc phục hồi (undo) bao gồm roll back, undo, và thay đổi (changes) đối với database mỗi khi cần. Oracle 9i cho phép ta tạo riêng một undo tablespace để lưu trữ các thông tin phục hồi này. Việc sử

dụng undo tablespace sẽ làm giảm bớt tính phức tạp của việc quản trị vùng không gian rollback segment, và cho phép phục hồi lại các thông tin dài mà không sợ bị trùng lên nhau.

Quản lý files trong Oracle

Một điểm mới trong Oracle 9i là quản lý files. Thông qua các tham số khởi tạo `DB_CREATE_FILE_DEST` và `DB_CREATE_ONLINE_LOG_DEST_ n` ta có thể chỉ ra cho hệ thống các đường dẫn cụ thể lưu trữ các file thuộc tablespace, online redo log file hay control file. Oracle luôn đảm bảo quản lý file duy nhất trong hệ thống.

Tự động xoá các datafiles

Oracle9*i* cung cấp một lựa chọn cho phép tự động xoá bỏ (remove) các datafiles mỗi khi tablespace tương ứng bị huỷ thông qua câu lệnh `DROP TABLESPAC`E. Tuỳ chọn tương tự trong câu lệnh `ALTER DATABASE TEMPFIL`E cũng được sử dụng để xoá các temporary file tương ứng.

Metadata API

Một `PL/SQL` package mới, `DBMS_METADATA.GET_DDL`, được đưa vào Oracle 9i cho phép ta lấy được các siêu dữ liệu (metadata) – Các thông tin tổng hợp về các schema object.

Các bảng ngoài - External tables

Oracle9*i* cho phép ta truy cập theo kiểu chỉ đọc các dữ liệu trong các bảng ngoài (external tables). External tables là các tables mà không nằm trong database, và có thể ở các khuôn dạng (format) nào đó. Câu lệnh `CREATE TABLE ... ORGANIZATION EXTERNAL` được sử dụng để chỉ ra metadata mô tả cho external table tương ứng. Oracle cung cấp điều khiển truy cập `ORACLE_LOADER`, qua đó cung cấp khả năng ánh xạ dữ liệu tương ứng với cú pháp lệnh trong control file.

Tăng cường cho constraint

Ta sử dụng mệnh đề `USING INDEX` trong câu lệnh `CREATE TABLE` hay `ALTER TABLE` để cho phép ta chỉ rõ index mỗi khi sử dụng ràng buộc unique key hay primary key. Thêm vào đó, ta cũng có thể ngăn cản việc huỷ (dropping).

File tham số trên server

Oracle lưu trữ các tham số khởi tạo cho session trong file tham số dưới khuôn dạng văn bản và được đặt tại các client machine.

Các tham số khởi tạo của server nằm trong file tham số trên server thường ở khuôn dạng nhị phân và có thể được lưu trong database.

Temporary tablespace mặc định

Có thể thêm vào mệnh mới `DEFAULT TEMPORARY TABLESPACE` vào câu lệnh `CREATE DATABASE` để cho phép tạo temporary tablespace ngay trong thời gian tạo database. Tablespace này sẽ được sử dụng như temporary tablespace mặc định.

Đặt tên cho transaction

Oracle cho phép ta gán tên cho mỗi một transaction. Tên của transaction rất có ích cho việc phân biệt giảm thiểu việc nhầm lẫn giữa các transactions.

Một số thay đổi trong Oracle Database Configuration Assistant

Oracle Database Configuration Assistant có một số thay đổi trong thiết kế. Theo đó, nó cung cấp các mẫu (templates) giúp cho việc tiết kiệm, giảm bớt việc định nghĩa các object trong database.

Người dùng cũng có thể tạo ra các mẫu này thông qua việc sửa đổi các mẫu có sẵn. Khi tạo database bằng công cụ Database Configuration Assistant ta cũng có thể thêm vào ngay hoặc sau đó các khuôn mẫu gọi là các Oracle's new Sample Schemas. Những schemas này là những ví dụ tài liệu cơ bản trong Oracle.

Quản lý việc sử dụng index

Ta thêm mệnh đề `MONITORING USAGE` vào trong câu lệnh `ALTER INDEX` để có thể xác định và quản lý index khi nó được thực hiện.

Liệt kê các phân vùng

Oracle 9i giới thiệu sử dụng liệt kê các phân vùng, nó cho phép ta chỉ ra một danh sách các giá trị rời rạc tương ứng với các partitioning column của mỗi phân vùng. Phương thức liệt kê phân vùng (list partitioning method) được đưa ra nhằm mục đích mô hình hoá dữ liệu phân tán đối với các giá trị rời rạc. Việc này khó thực hiện được bằng các phương pháp range partitioning (phân khu theo khoảng giá trị) hay hash partitioning (phân khu theo hàm băm).

Phân khu theo hàm băm cho các index-organized tables

Oracle 9i cho phép sử dụng phương pháp băm khi phân khu các index-organized tables. Ở các phiên bản trước, việc phân khu cho index-organized tables vẫn thực hiện được nhưng chỉ bằng phương pháp range method.

Xử lý các job queue process linh hoạt

Các job queue process được tạo một cách linh hoạt và nó chỉ cần tới số hiệu của processes được tạo để thực hiện các jobs của process đó đang sẵn sàng cho việc thực hiện. Tiến trình nền (background process) có tên là `CJQ` sẽ đảm nhiệm công việc này.

Điểm mới trong Database Resource Manager

Có một số chức năng mới được thêm vào Database Resource Manager:

- Có khả năng tạo một active session pool, là nơi lưu chứa được một số lượng lớn nhất các user sessions đồng thời đang được thực hiện. Nếu có nhiều hơn số lượng lớn nhất các sessions cùng được thực hiện thì các sessions mới này sẽ được đưa vào hàng đợi để chờ thực hiện sau. Tuy nhiên ta cũng có thể đưa ra một khoảng thời gian trễ (timeout) để cho phép thực hiện hay huỷ việc thực hiện các sessions mới bổ sung này.
- Tự động chuyển users từ một nhóm này sang một nhóm khác tuỳ theo sự điều chỉnh của quản trị viên (administrator). Nếu một session được tạo bởi member thuộc một nhóm users nào đó thực hiện trong khoảng thời gian dài hơn thời gian cho phép thì

session đó có thể được tự động chuyển sang một nhóm khác với những yêu cầu tài nguyên khác.

- Có khả năng ngăn chặn thực hiện các thao tác mà được dự kiến là sẽ phải chạy trong một thời gian dài hơn là khoảng thời gian cho phép.
- Có khả năng tạo một undo pool, là nơi chứa một số lượng nhất định vùng không gian dành cho việc khôi phục thông tin (undo).

Cơ chế xác thực và nhờ xác thực (Proxy authentication and authorization)

Oracle9*i* cho phép một server nằm ở lớp giữa (middle-tier) xác nhận hộ một client. Ta có thể thực hiện việc này bằng cách đưa vào mệnh đề `GRANT CONNECT THROUGH` trong câu lệnh `ALTER USER`. Ta cũng có thể chỉ rõ vai trò của lớp giữa (middle tier) trong việc kết nối tới client.

Application roles

Oracle cho phép gán roles cho các application users mà được kích hoạt bằng cách sử dụng PL/SQL package. Sử dụng mệnh đề `IDENTIFIED USING` package trong câu lệnh `CREATE ROLE` để thực hiện việc này.

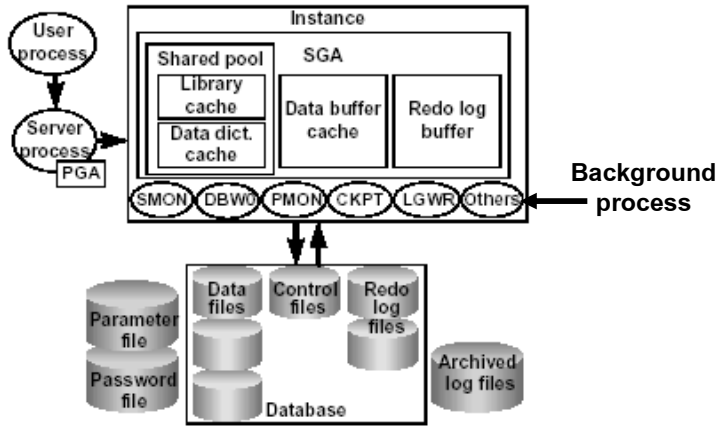
Chương 2. CÁC THÀNH PHẦN KIẾN TRÚC

2.1. KIẾN TRÚC ORACLE SERVER

Oracle server là một hệ thống quản trị cơ sở dữ liệu đối tượng-quan hệ cho phép quản lý thông tin một cách toàn diện. *Oracle server bao gồm hai thành phần chính là Oracle instance và Oracle database*.

2.1.1. Oracle Instance

Oracle instance bao gồm một cấu trúc bộ nhớ ***System Global Area (SGA)*** và các ***background processes (tiến trình nền)*** được sử dụng để quản trị cơ sở dữ liệu. Oracle instance được xác định qua tham số môi trường `ORACLE_SID` của hệ điều hành.



Hình vẽ 1. Kiến trúc Oracle Server

System Global Area - SGA

SGA là vùng bộ nhớ chia sẻ được sử dụng để lưu trữ dữ liệu và các thông tin điều khiển của Oracle server. SGA được cấp phát (allocated) trong bộ nhớ của máy tính mà Oracle server đang hoạt động trên đó. Các User kết nối tới Oracle sẽ chia sẻ các dữ liệu có trong SGA, việc mở rộng không gian bộ nhớ cho SGA sẽ làm nâng cao hiệu suất của hệ thống, lưu trữ được nhiều dữ liệu trong hệ thống hơn đồng thời giảm thiểu các thao tác truy xuất đĩa (disk I/O).

SGA bao gồm một vài cấu trúc bộ nhớ chính:

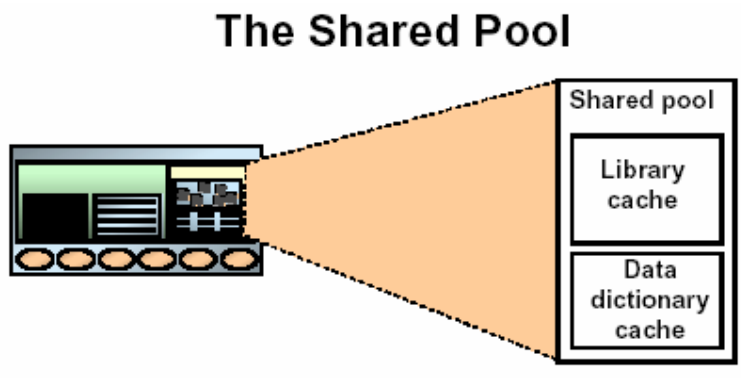
- Shared pool: Là một phần của SGA lưu các cấu trúc bộ nhớ chia sẻ.
- Database buffer cache: Lưu trữ các dữ liệu được sử dụng gần nhất.
- Redo log buffer: Được sử dụng cho việc dò tìm lại các thay đổi trong cơ sở dữ liệu và được thực hiện bởi các background process.

Để chi tiết hơn, ta sẽ xem xét cụ thể từng thành phần.

Share Pool

Shared pool là một phần trong SGA và được sử dụng khi thực hiện phân tích câu lệnh (parse phase). Kích thước của Shared pool được xác định bởi tham số `SHARED_POOL_SIZE` có trong parameter file (file tham số).

Các thành phần của Shared pool gồm có: Library cache và Data dictionary cache.



Hình vẽ 2. Cấu trúc Share Pool

Library Cache

Library cache lưu trữ thông tin về các câu lệnh SQL được sử dụng gần nhất bao gồm:

- Nội dung của câu lệnh dạng text (văn bản).
- Parse tree (cây phân tích) được xây dựng tuỳ thuộc vào câu lệnh.
- Execution plan (sơ đồ thực hiện lệnh) gồm các bước thực hiện và tối ưu lệnh.

Do các thông tin trên đã được lưu trữ trong Library cache nên khi thực hiện lại một câu lệnh truy vấn, trước khi thực hiện câu lệnh, Server process sẽ lấy lại các thông tin đã được phân tích mà không phải phân tích lại câu lệnh. Do vậy, Library cache có thể giúp nâng cao hiệu suất thực hiện lệnh.

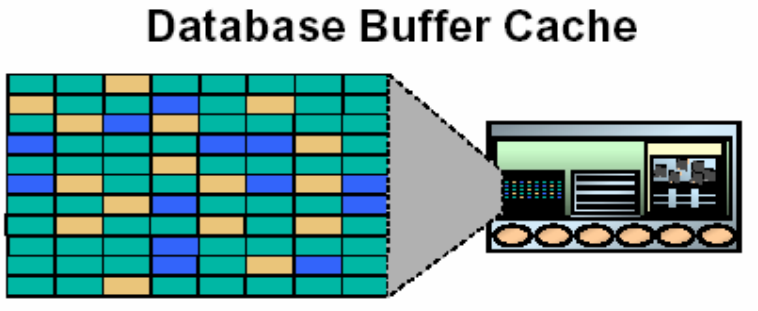
Data Dictionary Cache

Data dictionary cache là một thành phần của Shared pool lưu trữ thông tin của dictionary cache được sử dụng gần nhất như các định nghĩa các bảng, định nghĩa các cột, usernames, passwords, và các privileges (quyền).

Trong giai đoạn phân tích lệnh (parse phase), Server process sẽ tìm các thông tin trong dictionary cache để xác định các đối tượng trong câu lệnh SQL và để xác định các mức quyền tương ứng. Trong trường hợp cần thiết, Server process có thể khởi tạo và nạp các thông tin từ các file dữ liệu.

Data buffer cache

Khi thực hiện một truy vấn, Server process sẽ tìm các blocks cần thiết trong database buffer cache. Nếu không tìm thấy block trong database buffer cache, Server process mới đọc các block từ data file và tạo luôn một bản sao của block đó vào trong vùng nhớ đệm (buffer cache). Như vậy, với các lần truy xuất tới block đó sau này sẽ không cần thiết phải truy xuất vào datafile nữa.



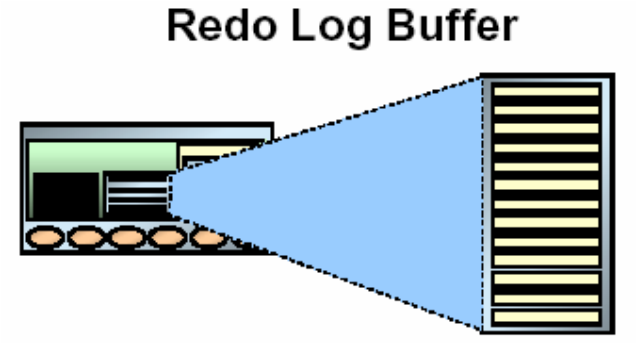
Hình vẽ 3. Database buffer cache

Database buffer cache là vùng nhớ trong SGA sử dụng để lưu trữ các block dữ liệu được sử dụng gần nhất. Tương tự như kích thước của blocks dữ liệu được xác định bởi tham số `DB_BLOCK_SIZE`, kích thước của vùng đệm trong buffer cache cũng được xác định bởi tham số `DB_BLOCK_BUFFERS`.

Oracle server sử dụng giải thuật least recently used (LRU) algorithm để làm tươi lại vùng nhớ. Theo đó, khi nạp mới một block vào bộ đệm, trong trường hợp bộ đệm đã đầy, Oracle server sẽ loại bớt block ít được sử dụng nhất ra khỏi bộ đệm để nạp block mới vào bộ đệm.

Redo log buffer

Server process ghi lại các thay đổi của một instance vào redo log buffer, đây cũng là một phần bộ nhớ SGA.



Hình vẽ 4. Redo log buffer

Có một số đặc điểm cần quan tâm của Redo log buffer:

- Kích thước được xác định bởi tham số `LOG_BUFFER`.
- Lưu trữ các redo records (bản ghi hồi phục) mỗi khi có thay đổi dữ liệu.
- Redo log buffer được sử dụng một cách thường xuyên và các thay đổi bởi một transaction có thể nằm đan xen với các thay đổi của các transactions khác.
- Bộ đệm được tổ chức theo kiểu circular buffer (bộ đệm nối vòng) tức là dữ liệu thay đổi sẽ tiếp tục được nạp lên đầu sau khi vùng đệm đã được sử dụng hết.

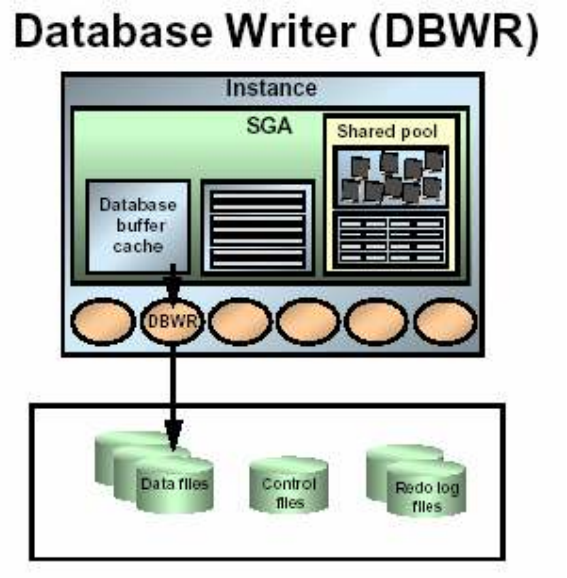
Background process

Background process (các tiến trình nền) thực hiện các chức năng thay cho lời gọi tiến trình xử lý tương ứng. Nó điều khiển vào ra, cung cấp các cơ chế xử lý song song nâng cao hiệu quả và độ tin cậy. Tùy theo từng cấu hình mà Oracle instance có các Background process như:

- Database Writer (`DBW0`): Ghi lại các thay đổi trong data buffer cache ra các file dữ liệu.
- Log Writer (`LGWR`): Ghi lại các thay đổi được đăng ký trong redo log buffer vào các redo log files.
- System Monitor (`SMON`): Kiểm tra sự nhất quán trong database.
- Process Monitor (`PMON`): Dọn dẹp lại tài nguyên khi các tiến trình của Oracle gặp lỗi.
- Checkpoint Process (`CKPT`): Cập nhật lại trạng thái của thông tin trong file điều khiển và file dữ liệu mỗi khi có thay đổi trong buffer cache.

Database Writer (DBW0)

Server process ghi lại các dữ liệu thay đổi để rollback và dữ liệu của các block trong buffer cache. Database writer (DBWR) ghi các thông tin được đánh dấu thay đổi từ database buffer cache lên các data files nhằm đảm bảo luôn có khoảng trống bộ đệm cần thiết cho việc sử dụng.



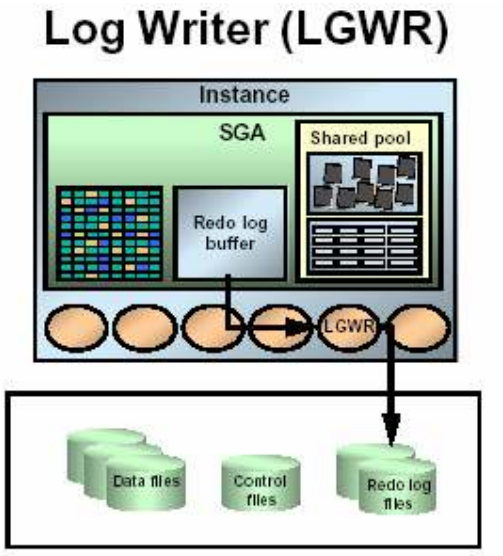
Hình vẽ 5. Database Writer (DBWR)

Với việc sử dụng này, hiệu suất sử dụng database sẽ được cải thiện do Server processes chỉ tạo các thay đổi trên buffer cache, DBWR ghi dữ liệu vào các data file cho tới khi:

- Số lượng buffers đánh bị dấu đạt tới giá trị ngưỡng.
- Tiến trình duyệt tất cả buffer mà vẫn không tìm thấy dữ liệu tương ứng.
- Quá thời gian quy định.

Log Writer

Log Writer (LGWR) là một trong các background process có trách nhiệm quản lý redo log buffer để ghi lại các thông tin trong Redo log buffer vào Redo log file. Redo log buffer là bộ đệm dữ liệu được tổ chức theo kiểu nối vòng.



Hình vẽ 6. Log Writer (LGWT)

LGWR ghi lại dữ liệu một cách tuần tự vào redo log file theo các tình huống sau:

- Khi redo log buffer đầy
- Khi xảy ra timeout (thông thường là 3 giây)
- Trước khi DBWR ghi lại các blocks bị thay đổi trong data buffer cache vào các data files.
- Khi commit một transaction.

System Monitor (SMON)

Tiến trình*system monitor (*SMON*)* thực hiện phục hồi các sự cố (crash recovery) ngay tại thời điểm instance được khởi động (startup), nếu cần thiết. SMON cũng có trách nhiệm dọn dẹp các temporary segments không còn được sử dụng nữa trong dictionary-managed tablespaces. SMON khôi phục lại các transactions bị chết mỗi khi xảy ra sự cố. SMON đều đặn thực hiện kiểm tra và khắc phục các sự cố khi cần.

Trong môi trường Oracle Parallel Server, SMON process của một instance có thể thực hiện khôi phục instance trong trường hợp instance hay CPU của máy tính đó gặp sự cố.

Process Monitor (PMON)

Tiến trình *process monitor (*PMON*)* thực hiện tiến trình phục hồi mỗi khi có một user process gặp lỗi. PMON có trách nhiệm dọn dẹp database buffer cache và giải phóng tài nguyên mà user process đó sử dụng. Ví dụ, nó thiết lập lại (reset) trạng thái của các bảng đang thực hiện trong transaction, giải phóng các locks trên bảng này, và huỷ bỏ process ID của nó ra khỏi danh sách các active processes.

PMON kiểm tra trạng thái của nơi gửi (dispatcher) và các server processes, khởi động lại (restarts) mỗi khi xảy ra sự cố. PMON cũng còn thực hiện việc đăng ký các thông tin về instance và dispatcher processes với network listener.

Tương tự như SMON, PMON được gọi đến mỗi khi xảy ra sự cố trong hệ thống.

Checkpoint Process (CKPT)

Cập nhật lại trạng thái của thông tin trong file điều khiển và file dữ liệu mỗi khi có thay đổi trong buffer cache. Xảy ra checkpoints khi:

- Tất cả các dữ liệu trong database buffers đã bị thay đổi tính cho đến thời điểm checkpointed sẽ được background process $DBWR_n$ ghi lên data files.
- Background process CKPT cập nhật phần headers của các data files và các control files.

Checkpoints có thể xảy ra đối với tất cả các data files trong database hoặc cũng có thể xảy ra với một data files cụ thể.

Checkpoint xảy ra theo các tình huống sau:

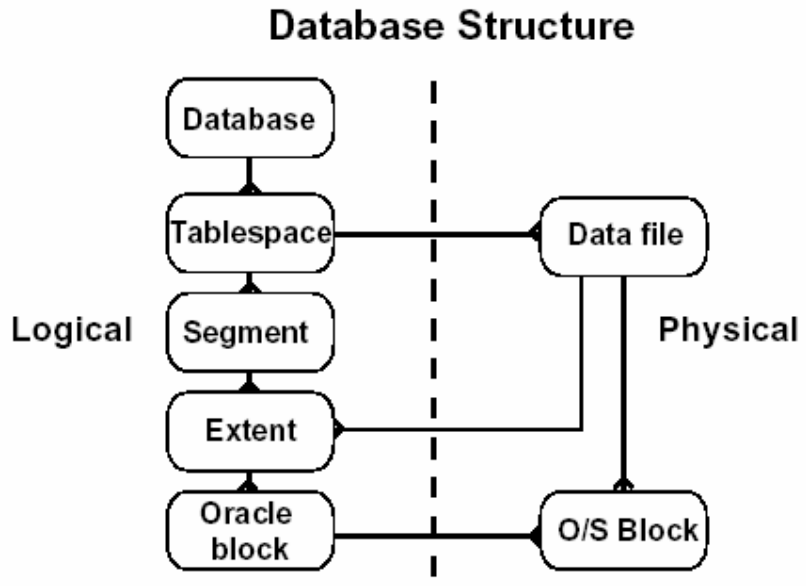
- Mỗi khi có log switch
- Khi một shut down một database với các chế độ trừ chế độ abort
- Xảy ra theo như thời gian quy định trong các tham số khởi tạo `LOG_CHECKPOINT_INTERVAL` và `LOG_CHECKPOINT_TIMEOUT`
- Khi có yêu cầu trực tiếp của quản trị viên

Thông tin về checkpoint được lưu trữ trong Alert file trong trường hợp các tham số khởi tạo `LOG_CHECKPOINTS_TO_ALERT` được đặt là `TRUE`. Và ngược lại với giá trị `FALSE`.

2.1.2. Oracle database

Oracle *database* là tập hợp các dữ liệu được xem như một đơn vị thành phần (Unit). Database có nhiệm vụ lưu trữ và trả về các thông tin liên quan. Database được xem xét dưới hai góc độ *cấu trúc logic* và *cấu trúc vật lý* . Tuy vậy, hai cấu trúc dữ liệu này vẫn tồn tại tách biệt nhau, việc quản lý dữ liệu theo cấu trúc lưu trữ vật lý không gây ảnh hưởng tới cấu trúc logic

Oracle database được xác định bởi tên một tên duy nhất và được quy định trong tham số DB_NAME của parameter file.



Hình vẽ 7. Cấu trúc database

Cấu trúc vật lý database

Cấu trúc vật lý bao gồm tập hợp các control file, online redo log file và các datafile:

Datafiles

Mỗi một Oracle database đều có thể có một hay nhiều *datafiles*. Các database datafiles chứa toàn bộ dữ liệu trong database. Các dữ liệu thuộc cấu trúc logic của database như tables hay indexes đều được lưu trữ dưới dạng vật lý trong các datafiles của database.

Một số tính chất của datafiles:

- Mỗi datafile chỉ có thể được sử dụng trong một database.
- Bên cạnh đó, datafiles cũng còn có một số tính chất cho phép tự động mở rộng kích thước mỗi khi database hết chỗ lưu trữ dữ liệu.
- Một hay nhiều datafiles tạo nên một đơn vị lưu trữ logic của database gọi là tablespace.
- Một datafile chỉ thuộc về một tablespace.

Dữ liệu trong một datafile có thể đọc ra và lưu vào vùng nhớ bộ đệm của Oracle. Ví dụ: khi một user muốn truy cập dữ liệu trong một table thuộc database. Trong trường hợp thông tin yêu cầu không có trong cache memory hiện thời, nó sẽ được đọc trực tiếp từ các datafiles ra và lưu trữ vào trong bộ nhớ.

Tuy nhiên, việc bổ sung hay thêm mới dữ liệu vào database không nhất thiết phải ghi ngay vào các datafile. Các dữ liệu có thể tạm thời ghi vào bộ nhớ để giảm thiểu việc truy xuất tới bộ nhớ ngoài (ổ đĩa) làm tăng hiệu năng sử dụng hệ thống. Công việc ghi dữ liệu này được thực hiện bởi DBW*n* background process.

Redo Log Files

Mỗi Oracle database đều có một tập hợp từ 02 *redo log files* trở lên. Các redo log files trong database thường được gọi là database's *redo log*. Một redo log được tạo thành từ nhiều redo entries (gọi là các *redo records*).

Chức năng chính của redo log là ghi lại tất cả các thay đổi đối với dữ liệu trong database. Redo log files được sử dụng để bảo vệ database khỏi những hỏng hóc do sự cố. Oracle cho phép sử dụng cùng một lúc nhiều redo log gọi là *multiplexed redo log* để cùng lưu trữ các bản sao của redo log trên các ổ đĩa khác nhau.

Các thông tin trong redo log file chỉ được sử dụng để khôi phục lại database trong trường hợp hệ thống gặp sự cố và không cho phép viết trực tiếp dữ liệu trong database lên các datafiles trong database. Ví dụ: khi có sự cố xảy ra như mất điện bất chợt chẳng hạn, các dữ liệu trong bộ nhớ không thể ghi trực tiếp lên các datafiles và gây ra hiện tượng mất dữ liệu. Tuy nhiên, tất cả các dữ liệu bị mất này đều có thể khôi phục lại ngay khi database được mở trở lại. Việc này có thể thực hiện được thông qua việc sử dụng ngay chính các thông tin mới nhất có trong các redo log files thuộc datafiles. Oracle sẽ khôi phục lại các database cho đến thời điểm trước khi xảy ra sự cố.

Công việc khôi phục dữ liệu từ các redo log được gọi là *rolling forward*.

Control Files

Mỗi Oracle database đều có ít nhất một *control file*. Control file chứa các mục thông tin quy định cấu trúc vật lý của database như:

- Tên của database.
- Tên và nơi lưu trữ các datafiles hay redo log files.

- Time stamp (mốc thời gian) tạo lập database, ...

Mỗi khi nào một instance của Oracle database được mở, control file của nó sẽ được sử dụng để xác định data files và các redo log files đi kèm. Khi các thành phần vật lý cả database bị thay đổi (ví dụ như, tạo mới datafile hay redo log file), Control file sẽ được tự động thay đổi tương ứng bởi Oracle.

Control file cũng được sử dụng đến khi thực hiện khôi phục lại dữ liệu.

Cấu trúc logic databse

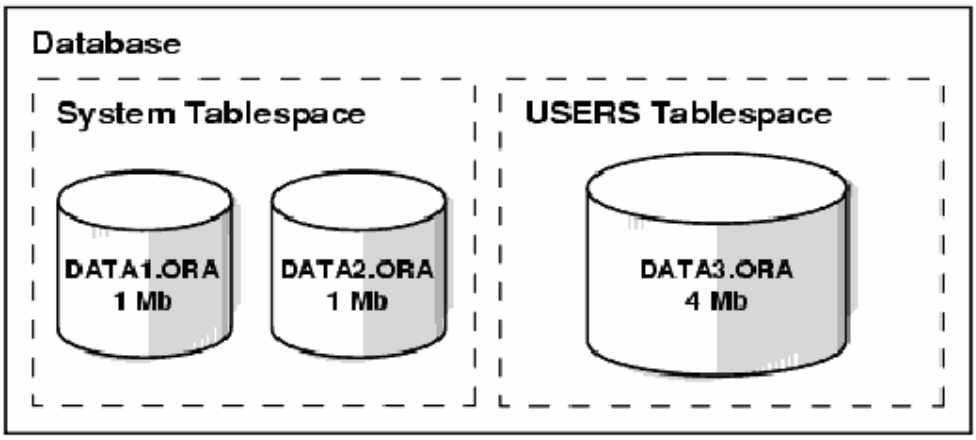
Cấu trúc logic của Oracle database bao gồm các đối tượng tablespaces, schema objects, data blocks, extents, và segments.

Tablespaces

Một database có thể được phân chia về mặt logic thành các đơn vị gọi là các *tablespaces*, Tablespaces thường bao gồm một nhóm các thành phần có quan hệ logic với nhau.

Databases, Tablespaces, và Datafiles

Mối quan hệ giữa các databases, tablespaces, và datafiles có thể được minh hoạ bởi hình vẽ sau:



Hình vẽ 8. Quan hệ giữa database, tablespace và datafile

Có một số điểm ta cần quan tâm:

- Mỗi database có thể phân chia về mặt logic thành một hay nhiều tablespace.
- Mỗi tablespace có thể được tạo nên, về mặt vật lý, bởi một hoặc nhiều datafiles.
- Kích thước của một tablespace bằng tổng kích thước của các datafiles của nó. Ví dụ: trong hình vẽ ở trên SYSTEM tablespace có kích thước là 2 MB còn USERS tablespace có kích thước là 4 MB.
- Kích thước của database cũng có thể xác định được bằng tổng kích thước của các tablespaces của nó. Ví dụ: trong hình vẽ trên thì kích thước của database là 6 MB.

Schema và Schema Objects

S*chema* là tập hợp các đối tượng (objects) có trong database. *Schema objects* là các cấu trúc logic cho phép tham chiếu trực tiếp tới dữ liệu trong database. Schema objects bao gồm các cấu trúc như tables, views, sequences, stored procedures, synonyms, indexes, clusters, và database links.

Data Blocks, Extents, and Segments

Oracle điều khiển không gian lưu trữ trên đĩa cứng theo các cấu trúc logic bao gồm các data blocks, extents, và segments.

Oracle Data Blocks

Là mức phân cấp logic thấp nhất, các dữ liệu của Oracle database được lưu trữ trong các *data blocks*. Một data block tương ứng với một số lượng nhất định các bytes vật lý của database trong không gian đĩa cứng. Kích thước của một data block được chỉ ra cho mỗi Oracle database ngay khi database được tạo lập. Database sử dụng, cấp phát và giải phóng vùng không gian lưu trữ thông qua các Oracle data blocks.

Extents

Là mức phân chia cao hơn về mặt logic các vùng không gian trong database. Một *extent* bao gồm một số data blocks liên tiếp nhau, cùng được lưu trữ tại một thiết bị lưu giữ. Extent được sử dụng để lưu trữ các thông tin có cùng kiểu.

Segments

Là mức phân chia cao hơn nữa về mặt logic các vùng không gian trong database. Một *segment* là một tập hợp các extents được cấp phát cho một cấu trúc logic . Segment có thể được phân chia theo nhiều loại khác nhau:

Data segment	Mỗi một non-clustered table có một data segment. Các dữ liệu trong một table được lưu trữ trong các extents thuộc data segment đó. Với một partitioned table thì mỗi each partition lại tương ứng với một data segment. Mỗi Cluster tương ứng với một data segment. Dữ liệu của tất cả các table trong cluster đó đều được lưu trữ trong data segment thuộc Cluster đó.
index segment	Mỗi một index đều có một index segment lưu trữ các dữ liệu của nó. Trong partitioned index thì mỗi partition cũng lại tương ứng với một index segment.
rollback segment	Một hoặc nhiều rollback segments của database được tạo lập bởi người quản trị database để lưu trữ các dữ liệu trung gian phục vụ cho việc khôi phục dữ liệu. Các thông tin trong Rollback segment được sử dụng để: ▪ Tạo sự đồng nhất các thông tin đọc được từ database ▪ Sử dụng trong quá trình khôi phục dữ liệu ▪ Phục hồi lại các giao dịch chưa commit đối với mỗi user
temporary segment	Temporary segments được tự động tạo bởi Oracle mỗi khi một câu lệnh SQL statement cần đến một vùng nhớ trung gian để thực hiện các công việc của mình như sắp xếp dữ liệu. Khi kết thúc câu lệnh đó, các extent thuộc temporary segment sẽ lại được hoàn trả cho hệ thống.

Oracle thực hiện cấp phát vùng không gian lưu trữ một cách linh hoạt mỗi khi các extents cấp phát đã sử dụng hết.

Các cấu trúc vật lý khác

Ngoài ra, Oracle Server còn sử dụng các file khác để lưu trữ thông tin. Các file đó bao gồm:

- Parameter file: Parameter file chỉ ra các tham số được sử dụng trong database. Người quản trị database có thể sửa đổi một vài thông tin có trong file này. Các tham số trong parameter file được viết ở dạng văn bản.
- Password file: Xác định quyền của từng user trong database. Cho phép người sử dụng khởi động và tắt một Oracle instance.
- Archived redo log files: Là bản off line của các redo log files chứa các thông tin cần thiết để phục hồi dữ liệu.

2.1.3. Quản trị cơ sở dữ liệu Oracle

Quản trị cơ sở dữ liệu là công việc bảo trì và vận hành Oracle server để nó có thể tiếp nhận và xử lý được tất cả các yêu cầu (requests) từ phía Client. Để làm được điều này, người quản trị viên cơ sở dữ liệu cần phải hiểu được kiến trúc của Oracle database.

2.1.4. Thiết lập các tham số khởi tạo ảnh hưởng tới kích cỡ bộ nhớ SGA

Tham số khởi tạo ảnh hưởng tới kích thước bộ nhớ cấp phát cho vùng System Global Area. Ngoại trừ tham số `SGA_MAX_SIZE`, còn lại các tham số khác đều là tham số động tức là có thể thay đổi giá trị của chúng ngay trong lúc database đang chạy thông qua câu lệnh `ALTER SYSTEM`. Kích thước của SGA cũng có thể thay đổi được trong quá trình chạy database.

Thiết lập tham số cho Buffer Cache

Tham số khởi tạo buffer cache quy định kích thước của buffer cache là một phần của SGA. .

Ta sử dụng các tham số `DB_CACHE_SIZE` và một trong những tham số `DB_nK_CACHE_SIZE` để cho phép sử dụng chế độ multiple block sizes đối với database. Oracle sẽ tự động gán các giá trị mặc định cho tham số the `DB_CACHE_SIZE`, còn tham số `DB_nK_CACHE_SIZE` sẽ được gán mặc định bằng 0.

Kích thước của buffer cache sẽ có ảnh hưởng nhiều tới hiệu suất thực hiện của hệ thống. Kích thước càng lớn thì càng giảm bớt việc đọc và ghi đĩa. Tuy nhiên, kích thước của cache lớn sẽ tốn nhiều bộ nhớ và sẽ có nhiều tốn kém trong việc thực hiện paging (phân trang) hay swapping (trao đổi) bộ nhớ.

Tham số DB_CACHE_SIZE

Tham số khởi tạo `DB_CACHE_SIZE` được sử dụng thay thế cho tham số `DB_BLOCK_BUFFERS` của các phiên bản Oracle trước kia. Tham số `DB_CACHE_SIZE` quy định kích thước của block buffers chuẩn. Kích thước của một block chuẩn lại được quy định trong tham số `DB_BLOCK_SIZE`.

Tuy vậy, tham số `DB_BLOCK_BUFFERS` vẫn được sử dụng để tương thích với các phiên bản trước, tuy nhiên giá trị của nó không được sử dụng cho các tham số động.

Tham số DB_*n*K_CACHE_SIZE

Chỉ ra kích cỡ là bội số nguyên lần kích thước của block buffers. Nó được chỉ ra bởi các tham số:

- `DB_2K_CACHE_SIZE`

- DB_4K_CACHE_SIZE
- DB_8K_CACHE_SIZE
- DB_16K_CACHE_SIZE
- DB_32K_CACHE_SIZE.

Mỗi tham số chỉ ra kích cỡ của buffer cache tương ứng với kích cỡ của block.

Ví dụ:

```
DB_BLOCK_SIZE=4096
DB_CACHE_SIZE=12M
DB_2K_CACHE_SIZE=8M
DB_8K_CACHE_SIZE=4M
```

Ở ví dụ này, các tham số chỉ ra kích thước block chuẩn của database là 4K. Kích thước cache tương ứng với kích thước block chuẩn là 12M. Các kích thước mở rộng của cache là 2K và 8K sẽ được đặt lại với giá trị tương ứng là 8M và 4M.

Điều chỉnh kích cỡ của Shared Pool

Tham số SHARED_POOL_SIZE trong phiên bản Oracle 9i là tham số động, tức là có thể thay đổi được giá trị của nó (điều này không thể thực hiện được trong các phiên bản trước). Nó cho phép ta thay đổi kích thước của shared pool là một trong các thành phần của SGA. Theo mặc định Oracle cũng chọn một giá trị mặc định phù hợp cho tham số này.

Điều chỉnh kích cỡ của Large Pool

Tương tự như SHARED_POOL_SIZE, tham số LARGE_POOL_SIZE cũng là một tham số động, nó cho phép ta điều chỉnh kích cỡ của large pool, đây cũng là một thành phần trong SGA. .

Giới hạn kích cỡ của SGA

Tham số SGA_MAX_SIZE quy định kích cỡ lớn nhất của System Global Area . Ta cũng có thể thay đổi kích cỡ của buffer caches, shared pool và large pool, tuy nhiên việc thay đổi này nên là mở rộng giá trị kích thước cho các thành phần của SGA. Giá trị mở rộng thêm này cũng không nên đặt tới ngưỡng của SGA_MAX_SIZE.

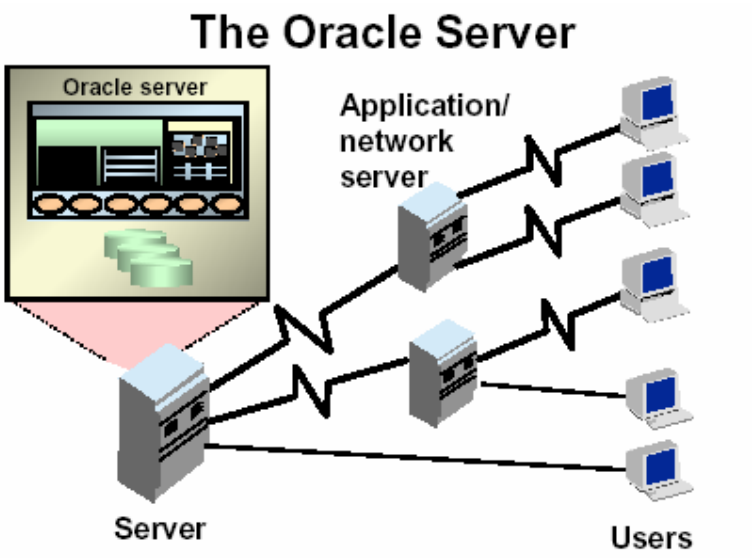
Trong trường hợp ta không chỉ rõ giá trị của SGA_MAX_SIZE thì Oracle sẽ tự động gán giá trị này bằng tổng số kích cỡ của các thành phần của SGA lúc ban đầu.

2.2.KẾT NỐI TỚI ORACLE SERVER

2.2.1. Mô hình kết nối

Các Client có thể kết nối tới Oracle Server thông qua 03 cách sau:

- Kết nối trực tiếp: kết nối mà Client nằm trên cùng một máy chủ Oracle server.
- Kết nối hai lớp (two-tiered) client-server: Client nằm trên một máy tính khác và kết nối trực tiếp tới máy chủ Oracle Server.
- Kết nối ba lớp (three-tiered): Client nằm trên máy tính khác với máy chủ Oracle Server, nó giao tiếp với một ứng dụng hay một máy chủ mạng (network server) và điều khiển ứng dụng hay máy chủ này kết nối tới Oracle server.



Hình vẽ 9. Kết nối tới Oracle server

2.2.2. Một số khái niệm cơ bản liên quan đến kết nối

Connection (liên kết)

Liên kết là đường liên lạc giữa một user process và một Oracle server. Trong trường hợp user sử dụng các tool hoặc các ứng dụng ngay trên cùng một máy với Oracle server, đường liên lạc sẽ được tạo lập ngay trên máy đó. Trong trường hợp user nằm trên một máy khác thì liên kết sẽ sử dụng đường mạng để kết nối tới Oracle server.

Session (phiên)

Một phiên tương ứng với một liên kết cụ thể của một user tới một Oracle server. Phiên bắt đầu khi user kết nối tới Oracle Server đã được kiểm tra hợp lệ và kết thúc khi user thực hiện log out khỏi Oracle Server hoặc user kết thúc một cách bất thường. Một user cùng một lúc có thể có nhiều phiên làm việc để kết nối tới Oracle Server thông qua các ứng dụng hay các tool khác nhau. Ví du: User có thể đồng thời có các phiên làm việc giữa SQL*Plus, Developer/2000 Form,... tới Oracle Server.

Lưu ý: Phiên chỉ tạo lập được khi Oracle Server đã sẵn sàng cho việc kết nối của các client.

2.2.3. Kết nối tới database

Các bước thực hiện kết nối

Để kết nối tới database trước tiên, cần phải tạo liên kết tới Oracle Server. Liên kết tới Oracle Server được tạo theo các bước sau:

- User sử dụng công cụ SQL*Plus hay sử dụng các công cụ khác của Oracle như Developer/2000 Forms để khởi tạo tiến trình. Trong mô hình Client-Server, các công cụ hay ứng dụng này được chạy trên máy Client.
- User thực hiện log in vào Oracle server với việc khai báo username, password và tên liên kết tới database. Các ứng dụng tools sẽ tạo một tiến trình để kết nối tới Oracle server qua các tham số này. Tiến trình này được gọi là tiến trình phục vụ. Tiến trình phục vụ sẽ giao tiếp với Oracle server thay cho tiến trình của user chạy trên máy Client.

Ví dụ thực hiện kết nối tới database

Để hiểu rõ hơn về các bước thực hiện kết nối, ta hãy xem xét một ví dụ mô tả việc kết nối tới Oracle database thực hiện bởi một user tại một máy tính khác có kết nối tới máy tính mà Oracle server đang chạy trên đó. Việc kết nối được thực hiện thông qua đường mạng bằng cách sử dụng dịch vụ Oracle Net8.

1. Tại máy chủ, cần đảm bảo Oracle server đang chạy và sẵn sàng đón nhận các tín hiệu từ phía Client. Máy chủ này được gọi là *host* hay *database server*.
2. Tại một máy trạm có chạy các ứng dụng (gọi là *local machine* hay *client workstation*) sẽ thực hiện các user process để kết nối tới database. Client application thực hiện thiết lập một kết nối tới server thông qua Net8 driver.
3. Máy chủ server trên đó có các Net8 driver. Server sẽ thực hiện việc nghe và dò tìm tất cả các yêu cầu gửi đến từ phía client và sau đó sẽ tạo một server process tương ứng với user process.
4. Khi user thực hiện một câu lệnh SQL hay commit một transaction. Ví dụ như user dữ liệu trên một dòng trong một table.
5. Server process sẽ nhận về câu lệnh gửi tới từ Client, kiểm tra và phân tích câu lệnh, việc này được thực hiện trong shared pool. Tiếp theo đó, Server process sẽ kiểm tra quyền truy nhập dữ liệu của user.
6. Server process trả về các giá trị dữ liệu yêu cầu từ các dữ liệu có trong datafile hay trong system global area.
7. Server process thay đổi các dữ liệu có trong system global area. DBW*n* process ghi lại các blocks đã thay đổi ra ổ đĩa. `LGWR` process sẽ ghi lại ngay lập tức các bản ghi thay đổi vào online redo log file ngay khi transaction được commit.
8. Trong trường hợp transaction thực hiện thành công, server process sẽ gửi thông báo hoàn tất qua đường mạng tới Client. Ngược lại, sẽ có một error message gửi tới Client.

Chương 3. CÁC CÔNG CỤ QUẢN TRỊ ORACLE

3.1.CÁC CÔNG CỤ QUẢN TRỊ CƠ SỞ DỮ LIỆU ORACLE

Oracle hiện tại đã cung cấp rất nhiều công cụ cho phép quản trị cơ sở dữ liệu. Có thể kể ra sau đây một số công cụ cơ bản sau:

Công cụ	Diễn giải
Server Manager Line Mode	Công cụ quản trị cơ sở dữ liệu Oracle theo chế độ dòng lệnh như start (khởi động), shutdown (dừng) database,...
SQL*Plus Line Mode	Đây là một tiện ích sử dụng cho việc công quản trị như starting up, shutting down, hoặc phục hồi database.
Oracle Enterprise Manager	Là công cụ có giao diện đồ họa để thực hiện việc quản trị, điều khiển và thay đổi một hoặc nhiều database.
SQL*Loader	Tiện ích sử dụng để nạp các file bên ngoài vào trong các bảng của Oracle .
Công cụ Export và Import	Tiện ích sử dụng để exporting hoặc importing dữ liệu theo khuôn dạng của Oracle.
Password File	Tiện ích sử dụng để tạo file mật khẩu trong database.

Để thuận tiện, tài liệu sẽ trình bày hai công cụ thường dùng nhất để quản trị cơ sở dữ liệu là:

- Server Manager Line Mode
- Oracle Enterprise Manager

3.2.SERVER MANAGER LINE MODE

3.2.1. Truy nhập Server Manager Line Mode

User (người sử dụng) có thể vào Server Manager Line Mode theo hai cách:

```
C:\svrmgrl
```

Cách này chỉ vào Server Manager Line Mode mà chưa thực hiện kết nối cụ thể tới database

Lưu ý: trong các phiên bản cũ, ta gõ `svrmgrl30` thay vì `svrmgrl`

Hoặc:

```
C:\svgrmrl command=”connect internal/admin”
C:\svgrmrl command=@c:\example.sql
```

Vào Server Manager Line Mode đồng thời thực hiện lệnh luôn.

Kí tự sử dụng trong Server Manager Line Mode

Với Server Manager Line Mode, ta có thể thực hiện câu lệnh SQL hoặc đoạn lệnh PL/SQL. Các câu lệnh được kết thúc bởi ký tự chấm phẩy ' ; '

Sử dụng ký tự ' /' để kết thúc câu lệnh trong trường hợp đã bấm phím Enter để xuống dòng.

Ngoài ra, ta có thể chạy file script chứa các câu lệnh SQL và PL/SQL. Bằng cách sử dụng ký tự ' @ ' ở trước tên file script.

3.2.2. Phân nhóm các lệnh trong Server manager

Loại lệnh	Tên lệnh
Lệnh không cần kết nối tới database	EXIT REMARK SET SHOW SPOOL
Các lệnh cần đến mức quyền truy nhập	CONNECT/DISCONNECT DESCRIBE EXECUTE SHOW ERRORS SHOW PARAMETER SHOW SGA
Các lệnh cần đến mức quyền truy nhập đặc biệt	CONNECT… AS SYSDBA CONNECT… AS SYSOPER ARCHIVE LOG RECOVER DATABASE STARTUP/SHUTDOWN

3.2.3. Diễn giải các lệnh trong Server manager

Tên lệnh	Diễn giải
EXIT	Đóng SQL Worksheet, thoát khỏi Server Manager
REMARK	Thêm vào lời chú dẫn, thường hay sử dụng trong file SQL script
SET	Thiết lập hoặc thay đổi các tính chất có trong phiên làm việc hiện thời.
SHOW	Hiển thị các thiết đặt hiện thời
SPOOL	Cho phép hoặc thôi cho phép chuyển hướng kết xuất dữ liệu ra file
CONNECT/ DISCONNECT	Kết nối hoặc huỷ kết nối tới database
DESCRIBE	Xem cấu trúc của một function, package, package body, procedure, table, object, view
EXECUTE	Thực hiện một dòng lệnh PL/SQL
SHOW ERRORS	Hiển thị các lỗi phát sinh của thủ tục, hàm hay package
SHOW PARAMETER	Hiển thị gái trị hiện thời của một hay nhiều tham số đã khởi tạo
SHOW SGA	Hiển thị thông tin về SGA của Instance hiện thời
CONNECT/AS SYSDBA	Kết nối tới database với đặc quyền quản trị
ARCHIVE LOG	Khởi động và dừng việc lưu trữ tự động đối với các file online redo log files, redo log file
RECOVER DATABASE	Phục hồi lại một hay nhiều tablespaces

STARTUP/ SHUTDOWN	Khởi động hoặc tắt Oracle instance

3.3.ORACLE ENTERPRISE MANAGER

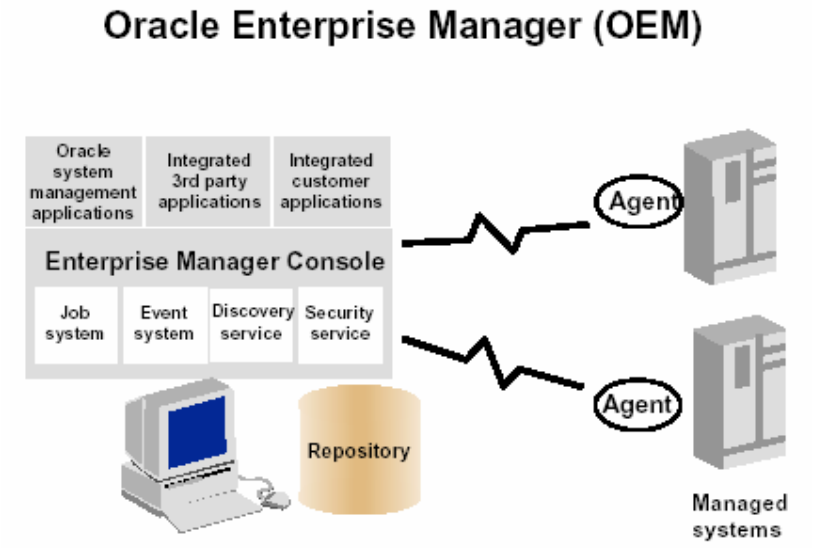
Oracle Enterprise Manager (OME) là phương tiện cho phép có được cái nhìn tổng thể về toàn bộ hệ thống. Trong đó có cây phân cấp và các hình ảnh đồ họa về các đối tượng và quan hệ giữa chúng trong hệ thống.

OME có các tiến trình Intelligent Agent processes cho phép quản lý từ xa các dịch vụ chung - common services như jobs, events,… một cách dễ dàng.

OME cũng bao gồm cả những ứng dụng quản lý chuyên biệt: DBA Management Pack, Advanced Management Packs.

Bên cạnh đó, OME còn cung cấp một lượng lớn các hàm API cho phép tích hợp với các hệ thống quản lý ứng dụng khác. Bao gồm cả các hệ thống quản lý của Oracle và không phải của Oracle.

OEM Console có trong cả Windows NT và Windows 95.



Hình vẽ 10. Oracle Enterprise Manager

3.3.1. Kiến trúc OME

Kiến trúc OME là mở rộng của kiến trúc Client/Server, nó có kiến trúc ba lớp.

- Lớp thứ nhất chứa các Java-based console và các ứng dụng tích hợp cho phép cài đặt và chạy bởi các Web browser.
- Lớp thứ hai là Oracle Management Server - OMS. Chức năng chính của OMS là xử lý và quản trị tất cả các tác vụ của hệ thống, tập trung quản lý và phân phối điều khiển giữa các clients và các nút điều khiển - managed nodes. OEM sử dụng Oracle Enterprise Manager repository để duy trì dữ liệu hệ thống, dữ liệu ứng dụng và các trạng thái của các thực thể điều khiển phân tán trong hệ thống, cho phép người dùng có thể truy cập và chia sẻ các vùng dữ liệu lưu trữ.
- Lớp thứ ba bao gồm các đích như databases, nodes và các dịch vụ quản lý khác.

3.3.2. Các dịch vụ chung

OEM có các dịch vụ cho phép quản lý các nodes trên mạng (network)

- Dịch vụ phát hiện - Discovery service: OEM tự động phát hiện (định vị) tất cả các database và các dịch vụ chạy trên các nodes, một khi các nodes được xác định. Các dịch vụ này bao gồm Web servers, listeners, machines, parallel servers, video servers, và các services khác.
- Job Scheduling System: cho phép thực hiện tự động lặp lại các tác vụ. Hệ thống cho phép tạo và quản lý các jobs, lên kế hoạch thực hiện chúng và cho phép xem, chia sẻ thông tin xác định Jobs.
- Event Management System: cho phép quản lý môi trường mạng (network environment) xử lý các trường hợp mất dịch vụ, thiếu hoặc hết vùng lưu trữ, và các vấn đề khác như sử dụng tài nguyên CPU. Mỗi khi các events được phát hiện, người quản trị có thể thông báo hoặc sửa nó.
- Bảo mật - Security: các tham số bảo mật xác định cho từng dịch vụ (services), đối tượng (objects), và từng user quản trị (administrators).
- Dịch vụ kho lưu trữ chia sẻ (Shared Repository)

OEM là một hệ thống đa người dùng - multiuser system. Mỗi quản trị viên có một account riêng để đăng nhập vào hệ thống. Tùy theo việc thiết đặt quyền hạn, mà quản trị viên có thể truy cập vào các dữ liệu lưu trong kho trung tâm, kho được chia sẻ cho tất cả các quản trị viên của OEM để thực hiện công việc quản lý.

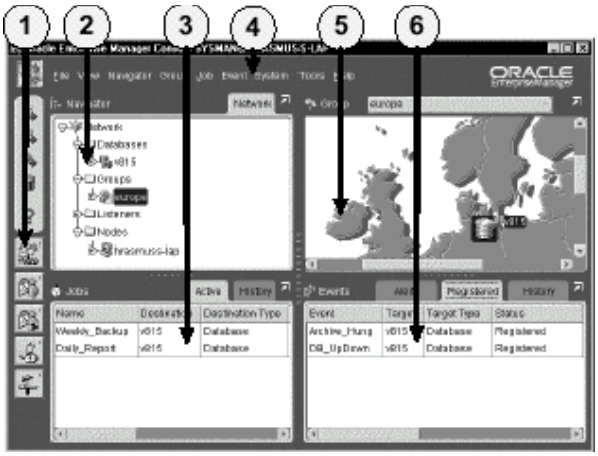
3.3.3. Oracle Configuration Assistant

Configuration Assistant là công cụ cho phép tạo các shared repositories, đặt lại cấu hình cho database và thiết đặt cấu hình cho các local console (đơn vị điều khiển cục bộ). Configuration Assistant được tự động khởi động ngay sau khi hoàn tất quá trình cài đặt của Universal Installer. Ta cũng có thể khởi động ứng dụng này bằng tay (chạy lệnh `%emrepmgr` từ dấu nhắc hệ thống).

3.3.4. Oracle Enterprise Manager Console

Bao gồm cả cây phân cấp và hình ảnh đồ họa biểu diễn các đối tượng trong hệ thống.

1. **Các nút có biểu tượng:** cho phép gọi các ứng dụng khác để cùng thực hiện việc quản trị các tác vụ (task). Việc này cũng có thể thực hiện thông qua mục chọn tương ứng trên menu.
2. **Navigator** hay **object explorer:** được tổ chức dưới dạng cây phân cấp. Nó cho phép xem các Oracle services trong mạng làm việc. Navigator cho phép quản trị viên có thể browse các Oracle services, như databases, listeners, nodes, và name servers, qua đó có thể sửa đổi các tính chất của các đối tượng; ví dụ: người dùng có thể thay đổi nội dung của bảng.
3. **Job system:** cho phép thực hiện các tác vụ từ xa liên quan tới listeners, databases. Job system dựa trên các thủ tục trong Tool Control Language (TCL) engine.



Hình vẽ 11. Oracle Enterprise Manager

4. **Menu** cho phép khởi tạo các ứng dụng quản trị khác và thực hiện nhiều tác vụ khác nhau.
5. **Map** hay **topographical view** cho phép các Oracle services có thể được gộp lại tùy theo quan hệ về không gian, chức năng, hay cả hai. Map view cho phép người sử dụng tập trung vào các đối tượng cần quản lý.
6. **Event system** điều khiển và thông báo các trạng thái của hệ thống.

3.4.CÁC CÔNG CỤ QUẢN TRỊ KHÁC

Ngoài hai công cụ chính như đã kể trên, Oracle còn hỗ trợ bộ các công cụ chuẩn khác như:

- **Instance Manager:** dùng để điều khiển database định nghĩa và khởi tạo các tham số liên quan tới các tính chất của instance.
- **Schema Manager:** dùng để tạo lập và quản lý các đối tượng như tables, indexes, và views.
- **Security Manager:** dùng để quản lý các users và phân quyền cho các users này
- **Storage Manager:** dùng để tổ chức các database files và quản lý các rollback segments.
- **SQL Worksheet:** giao tiếp theo kiểu dòng lênh, nó cho phép thực hiện các câu lệnh SQL và PL/SQL cũng như là các câu lệnh của Server Manager
- **Backup Manager:** dùng để sao lưu, phục hồi và bảo trì databases, quản lý các redo log files.
- **Data Manager:** dùng để nạp và tổ chức lại dữ liệu trong databases.

Ngoài các các công cụ kể trên, Oracle còn hỗ trợ các công cụ làm tăng cường hiệu suất làm việc của DATABASE.

- **Performance Manager:** biểu diễn hiệu suất làm việc của database dưới dạng biểu đồ đồ hoạ.
- **Top-Session Manager:** hiển thị thông tin chi tiết về các session của 10 session có sử dụng tài nguyên hệ thống, sắp xếp theo thứ tự giảm dần. Công cụ này còn cho phép kill session.
- **Loock Manager:** cho biết các thông tin liên quan đến việc khoá (lock) các đối tượng trong database. Các thông tin được biểu diễn dưới dạng đồ hoạ.
- **Tablespaces Manager:** công cụ giúp cho dễ dàng quản lý các tablespace có trong database.

Chương 4. TẠO DATABASE

4.1.CÁC BƯỚC TẠO DATABASE

Oracle hiện đã hỗ trợ một công cụ cho phép tạo database trên hệ điều hành Windows một cách trực quan. Đó là công cụ Oracle Database Assistant. Tuy nhiên, trong một số trường hợp công cụ này tỏ ra không được thuận tiện lắm.

Bên cạnh việc sử dụng công cụ cung cấp sẵn của Oracle để tạo database, Oracle còn cho phép user có thể tạo database mà không sử dụng các công cụ của Oracle. Phương pháp này gọi là tạo database bằng tay – manually.

Việc tạo database được tiến hành theo các bước:

1. Quyết định chọn lựa tên instance và tên database duy nhất. Chọn character set – tập ký tự sử dụng trong database.
2. Thiết lập các biến hệ thống.
3. Chuẩn bị file tham số, tạo file mật khẩu (nên có thao tác này).
4. Chuẩn bị instance phục vụ quản trị
5. Tạo database.
6. Chạy scripts để tạo các dictionary cho database.

4.2.CHUẨN BỊ MÔI TRƯỜNG

4.2.1. Chuẩn bị hệ điều hành

Để tạo database, quản trị viên trước tiên phải có thể truy nhập vào hệ điều hành với đầy đủ quyền.

Trước khi tạo database, cần tính toán dung lượng bộ nhớ cho database căn cứ vào cấu hình của Server và đảm bảo có đủ bộ nhớ để thực hiện các tiến trình của Oracle một cách hiệu quả.

Tính toán lượng đĩa trống cần thiết cho việc lưu trữ các data files, các control files, các redo log file và các files khác...

4.2.2. Lên kế hoạch bố trí các file thông tin

Để bảo vệ an toàn cho database, ta cần có kế hoạch bố trí các file thông tin.

Control files

Để đảm bảo an toàn, một database cần ít nhất 02 control files và được đặt tại hai chỗ khác nhau. Các control files nên được đặt tên khác nhau sao cho dễ dàng có thể phân biệt.

Tên của Control files nên được đặt kèm với tên của database cho dễ nhớ, như sau:

```
CTL<n><database_name>.ORA
```

Với:

```
n               là số thứ tự của control file
database_name   tên của database
```

Trong parameter file, tên của các control files được đặt phân cách nhau bởi các dấu phẩy.

Ví dụ:

```
control_files =  ("C:\ORANT\DATABASE\CTL1KTKB.ORA",
                 "C:\ORANT\DATABASE\CTL2KTKB.ORA")
```

Online redo log files

Online redo log files thông thường bao gồm nhiều nhóm các online redo log files khác nhau. Với mỗi nhóm chứa các bản sao của các redo log file. Tương tự như control file. Các online redo log file cũng nên được đặt ở các nới khác nhau.

Cũng giống như Control files, việc đặt tên cho các Online redo log files nên được đặt kèm với tên của database cho dễ nhớ, như sau:

```
LOG<n><database_name>.ORA
```

Với:

```
n               là số thứ tự của control file
database_name   tên của database
```

Tên của các control files được đặt phân cách nhau bởi các dấu phẩy.

Ví dụ:

```
logfile =  'C:\ORANT\DATABASE\LOG1KTKB.ORA' SIZE 1024K,
           'C:\ORANT\DATABASE\LOG2KTKB.ORA' SIZE 1024K
```

Datafiles

Tên của datafiles nên được đặt theo như nội dung của nó.

Đối với các data files, ta cần quan tâm tới một số tính chất sau:

- Giảm thiểu việc phân đoạn trong các data files.
- Tách riêng các đối tượng trong database như tách các application data, temporary data trên các tablespaces khác nhau.

Các datafile được phân chia theo các segment khác nhau. Tên của chúng thường được đặt với đuôi là **.DBF** còn phần đầu sẽ được phân theo từng loại segment tương ứng.

Ví dụ:

```
C:\ORANT\DATABASE\KTKB\SYSTEM01.DBF
C:\ORANT\DATABASE\KTKB\RBS01.DBF
C:\ORANT\DATABASE\KTKB\RBS02.DBF
C:\ORANT\DATABASE\KTKB\USERS01.DBF
C:\ORANT\DATABASE\KTKB\TEMP01.DBF
C:\ORANT\DATABASE\KTKB\TOOLS01.DBF
C:\ORANT\DATABASE\KTKB\INDX01.DBF
```

4.2.3. Optimal Flexible Architecture – OFA

Điều quan trọng khi tạo database là tổ chức các file hệ thống sao cho dễ dàng cho việc quản trị, thêm mới và bổ sung các dữ liệu vào database tận dụng hiệu quả các thao tác vào ra của hệ thống.

OFA với các tiện ích giúp cho việc bảo trì database được đơn giản.

Cấu trúc của OFA:

1. Đặt tên các thiết bị để nó có thể chứa đựng các dữ liệu Oracle server giống như một tập hợp.
2. Phân biệt các file sản phẩm, bao gồm các phần mềm và các công cụ Oracle server, các file quản trị, file script khởi tạo,...
3. Lưu lại các phiên bản của các sản phẩm Oracle server
4. Tạo các thư mục lưu trữ dữ liệu Oracle server.

4.2.4. Cấu trúc thư mục phần mềm Oracle

Thư mục	Diễn giải
Bin	Chứa các file sản phẩm ở dạng nhị phân
Dbs	Chứa các file dữ liệu
Lib	Chứa các file thư viện sản phẩm của Oracle
Orainst	Chứa chương trình và các file phục vụ cho việc cài đặt
Rdbms	Các file server, các file thư viện và các file khác cần thiết cho database
Plsql	PL/SQL và các sản phẩm liên quan
Sqlplus	SQL*Plus
Network	Các sản phẩm Oracle Net8
Svrmgrl	Server manager

Cấu trúc thư mục con

Thư mục	Diễn giải
Admin	File scripts quản trị
Demo	File dữ liệu và các scripts minh hoạ
Doc	README file
Install	Các file phục vụ cho việc cài đặt
Lib	Các thư viện sản phẩm
Log	Các file log

4.2.5. Biến môi trường

Trên hệ điều hành Windows, ta thiết lập các biến môi trường. Các biến này tương ứng với các tham số trong registry như: `ORACLE_HOME`, `ORACLE_SID`, `NLS_LANG`.

Để tạo mới database, cần tạo mới biến môi trường `ORACLE_SID`:

```
C:\set ORACLE_SID = U16
```

4.3.CHUẨN BỊ CÁC THAM SỐ TRONG PARAMETER FILE

Khi tạo mới một database, ta cần quan tâm tới việc tạo parameter file. Parameter file chứa các thông tin cần thiết trong database, trong đó quan tâm nhất là các tham số sau:

Tham số	Diễn giải
DB_NAME	Tên định danh của database, tối đa 8 ký tự. Tên database phải trùng với giá trị của biến môi trường ORACLE_SID.
CONTROL_FILES	Liệt kê danh sách các control file sử dụng trong database. Tối thiểu có 01 control file trong database. Tuy nhiên, ta nên tạo 02 control files trở lên để đề phòng hỏng file. Các control files không cần thiết phải tồn tại. Khi tạo database, Oracle sẽ tạo các control files này
DB_BLOCK_SIZE	Xác định kích thước của một block sử dụng trong database. Kích thước này sẽ không thay đổi được sau khi database đã được tạo lập. Kích thước của các block được tính theo đơn vị K (Kilobytes). Kích thước của block thường được đặt bằng số nguyên lần luỹ thừa của 2. để tương ứng với số nguyên lần các block vật lý của hệ điều hành. Do đó, có thể tối ưu được số lần truy xuất đĩa cứng. Ví dụ: 2K, 4K, 8K, 16K, 32K, tuỳ theo phiên bản của Oracle và hệ điều hành.

Thông thường, khi chuẩn bị parameter file của một database sắp được tạo, ta có thể sao chép lại nội dung của parameter file mẫu rồi chỉnh sửa lại một vài thông số trong đó như db_name, control_files,...

Parameter file mẫu của oracle thường được đặt ở thư mục:

```
<%ORACLE_HOME%>\ADMIN\SAMPLES\PFILE
```

Ví dụ về nội dung của file tham số: file InitU16.ora

```
db_name = U16
db_files = 1020
control_files = ("C:\ORANT\database\ctl1U16.ora",
"C:\ORANT\database\ctl2U16.ora")
db_file_multiblock_read_count = 16
db_block_buffers = 2000
shared_pool_size = 30000000
log_checkpoint_interval = 8000
processes = 100
dml_locks = 200
log_buffer = 65536
sequence_cache_entries = 30
sequence_cache_hash_buckets = 23
#audit_trail = true
#timed_statistics = true
background_dump_dest = C:\ORANT\rdbms80\trace
user_dump_dest = C:\ORANT\rdbms80\trace
db_block_size =8192
```

```
compatible = 8.0.4.0.0
sort_area_size = 65536
log_checkpoint_timeout = 0
remote_login_passwordfile = shared
max_dump_file_size = 10240
```

4.4.CHUẨN BỊ INSTANCE PHỤC VỤ QUẢN TRỊ

Sử dụng công cụ ORADIM để tạo instance phục vụ cho việc tạo database. ORADIM sẽ tạo một service dành riêng cho database. Đây là một công cụ thực hiện ở chế độ dòng lệnh. Công cụ này chỉ cần thiết khi user tạo mới, sửa đổi hay huỷ instance của database bằng tay. Trong trường hợp sử dụng công cụ Oracle Database Configuration Assistant để can thiệp vào database thì không cần thiết phải biết tới công cụ này.

ORADIM	Oracle Database Configuration Assistant
Có thể tạo mới, start, stop, sửa đổi hay xoá bỏ instances. Không can thiệp tới database files	Chỉ có thể tạo mới hay huỷ bỏ databse. Không thể start hay stop database
Có thể sử dụng để sửa đổi instance	Không thể để sửa đổi instance
Dùng để tạo password file và service liên quan. Không tạo database được	Dùng để tạo password file và service liên quan, instance và cả database

Lưu ý: Ở các phiên bản trước của Oracle, công cụ ORADIM có tên là ORADIM80

4.4.1. Tạo một instance

Cú pháp:

```
C:\>ORADIM -NEW -SID SID | -SRVC SERVICE_NAME [-INTPWD
           INTERNAL_PWD] - SHUTTYPE SRVC | INST | SRVC, INST [-
           MAXUSERS NUMBER][-STARTMODE AUTO | MANUAL][-PFILE
           FILENAME]
```

Với:

```
-NEW            Tạo mới instance phục vụ cho database.
-SID SID        Tên của instance được tạo (tên này thường được
                lấy chính là tên của database).
-SRVC SERVICE_NAME
                Tên của service phục vụ database.
-INTPWD INTERNAL_PWD
                Mật khẩu của Internal account sử dụng để quản
                trị database
-MAXUSERS NUMBER
                Số lượng user tối đa định nghĩa trong password
                file
-STARTMODE AUTO, MANUAL
                Đặt chế độ khởi động instance phục vụ (khởi
                động service trên máy chủ server)
-PFILE FILENAME
```

```
                    Chỉ rõ parameter file INIT<Database_name>.ORA
-SHUTTYPE SRVC, INST
                    Dừng instance phục vụ (stop service)
```

Ví dụ:

```
C:\> ORADIM -NEW -SID PROD -INTPWD MYPASSWORD1 -STARTMODE AUTO
          -PFILE C:\ORACLE\ADMIN\PROD\PFILE\INIT.ORA
```

4.4.2. Khởi động instance

Cú pháp:

```
C:\ORADIM -STARTUP -SID SID [-USRPWD USER_PWD] [-STARTTYPE SRVC
          | INST | SRVC, INST] [-PFILE FILENAME]
```

Với:

```
-STARTUP        Khởi động instance phục vụ sẵn sàng cho việc
                tạo database.
-SID SID        Tên của instance được tạo (tên này thường được
                lấy chính là tên của database).
-USERPWD USER_PWD
                Mật khẩu.
-STARTTYPE SRVC, INST
                Chế độ khởi động là service hay instance
```

Ví dụ:

```
C:\> ORADIM     -STARTUP -SID PUMA -STARTTYPE SRVC
                -PFILE C:\ORACLE\ADMIN\PROD\PFILE\INIT.ORA
```

4.4.3. Dừng instance

Cú pháp:

```
C:\>ORADIM -SHUTDOWN -SID SID  [-USRPWD USER_PWD] [-SHUTTYPE
           SRVC | INST | SRVC, INST] [-SHUTMODE A | I | N]
```

Với:

```
-SHUTDOWN       Dừng (stop) instance phục vụ.
-SID SID        Tên của instance được tạo (tên này thường được
                lấy chính là tên của database).
-USERPWD USER_PWD
                Mật khẩu.
-SHUTMODE       Xác định chế độ dừng: A – abort mode,        I
                I – Immediate mode, N – Normal mode
```

Ví dụ:

```
C:\> ORADIM -SHUTDOWN -SID PUMA -SHUTTYPE SRVC INST
```

4.4.4. Huỷ instance

Cú pháp:

```
C:\>ORADIM -DELETE -SID sid
```

Ví dụ:

```
C:\> ORADIM -DELETE -SID PUMA
```

4.5.TẠO DATABASE

4.5.1. Khởi động Instance

Sử dụng user với mức quyền DBA. Dùng công cụ `ORADIM` để tạo Instance.

Khởi động Instance ở chế độ `NOMOUNT` và chỉ rõ file tham số sử dụng trong chương trình:

```
SVRMGR> STARTUP NOMOUNT \
      > PFILE=initU16.ora
```

4.5.2. Lệnh tạo database

Sử dụng câu lệnh `CREATE DATABASE` để tạo database

Cú pháp:

```
CREATE DATABASE [database]
      [CONTROLFILE REUSE]
      [LOGFILE [GROUP integer] filespec
      [, [GROUP integer] filespec]...]
      [MAXLOGFILES integer]
      [MAXLOGMEMBERS integer]
      [MAXLOGHISTORY integer]
      [MAXDATAFILES integer]
      [MAXINSTANCES integer]
      [ARCHIVELOG|NOARCHIVELOG]
      [CHARACTER SET charset]
      [NATIONAL CHARACTER SET charset]
      [DATAFILE filespec [autoextend_clause]
      [, filespec [autoextend_clause]...]]

filespec :== 'filename' [SIZE integer][K|M] [REUSE]

autoextend_clause :==
      [AUTOEXTEND {OFF
            |ON [NEXT integer[K|M]]
            [MAXSIZE {UNLIMITED|integer[K|M]}]
            }
      ]
```

Với:

```
Database          Tên của CSDL cần tạo (tên này giống với tên
                  của tham số DB_NAME trong parameter file)
```

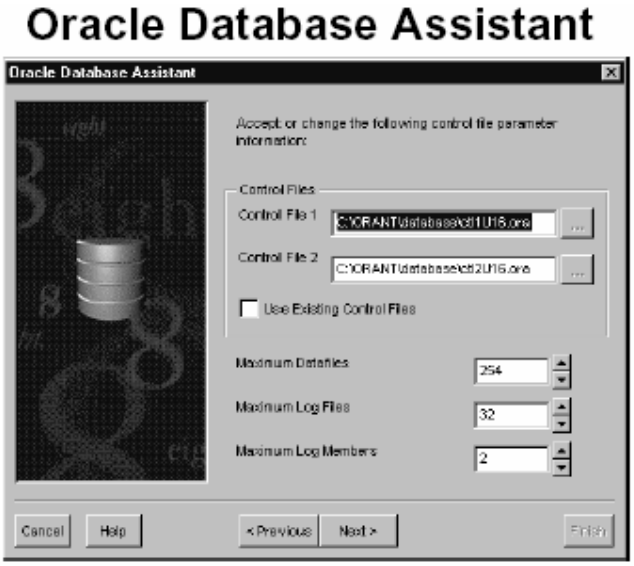
```
CONTROLFILE REUSE
                Tên file tham số đã tồn tại được tái sử dụng
LOGFILE GROUP
                Tên của log file được sử dụng
MAXLOGFILES
                Số lượng tối đa các log file group cho CSDL
MAXLOGMEMBERS
                Số lượng tối đa các log file member đối với
                một log file group
MAXLOGHISTORY
                Số lượng tối đa các redo log trong một group
DATAFILE filespec
                Tên file dữ liệu được sử dụng
AUTOEXTEND
                Cho phép hoặc không cho phép mở rộng tự động
                các file dữ liệu
MAXDATAFILES
                Số lượng tỗi đa các datafiles trong database
MAXINSTANCES
                Số lượng lớn nhất các instance có thể đồng
                thời mount và open database
ARCHIVELOG
                Xác định ràng redo log cần để ở chế độ archive
                trước khi được dùng lại
NOARCHIVELOG
                Xác định ràng redo log cần được dùng lại mà
                không cần đặt chế độ archive
CHARACTER SET, NATIONAL CHARACTER SET
                Chuẩn ký tự mà CSDL sử dụng để lưu trữ các dữ
                liệu
```

Ví dụ: tạo database

```
SPOOL creU16.log

STARTUP NOMOUNT PFILE=initU16.ora

CREATE DATABASE U16
     MAXLOGFILES 5
     MAXLOGMEMBERS 5
     MAXDATAFILES 100
     MAXLOGHISTORY 100
     LOGFILE
     GROUP 1 ('/DISK3/log1a.rdo',/DISK4/log1b.rdo') SIZE 1 M,
     GROUP 2 ('/DISK3/log2a.rdo',/DISK4/log2b.rdo') SIZE 1 M
     DATAFILE
     '/DISK1/system01.dbf' size 50M autoextend on
     CHARACTER SET WE8ISO8859P1;
```

4.5.3. Oracle Database Assistant

Để tạo database, Oracle hỗ trợ công cụ rất tiện lợi giúp người quản trị dễ dàng tạo database hơn thông qua giao diện đồ hoạ, đó là công cụ Oracle Database Assistant.



Hình vẽ 12. Công cụ tạo hỗ trợ database – Oracle Database Assistant

Với công cụ này, người quản trị chỉ việc khai báo các tham số cần thiết cho database. Oracle Database Assistant sẽ tự động kết sinh ra câu lệnh SQL tương ứng với các tham số đã được khai báo. Các câu lệnh SQL có thể được chạy luôn hoặc cũng có thể được lưu lại thành các script files sử dụng sau này.

4.5.4. File script ví dụ tạo một database

File sqlu16.bat

```
set ORACLE_SID=U16
C:\ORANT\bin\oradim -new -sid U16 -intpwd oracle -startmode
auto -pfile C:\ORANT\database\initU16.ora
C:\ORANT\bin\oradim -startup -sid U16 -starttype srvc,inst
-usrpwd oracle -pfile C:\ORANT\database\initU16.ora
C:\ORANT\bin\svrmgr @U16run.sql
```

File U16run.sql

```
spool C:\ORANT\database\spoolmain
set echo on
connect INTERNAL/oracle
startup nomount pfile=C:\ORANT\database\initU16.ora
CREATE DATABASE U16
LOGFILE 'C:\ORANT\database\logU161.ora' SIZE 1024K,
'C:\ORANT\database\logU162.ora' SIZE 1024K
MAXLOGFILES 5
MAXLOGMEMBERS 5
MAXLOGHISTORY 1
DATAFILE 'C:\ORANT\database\Sys1U16.ora' SIZE 50M
```

```
MAXDATAFILES 100
MAXINSTANCES 1
CHARACTER SET WE8ISO8859P1
NATIONAL CHARACTER SET WE8ISO8859P1;
spool off
```

File initU16.ora

```
db_name = U16
db_files = 1020
control_files = ("C:\ORANT\database\ctl1U16.ora",
"C:\ORANT\database\ctl2U16.ora")
db_file_multiblock_read_count = 16
db_block_buffers = 2000
shared_pool_size = 30000000
log_checkpoint_interval = 8000
processes = 100
dml_locks = 200
log_buffer = 65536
sequence_cache_entries = 30
sequence_cache_hash_buckets = 23
#audit_trail = true
#timed_statistics = true
background_dump_dest = C:\ORANT\rdbms80\trace
user_dump_dest = C:\ORANT\rdbms80\trace
db_block_size =8192
compatible = 8.0.4.0.0
sort_area_size = 65536
log_checkpoint_timeout = 0
remote_login_passwordfile = shared
max_dump_file_size = 10240
```

4.5.5. Lỗi xảy ra khi tạo database

Lỗi xảy ra khi tạo database phần lớn do các nguyên nhân sau:

- Lỗi cú pháp lệnh tạo database
- Các file dữ liệu cần tạo lập đã tồn tại
- Lỗi do hệ điều hành, không có đủ quyền, không đủ chỗ trống,...

4.5.6. Kết quả sau khi tạo database

Kết thúc các bước trên ta thu được một database với:

- 02 data files được đặt trong SYSTEM tablespace.
- Các control files và các redo log files phục vụ cho database
- Hai user quản trị database và mật khẩu tương ứng là: `SYS/change_on_install` và `SYSTEM/manager`
- 01 Rollback segment SYSTEM
- Các bảng dữ liệu internal với dữ liệu trống

4.6.TẠO DATA DICTIONARY CHO DATABASE

Trong trường hợp tạo database bằng tay, sau khi tạo xong database, Oracle server sẽ tạo cho ta một database hoàn toàn trống. Các bảng trong database này đều được lưu trữ dưới dạng mã và ta không thể nào quan sát các thông tin trong nó được. Để có thể quan sát được các thông tin trong database. Ta cần tạo data dictionary cho database này.

Data dictionary hay còn gọi là từ điển dữ liệu của database là tập hợp các views được thiết lập trong database cung cấp các thông tin về database.

Các file tạo data dictionary cho database được Oracle cung cấp sẵn và thường được đặt trong thư mục `<%ORACLE_HOME%>\RDBMS\ADMIN`

Các dictionary views được phân loại và đặt trong các file SQL khác nhau.

Một số file SQL hay dùng:

Tên file SQL	Diễn giải
`CATALOG.SQL`	Tạo các dictionary views cơ bản, trigger và store procedure cơ sở
`CATPROC.SQL`	Tạo các package cơ sở
`CATREP.SQL`	Tạo các chức năng Replication cho database

Ngoài ra còn có rất nhiều file script khác.

Chương 5. QUẢN TRỊ ORACLE DATABASE

5.1.PHÂN LOẠI USERS

Oracle là một hệ quản trị cơ sở dữ liệu lớn, chạy trên môi trường mạng. Để vận hành hệ thống được tốt, có thể có nhiều người sẽ cùng tham gia vào hệ thống với những vai trò khác nhau gọi là các user. Có thể phân ra làm một số loại user chính sau:

- Database Administrators
- Security Officers
- Application Developers
- Application Administrators
- Database Users
- Network Administrators

5.1.1. Database Administrators

Do hệ thống Oracle database có thể là rất lớn và có nhiều users cùng tham gia vào hệ thống, và khi đó sẽ có một hay một số người chịu trách nhiệm quản lý hệ thống. Những người có vai trò như vậy được gọi là *database administrator* (DBA). Mỗi một database cần ít nhất 01 người để thực hiện công việc quản trị.

Một database administrator có trách nhiệm thực hiện một số công việc sau:

- Cài đặt và nâng cấp Oracle server và các công cụ ứng dụng khác.
- Phân phối hệ thống lưu trữ và lên kế hoạch lưu trữ cho hệ thống cơ sở dữ liệu trong tương lai.
- Tạo những cấu trúc lưu trữ cơ bản như tablespaces phục vụ cho việc phát triển và hoạt động của các ứng dụng.
- Tạo các đối tượng trong database như tables, views, indexes sử dụng cho các ứng dụng được thiết kế.
- Thay đổi cấu trúc database khi cần thiết tuỳ theo các thông tin của các application.
- Quản lý các users và đảm bảo bảo mật hệ thống.
- Đảm bảo tương thích về bản quyền, phiên bản với hệ thống Oracle
- Điều khiển và quản trị các user access truy xuất tới database.
- QUản lý và tối ưu các truy xuất tới database.
- Lên kế hoạch backup (sao lưu) và recovery (phục hồi) các thông tin có trong database.
- Lưu trữ các archive data.
- Sao lưu và khôi phục database.
- Cập nhật các công nghệ mới đưa ra các câu hỏi bổ ích.

5.1.2. Security Officers

Trong một số trường hợp, hệ thống đòi hỏi chế độ bảo mật cao. Khi đó cần đến một hay một nhóm người chuyên thực hiện công tác bảo vệ database gọi là security officers. Security officer có thể kết nối tới database, điều khiển và quản lý việc truy cập database của các users và bảo mật hệ thống.

5.1.3. Application Developers

Application developer là người thiết kế và viết các ứng dụng database. Application developer có trách nhiệm thực hiện một số yêu cầu sau:

- Thiết kế và phát triển ứng dụng database.
- Thiết kế cấu trúc database cho từng ứng dụng.
- Đánh giá yêu cầu lưu trữ cho ứng dụng.
- Quy định các hình thức thay đổi cấu trúc database của ứng dụng.
- Thiết lập biện pháp bảo mật cho ứng dụng được phát triển.

5.1.4. Database Users

Database users tương tác với database thông qua các ứng dụng và các tiện ích. Một user điển hình có thể thực hiện được một số công việc sau:

- Truy nhập, sửa đổi, và xoá huỷ các dữ liệu được phép
- Tạo các báo cáo đối với dữ liệu

5.1.5. Network Administrators

Đối với database Oracle hoạt đông trên môi trường mạng, khi đó cần có một user thực hiện công việc quản trị mạng. User này có trách nhiệm đảm bảo các ứng dụng Oracle hoạt động trên môi trường mạng được tốt.

5.2.PHƯƠNG THỨC XÁC NHẬN ĐẶC QUYỀN TRUY NHẬP

Việc phân quyền sử dụng là cần thiết trong công việc quản trị. Có hai user account được tự động tạo ra ngay từ khi tạo database và được gán quyền DBA là: SYS và SYSTEM.

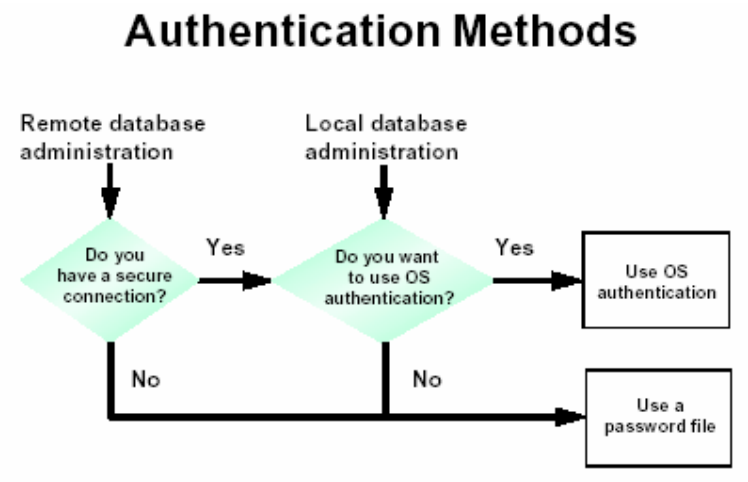
- **SYS:** được tạo tự động và gán quyền DBA. Mật khẩu mặc định là *change_on_install*. Có quyền sở hữu các bảng và các từ điển dữ liệu trong database.
- **SYSTEM:** được tự động tạo ra với mật khẩu ban đầu là *manager* và cũng được gán quyền DBA. Tuy nhiên, SYSTEM còn được sở hữu cả một số table, view mở rộng chứa các thông tin sử dụng cho các tools của Oracle.
- Quyền **DBA**: Ngay khi tạo database, Oracle đã tạo sẵn một quyền gọi là "DBA". Quyền này cho phép thực hiện các thao tác quản trị đối với database.

Lưu ý: Với quyền DBA, các users này sẽ có thể can thiệp được tới các quyền của các user khác sử dụng trong hệ thống. Vì thế, những quản trị viên database cần thay đổi mật khẩu của mình tránh sử dụng mật khẩu mặc định do Oracle cung cấp vì user khác có thể biết và sử dụng để truy nhập vào hệ thống một cách trái phép, gây xáo trộn hệ thống.

5.2.1. Phương thức xác nhận quyền

Trong một số trường hợp quản trị viên database cần đến phương thức xác nhận quyền truy nhập đặc biệt do có thể lúc đó database chưa được mở, ví dụ như với các trường hợp shutdown hoặc startup database.

Tuỳ thuộc vào việc quản trị database trên cùng một máy hay ở máy khác mà ta có thể sử dụng cơ chế xác nhận quyền truy nhập database bởi hệ điều hành hay hay password files.



Hình vẽ 13. Phương thức xác nhận quyền

5.2.2. Xác nhận quyền bởi hệ điều hành

Việc xác nhận quyền bởi hệ điều hành được tiến hành theo các bước:

1. Trong hệ điều hành Windows NT tạo một user's group với tên `ORA_<SID>_DBA` và một nhóm khác `ORA_<SID>_OPER` với `<SID>` tương ứng với tên của instance, hoặc `ORA_DBA` và `ORA_OPER` (khi này ta không quan tâm tới instance).
2. Thêm một user vào group để khi truy cập vào hệ điều hành, user có thể tự động được xác định quyền DBA.
3. Đặt tham số `REMOTE_LOGIN_PASSWORDFILE` trong parameter file là `NONE`.
4. Kết nối tới database với mức quyền `SYSDBA` hay `SYSOPER`:

```
CONNECT / AS { SYSDBA|SYSOPER }
```

Ghi chú:

- NET8 được cài đặt trên các hệ điều hành Windows 95, Windows NT để giúp cho việc xác nhận quyền.
- Các phiên bản trước của Oracle sử dụng lệnh: `CONNECT INTERNAL` với cú pháp: `CONNECT INTERNAL/pw AS SYSDBA`. Lệnh: `CONNECT INTERNAL` hiện tại vẫn được sử dụng.
- Với việc xác nhận quyền truy nhập bởi hệ điều hành, ta không cần quan tâm tới các mức quyền (privilege) thay vào đó, ta cần quan tâm tới hai quyền được cung cấp bởi hệ điều hành là `OSDBA` và `OSOPER`

OSOPER: là quyền cho phép user có thể `STARTUP`, `SHUTDOWN`, `ALTER DATABASE OPEN/MOUNT`, `ALTER DATABASE BACKUP`, `ARCHIVE LOG`, và `RECOVER`, ngoài ra còn có thêm cả quyền `RESTRICTED SESSION`.

OSDBA: là quyền cho phép user có thể có được tất cả các quyền của `OSOPER`, ngoài ra còn có thêm một số mức quyền phục vụ quản trị database là `ADMIN OPTION`, và `CREATE DATABASE`

5.2.3. Xác nhận quyền bằng file mật khẩu

Oracle hỗ trợ các tiện ích password cho phép kết nối tới Oracle Server sử dụng username và password. Việc truy cập vào database sử dụng password file được hỗ trợ bởi lệnh GRANT.

Sử dụng file mật khẩu:

1. Tạo file mật khẩu bằng lệnh:

   ```
   orapwd file=<fname> password=<password> entries=<entries>
   ```

   Với:

fname	là tên file mật khẩu
password	là mật khẩu của SYS hay INTERNAL
entries	là số lượng tối đa các quản trị viên được phép

2. Đặt tham số REMOTE_LOGIN_PASSWORDFILE là EXCLUSIVE hoặc SHARED.

   Với:

EXCLUSIVE	chỉ một instance có thể sử dụng file mật khẩu
SHARED	nhiều instance có thể dùng file mật khẩu

3. Gán quyền cho user

   ```
   GRANT SYSDBA TO admin;
   GRANT SYSOPER TO admin
   ```

4. Kết nối tới database theo cú pháp:

   ```
   SVRMGRL>CONNECT internal/admin AS SYSDBA
   ```

Xem thông tin về các member trong file mật khẩu

Thông tin về các member trong file mật khẩu được lưu trong view: V$PWFILE_USER. Nó cho biết có những user nào được gán quyền SYSDBA hay SYSOPER.

Diễn giải một số cột trong V$PWFILE_USER:

USERNAME	Tên user
SYSDBA	Cột này nhận giá trị TRUE thì User này được gán quyền SYSDBA
SYSOPER	Cột này nhận giá trị TRUE thì User này được gán quyền SYSOPER

Khi kết nối với database theo mức quyền SYSDBA hay SYSOPER, user đó sẽ được kết nối tới các schema mặc định, với SYSDBA thì shema mặc định là SYS, với SYSOPER thì shema mặc định là PUBLIC.

5.2.4. Thay đổi mật khẩu internal

Sử dụng tiện ích ORADIM để tạo lại file mật khẩu.

```
C:\>ORADIM -NEW -SID sid [-INTPWD internal_pwd][SRVC
          svrcname][MAXUSERS n][STARTMODE auto, manual][-PFILE
          filename]
```

Với:

sid	tên instance
internal_pwd	mật khẩu internal account
svrcname	tên service
n	số lượng tối đa file mật khẩu
auto or manual	chế độ khởi động service là: manual hay automatic
filename	cho phép sử dụng file mật khẩu không phải là mặc định

Để thay đổi mật khẩu INTERNAL ta thực hiện theo các bước sau:

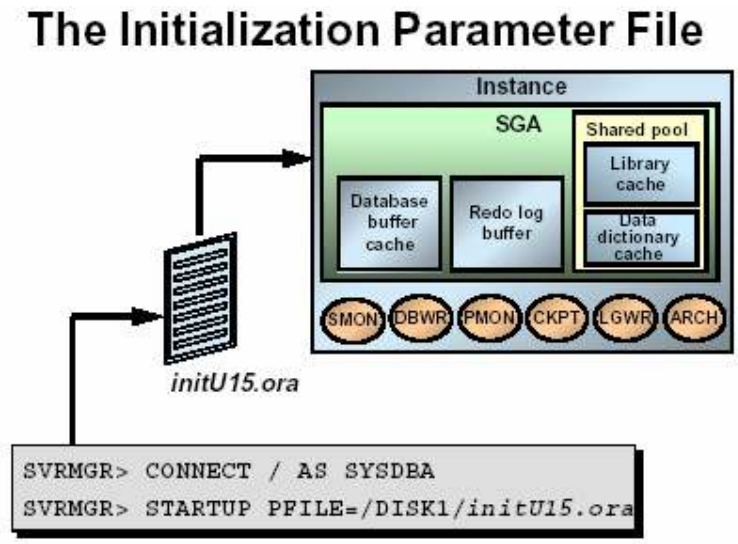
1. Xoá mật khẩu cũ

```
C:\> ORADIM -DELETE -SID sid
```

2. Tạo mật khẩu mới

```
C:\> ORADIM -NEW -SID sid -INTPWD internal_pwd - MAXUSERS N
```

5.3.TẠO PARAMETER FILE



Hình vẽ 14. Khởi tạo tham số

File tham số thông thường có tên init<SID>.ora.

Theo mặc định, file tham số được đặt trong thư mục %ORACLE_HOME%\DATABASE.

File tham số chỉ được đọc một lần khi khởi động instance. Khi thay đổi nội dung của file tham số, để sử dụng được các giá trị mới cần shut down rồi sau đó restart lại instance.

5.3.1. Sử dụng các tham số

Các tham số có thể ảnh hưởng tới hiệu quả sử dụng database. Các thông số trong file tham số bao gồm:

- Kích thước của vùng System Global Area (SGA) để tối ưu hiệu suất.
- Đặt mặc định cho database và instance.
- Đặt các hạn chế đối với user hay process.
- Đặt các hạn chế đối với tài nguyên database.

- Xác định các thuộc tính vật lý của database, như kích thước của block.
- Chỉ ra các control files, archived log files, Alert file, và trace file locations.

5.3.2. Một số quy tắc đối với các tham số

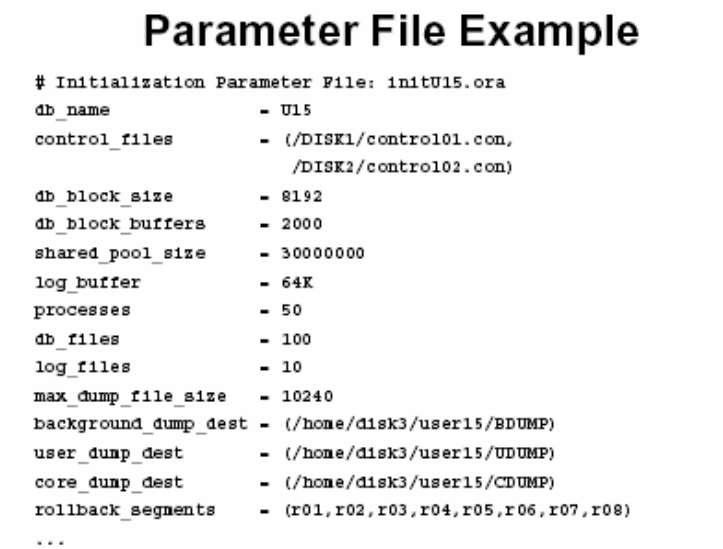
- Các giá trị được chỉ ra theo khuôn dạng: `<Keyword> = <Giá trị>`.
- Một số tham số đều là tuỳ chọn và một số khác là bắt buộc ví dụ như `DB_NAME`.
- Server đều có giá trị mặc định đối với mỗi tham số. Các giá trị này là tuỳ theo hệ điều hành và tuỳ theo tham số.
- Các tham số có thể được chỉ ra không cần phải tuân theo một thứ tự nào cả (đặt trước, sau không quan trọng).
- Phần chú dẫn được bắt đầu bằng ký hiệu #.
- Các tham số là ký tự được đặt trong dấu nháy kép.
- Cũng có thể included các file bởi từ khoá IFILE.
- Các giá trị là tập hợp được đặt trong dấu ngoặc đơn '(,)'và được ngăn cách nhau bởi dấu phẩy (,).

5.3.3.

5.3.4. Các tham số cơ bản

Tham số	Diễn giải
CONTROL_FILES	Tên của các control files.
DB_BLOCK_BUFFERS	Số lượng các data blocks được cach trong SGA.
DB_BLOCK_SIZE	Kích thước của một data block. Kích thước này nên được chọn bằng số số nguyên lần mũ 2, có thể là 2K, 4K, 8K, 16K và 32K tuỳ theo phiên bản của Oracle và của Hệ điều hành.
DB_NAME	Định danh database từ 8 ký tự trở xuống. Tham số này chỉ cần thiết khi tạo mới một database.
IFILE	Tên của file tham số được include vào file tham số hiện thời. Cho phép có thể được lồng tối đa là ba cấp.
LOG_BUFFER	Số byte được cấp phát cho redo log buffer trong SGA.
MAX_DUMP_FILE_SIZE	Kích thước tối đa của trace files, được xác định bằng số lượng block của hệ điều hành.
OPEN_CURSOR	Số lượng cursor tối đa được đồng thời mở.
ROLLBACK_SEGMENTS	Số lượng rollback segments được sử dụng cho mỗi instance
PROCESSES	Số lượng tối đa các tiến trình hệ điều hành có thể kết nối với instance.
SHARED_POOL_SIZE	Kích thước của Shared Pool

Ví dụ một parametersfile:



Hình vẽ 15. File tham số ví dụ

5.4.START VÀ SHUT DOWN DATABASE

5.4.1. Các bước Start và Shut down database

Start Instance ở chế độ Nomount

Ta có thể khởi động một Instance mà không cần thiết phải gắn với một database cụ thể. Khi khởi động Instance, các công việc sau đây sẽ được thực hiện:

- Đọc file tham số : `init<SID>.ora`
- Thu xếp vùng bộ nhớ SGA
- Khởi động các background process
- Mở các trace file và các Alert file

Lưu ý: Tên database nằm trong tham số `DB_NAME` của file tham số.

Câu lệnh:

```
STARTUP NOMOUNT;
```

Start Instance ở chế độ mount

Để thực hiện một vài thao tác đặc biệt khi vận hành database, ta có thể khởi động một instance và mount database nhưng chưa mở database.

Ví dụ như:

- Đổi tên datafiles
- Enable hoặc Disable các redo log files
- Thực hiện phục hồi dữ liệu (recovery).

Các công việc khi mount database:

- Gắn database với một instance đã khởi động
- Định vị và mở các control files theo như thông số có trong file tham số
- Đọc nội dung của control file và xác định trạng thái cho các data files và các redo log files.

Câu lệnh:

```
STARTUP MOUNT;
```

Start Instance ở chế độ open

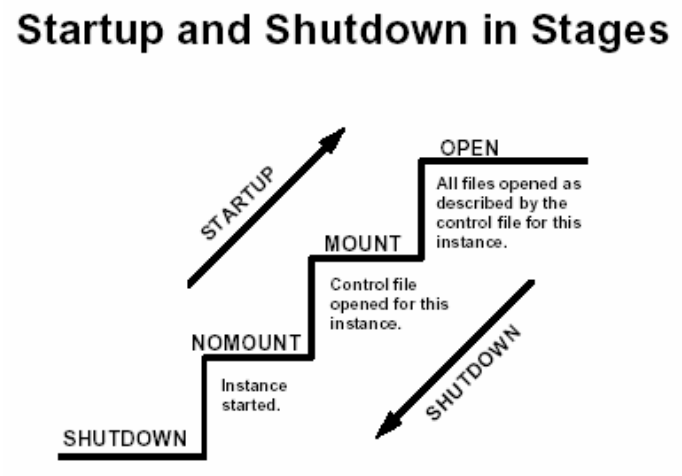
Sau khi databsae đã được mở, những người sử dụng hợp lệ có thể kết nối tới database và thực hiện các thao tác truy nhập vào database.

Việc mở database diễn ra theo hai bước:

- Mở các online data files
- Mở các online redo log files.

Câu lệnh:

```
STARTUP OPEN;
```



Hình vẽ 16. Các bước khởi động và dừng Instance

Khôi phục Instance

Trong một số trường hợp Instance có thể gặp lỗi và không thể làm việc được. Ví dụ như: có lỗi hệ thống xảy ra. Việc khôi phục Instance sẽ được thực hiện theo các bước sau:

- Khôi phục lại tất cả các dữ liệu có thể khôi phục được (dữ liệu chưa được lưu vào data files nhưng đã lưu vào trong online redo log files)
- Mở database.
- Khôi phục lại tất cả các transaction chưa được commit.

Close database

Đây là bước đầu tiên khi tắt hẳn một database. Sau khi đóng database, tất cả các dữ liệu còn trong bộ đệm (redo log buffer cache) sẽ được ghi ra file (online redo log file). Các control file vẫn được mở.

Dismount database

Dissmount database sẽ đóng nốt các control file thuộc database đang mở.

Shoutdown Instance

Đây là bước cuối cùng, intance sẽ được tắt hẳn. Các trace file và Alert file của instance bị đóng. Các background process bị dừng và vùng nhớ SGA cấp cho instance bị thu hồi.

5.4.2. Start database

Cú pháp:

```
STARTUP [FORCE] [RESTRICT] [PFILE=filename]
      [EXCLUSIVE | PARALLEL | SHARED]
      [OPEN [RECOVER][database]|MOUNT |NOMOUNT]
```

Với:

OPEN	cho phép các users truy cập vào database.
MOUNT	mounts database sẵn sàng cho các thao tác DBA, người sử dụng chưa truy cập được database.
NOMOUNT	Bố trí SGA và khởi động các background process, chưa sẵn sàng cho DBA.
EXCLUSIVE	chỉ cho phép instance hiện thời truy cập vào database.
PARALLEL	cho phép nhiều instances cùng được gắn với database (sử dụng Oracle Parallel Server)
SHARED	tương tự như PARALLEL.
PFILE=parfile	cho phép sử dụng file tham số không phải là mặc định để xác định cấu hình cho instance.
FORCE	huỷ bỏ các instance đang chạy trước đó, khởi động instance bình thường.
RESTRICT	chỉ cho phép các users truy cập với chế độ RESTRICTED.
SESSION	quyền truy nhập vào database.
RECOVER	bắt đầu khôi phục dữ liệu khi database.

5.4.3. Thay đổi tính sẵn dùng của database hiện thời

Khởi động database ở chế độ NOMOUNT

Thực hiện sửa đổi database theo lệnh:

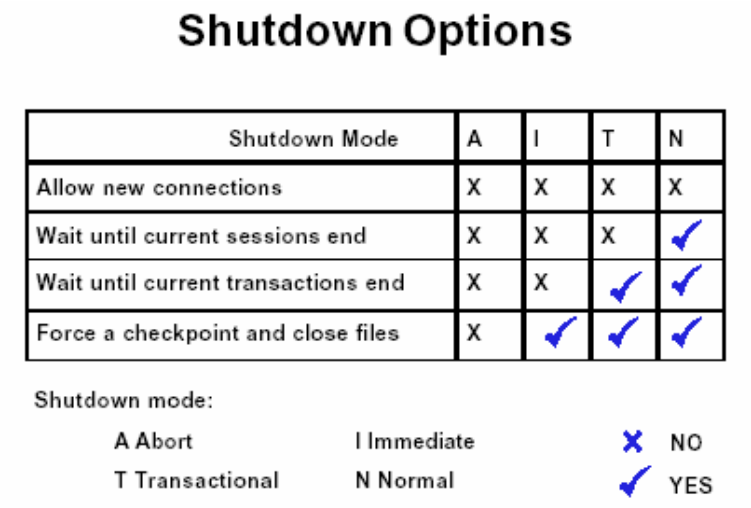
```
ALTER database { MOUNT | OPEN |
                 OPEN READ ONLY | OPEN READ WRITTE}
```

Với:

MOUNT	Gắn database với instance. Lúc này ta chỉ có thể thực hiện các thao tác quản trị trên database mà chưa thể sử dụng database được.
OPEN READ WRITE	Mở database, sẵn sàng cho việc sử dụng database, cả đọc lẫn ghi.
OPEN READ ONLY	Mở database nhưng chỉ cho đọc database như sử dụng các câu lệnh truy vấn chẳng hạn. Các thao tác ghi không thể thực hiện được. Tuỳ chọn này được sử dụng khi ta cần sao chép các redo log files của database.
OPEN	Tương tự như OPEN READ ONLY, đây là biểu diễn mặc định của OPEN READ WRITE.

5.4.4. Shut down database

Có một số chế độ tắt database tương ứng với các khả năng khác nhau.



Hình vẽ 17. So sánh các chế độ tắt database

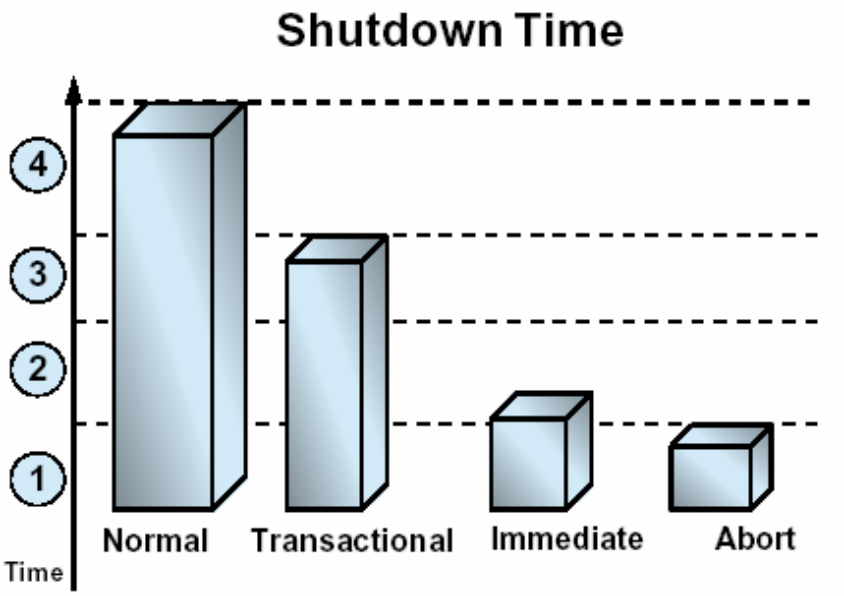
Cú pháp:

```
SHUTDOWN [NORMAL | TRANSACTIONAL | IMMEDIATE | ABORT ]
```

Với:

NORMAL	Không cho tạo thêm các connection tới database, chờ cho connection hiện thời kết thúc thì shutdown database.
TRANSACTION	Không cho phát sinh thêm các transaction, chờ cho transaction hiện thời kết thúc thì shutdown database.
IMMEDIATE	Kết thúc luôn transaction hiện thời nhưng vẫn chờ hệ thống commit hay rollback rồi mới shutdown database.
ABORT	Shutdown database tức thời không đòi hỏi bất cứ điều kiện gì.

Tương ứng với các cách tắt database trên, ta có biểu đồ về thời gian như sau:



Hình vẽ 18. So sánh thời gian giữa các cách tắt database

Hình vẽ trên so sánh tiêu tốn về thời gian khi thực hiện một thao tác chuyển đổi dữ liệu:

1. Thực hiện truy vấn để lấy dữ liệu
2. Thực hiện lệnh `INSERT` và `DELETE` để cập nhật và chuyển đổi dữ liệu
3. Phát lệnh `COMMIT` để cập nhật dữ liệu vào database
4. Huỷ bỏ liên kết tới database.

5.4.5. Thay đổi trạng thái của database

Cú pháp:

```
ALTER system { SUSPEND | RESUME }
SUSPEND          Đưa database vào trạng thái treo. Tạm thời
                 không cho phép thực hiện các thao tác vào ra
                 đối với datafiles và control files. Thao tác
                 này được thực hiện khi ta chuẩn bị backup
                 database.
RESUME           Ngược lại với SUSPEND, thao tác này sẽ đưa
                 database trở lại trạng thái bình thường sau
                 khi đã backup xong database.
```

Ví dụ:

```
SQL> ALTER SYSTEM SUSPEND;
System altered
SQL> SELECT database_status FROM v$instance;
DATABASE_STATUS
---------
SUSPENDED

SQL> ALTER SYSTEM RESUME;
System altered
SQL> SELECT database_status FROM v$instance;
```

```
DATABASE_STATUS
---------
ACTIVE
```

5.4.6. Tạm treo và phục hồi Database

Oracle9*i* cung cấp chức năng suspend/resume. Quản trị viên sử dụng lệnh `ALTER SYSTEM SUSPEND` để tạm treo database, dừng mọi thao tác truy xuất vào ra đối với các datafiles và control files. Khi database ở trạng thái tạm treo, các thao tác vào ra (`I/O operations`) đang thực hiện sẽ được kết thúc và những truy cập vào database mới phát sinh sẽ được đẩy vào queue. Thực hiện lệnh `ALTER SYSTEM RESUME` để khôi phục lại tình trạng bình thường của database.

Ta sử dụng lệnh `ALTER SYSTEM SUSPEND` để tạm treo một database, ngăn thực hiện các thao tác vào ra (`I/O`) đối với các datafiles và control files. Do đó, cho phép database có thể dễ dàng thực hiện các thao tác back up. Khi thực hiện việc treo database tất cả các thao tác vào ra đang có sẽ được tiếp tục cho phép thực hiện cho đến khi hoàn tất, các phép thao tác vào ra mới phát sinh sau này sẽ được tạm thời đưa vào queue chờ xử lý sau.

Lệnh suspend (tạm treo) database được thực hiện đối với database chứ không phải chỉ đối với instance. Do vậy, ở trong môi trường Oracle Real Application Clusters, một khi lệnh suspend được phát ra thì sau đó một cơ chế khoá sẽ được thiết lập và chặn tất cả các yêu cầu gửi tới instance.

Sử dụng lệnh `ALTER SYSTEM RESUME` để phục hồi (resume) lại các hoạt động thông thường của database. Ta cũng có thể chỉ rõ `SUSPEND` và `RESUME` từ các instances khác nhau. Ví dụ, nếu các instances 1, 2, và 3 đang chạy, và ta phát lệnh `ALTER SYSTEM SUSPEND` từ `instance 1`, sau đó ta cũng có thể phát lệnh `RESUME` từ `các instances 1, 2`, hay `3` đều như nhau.

Khả năng `suspend/resume` là rất hữu ích cho hệ thống nó cho phép ta thực hiện mirror một ổ đĩa hay một file rồi sau đó sử dụng vào việc sao lưu, phục hồi dữ liệu cho toàn bộ hệ thống. .

Tuy vậy, đặc điểm `suspend/resume` không thay thế cho các thao tác `normal shutdown database` vì khi đó việc sao chép `database` được `suspend` có thể chứa cả các dữ liệu cập nhật chưa được `commit`.

Cau lệnh sau minh hoạ việc sử dụng lệnh `ALTER SYSTEM SUSPEND/RESUME`. Sử dụng thông tin cung cấp trong `V$INSTANCE` để biết được trạng thái của `database`.

```
SQL> ALTER SYSTEM SUSPEND;
System altered
SQL> SELECT DATABASE_STATUS FROM V$INSTANCE;
DATABASE_STATUS
---------
SUSPENDED


SQL> ALTER SYSTEM RESUME;
System altered
SQL> SELECT DATABASE_STATUS FROM V$INSTANCE;
DATABASE_STATUS
---------
ACTIVE
```

5.4.7. Đặt chế độ hoạt động tĩnh cho database

Oracle9*i* cho phép đưa database vào chế độ hoạt động tĩnh (`quiesced state`), Theo đó chỉ cáo các `DBA transactions, queries`, và các lệnh `PL/SQL` là được phép thực hiện. Trạng thái này cho phép người dùng thực hiện các thao tác quản trị một cách an toàn. Sự dụng câu lệnh `ALTER SYSTEM QUIESCE RESTRICTED` để đưa database về chế độ hoạt động tĩnh.

5.5. ĐẶT TRẠNG THÁI TĨNH CHO DATABASE

Có nhiều khi ta cần phải đưa database vào trạng thái mà chỉ có các `DBA transactions`, `queries` (truy vấn), `fetches` (tìm kiếm dữ liệu), hay các câu lệnh `PL/SQL` là được phép thực hiện. Chế độ này được gọi là quiesced state - tạm dịch là chế độ tĩnh. Chế độ này cho phép quản trị viên có thể thực hiện một số thao tác không an toàn lắm trên database bao gồm các thao tác sau đây:

- Các thao tác có thể gặp lỗi nếu đồng thời có một user transactions truy cập vào cùng một đối tượng. Ví dụ như khi thay đổi table, thêm mới cột dữ liệu vào một table đang có và không yêu cầu khoá (`no-wait lock is required`).
- Các thao tác không mong muốn gây ảnh hưởng tức thì giữa các `user transactions` xảy ra đồng thời. Ví dụ khi có một thủ tục chứa nhiều bước thao tác trên một table chẳng hạn như table ban đầu được export dữ liệu, rồi bị xoá đi và cuối cùng lại được import dữ liệu trở lại. Cùng lúc đó có user khác muốn truy cập vào table và ngay tại thời điểm table vừa bị huỷ. Khi này sẽ phát sinh lỗi hệ thống.

Nếu không áp dụng trạng thái tĩnh cho database, thì ta cần phải shutdown database rồi open lại nó ở chế độ restrict. Và việc này sẽ trở nên nghiêm trọng hơn khi hệ thống yêu cầu phải chạy liên tục 24 x 7. Áp dụng chế độ tĩnh cho database sẽ giảm bớt đi các hạn chế vì restriction vì nó loại bớt đi được các xấu xảy ra với database.

5.5.1. Đưa Database vào trạng thái tĩnh

Để đưa database vào trạng thái tĩnh, đơn giản ta chỉ cần sử dụng lệnh:

```
ALTER SYSTEM QUIESCE RESTRICTED
```

Tất cả các `non-DBA active sessions` sẽ được tiếp tục xử lý cho tới khi chúng chuyển sang trạng thái `inactive`. Một session được xem là active nếu lúc đó nó đang có các phép thực như transaction, query, fetch, hay đang xử lý một câu lệnh `PL/SQL`; hoặc cũng có thể là session đó đang nắm giữ phần tài nguyên chia sẻ (`shared resources`).

Khi tất cả các non-DBA sessions chuyển sang trạng thái inactive, câu lệnh `ALTER SYSTEM QUIESCE RESTRICTED` kết thúc và database được xem như là chuyển sang trạng thái tĩnh quiesce state. Trong môi trường `Oracle Real Application Clusters`, câu lệnh này có ảnh hưởng tới tất cả các instances, chứ không chỉ là đối với instance nơi phát ra câu lệnh.

Lệnh `ALTER SYSTEM QUIESCE RESTRICTED` có thể phải chờ trong một thời gian khá dài để cho active sessions chuyển sang trạng thái inactive. Nếu ta huỷ bỏ yêu cầu, hoặc nếu session bị kết thúc một cách đột ngột vì nhiều lý do khác nhau thì Oracle sẽ tự động phục hồi lại (undo) trạng thái trước khi thực hiện lệnh.

Nếu một truy vấn được đưa ra bởi các Oracle Call Interface (OCI), thì câu lệnh `ALTER SYSTEM QUIESCE RESTRICTED` sẽ không chờ fetch hết tất cả các dữ liệu mà chỉ chờ fetch xong dòng dữ liệu hiện thời mà thôi.

Khi ở trạng thái quiesce state, ta không sử dụng hệ điều hành để sao chép các file trong hệ thống giống như khi thực hiện backup lạnh đối với database, cho dù ta có các checkpoint tại mỗi một instance. Lý do là vì khi ở trạng thái quiesce state thì các file headers của online datafiles vẫn luôn được liên tục truy cập.

5.5.2. Phục hồi hệ thống trở lại hoạt động như bình thường

Thực hiện câu lệnh sau:

```
ALTER SYSTEM UNQUIESCE
```

Khi này tất cả các non-DBA activity sẽ được tiếp tục thực hiện. Trong môi trường Oracle Real Application Clusters, ta có thể phát lệnh này từ bất kỳ một instance nào có kết nối tới server không nhất thiết phải là instance phát lệnh đặt trạng thái tĩnh. Trong trường hợp session phát lệnh `ALTER SYSTEM UNQUIESCE` gặp lỗi, Oracle database server sẽ luôn đảm bảo việc thực hiện unquiesce sẽ kết thúc.

5.5.3. Xem trạng thái của database

Ta có thể xem trạng thái của database qua các thông tin có trong `V$INSTANCE` . Các thông tin này được lưu trong cột `ACTIVE_STATE` với các nội dung như sau:

ACTIVE_STATE	Diễn giải
NORMAL	Trạng thái thông thường
QUIESCING	Đang ở trạng thái tĩnh – quiesce state, nhưng các active non-DBA sessions vẫn được thực hiện
QUIESCED	Ở trạng thái quiesce state, và không có bất kỳ một active non-DBA sessions nào được phép thực hiện

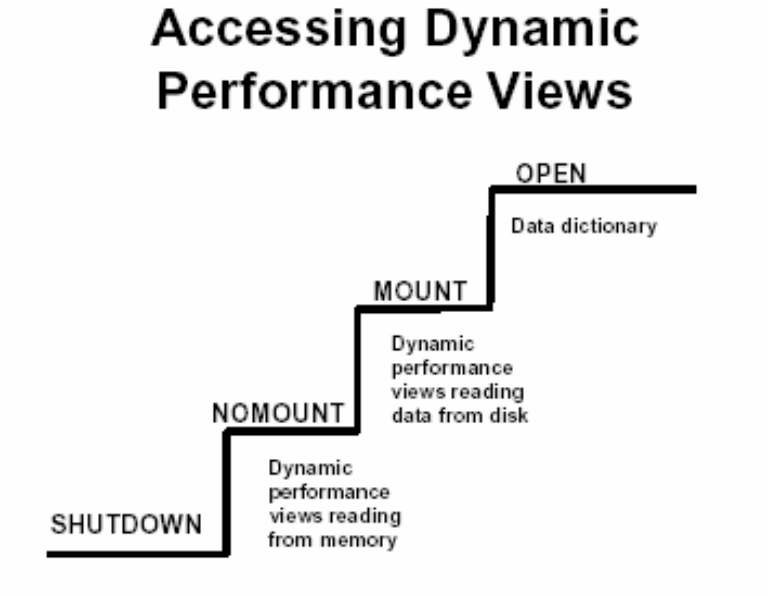
5.6.LẤY CÁC THÔNG TIN VỀ HỆ THỐNG

Các thông số hệ thống được đặt trong các tables hệ thống. Ta có thể quan sát và truy xuất tới chúng thông qua các view gọi là Dynamic performance views. Các view này thường có tên viết đầu là `V_$`. Oracle thường tạo ra các Sysnonym tương ứng với các view này với tên có đầu là `V$`.

Khi khởi động database ở chế độ `NOMOUNT`, user quản trị có thể đọc được các dữ liệu có trong các view này. Thông tin trong view này là cần thiết cho việc mount database.

View `V$FIXED_TABLE` chứa tên của tất cả các view `V$` có trong hệ thống.

Biểu đồ dưới đây diễn tả các mức độ truy cập các view của hệ thống



Hình vẽ 19. Các mức độ truy cập view hệ thống

5.6.1. Một số views cần quan tâm

Dynamic Performance View	Diễn giải
V$PARAMETER	Thông tin về các tham số khởi tạo
V$SGA	Thông tin tổng hợp về SGA
V$OPTION	Các tuỳ chọn cho Oracle server đã được cài đặt
V$PROCESS	Thông tin về các hoạt động của process hiên thời
V$SESSION	Thông tin về session
V$VERSION	Thông tin về phiên bản của các thành phần Oracle
V$INSTANCE	Thông tin về trạng thái của Instance hiện thời
V$THREAD	Thông tin về các thread trong hệ thống
V$CONTROLFILE	Liệt kê tên của các control files
V$DATABASE	Thông tin về database
V$DATAFILE	Thông tin về các data file được sử dụng
V$DATAFILE_HEADER	Thông tin header của các data file được sử dụng
V$LOGFILE	Thông tin về các online redo log files

5.6.2. Hiển thị giá trị của các thông số hệ thống

Ta có thể xem thông tin hệ thống bằng hai cách:

- Sử dụng lệnh xem tham số của Server manager.

```
SVRMGRL> SHOW PARAMETER control
```

- Truy xuất trực tiếp vào view hệ thống

```
SELECT name, type from v$control WHERE name like
      '%control%';
```

Với hai cách trên ta đều thu được một kết quả:

```
SVRMGR> SHOW PARAMETER control
NAME
                                 TYPE        VALUE
------------------------------ -------   --------------------
control_file_record_keep_time    integer   7
control_files                    string    /DISK1/control01.con
```

5.6.3. Tham số hệ thống động (có thể thay đổi)

Trong các tham số hệ thống, có một vài tham số là động và ta có thể thay đổi được các tham số này. Thông qua các lệnh:

- `ALTER SESSION`: chỉ thay đổi giá trị của các tham số trong session hiện thời
- `ALTER SYSTEM`: thay đổi giá trị trong toàn bộ hệ thống nói chung.
- `ALTER SYSTEM DEFERRED`: chỉ thay đổi tham số hệ thống của các session sẽ kết nối vào database sau này, kể từ sau thời điểm thay đổi.

Cú pháp:

```
ALTER SESSION SET parameter_name = value
ALTER SYSTEM SET parameter_name = value [DEFERRED]
```

Ví dụ:

```
ALTER SESSION SET SQL_TRACE=true;
ALTER SYSTEM SET TIMED_STATISTICS=true;
ALTER SYSTEM SET SORT_AREA_SIZE=131072 DEFERRED;
```

Xem lại thông tin mà ta vừa thay đổi:

```
SVRMGR> SELECT isses_modifiable,issys_modifiable,
3> ismodified, name
2> FROM v$system_parameter
4> WHERE ismodified != 'FALSE';
ISSES      ISSYS_MOD ISMODIFI NAME
-----      --------- -------- ------------------------------
TRUE       IMMEDIATE MODIFIED timed_statistics
1 row selected.
```

5.6.4. Quản lý session

Restrict session

Restrict session cần thiết khi bảo trì cơ sở dữ liệu, import, export và sửa đổi cấu trúc của database.

Ta có thể đặt chế độ cho restrict session cho database thông qua lệnh:

```
ALTER SYSTEM {ENABLE|DISABLE}RESTRICTED SESSION
```

Với:

```
ENABLE RESTRICTED
                      chỉ cho phép các users  có quyền
                      RESTRICTED SESSION  truy nhập
DISABLE RESTRICTED SESSION
                      cho phép tất cả các users truy nhập vào
                      database
```

Kết thúc session

Ta có thể kết thúc (Terminate) các session của một Instance đã ở trong chế độ restrict, trước khi thực hiện các thao tác quản trị.

Cú pháp:

```
ALTER SYSTEM KILL SESSION 'integer1,integer2'
```

Với:

```
KILL SESSION     tên session cần kết thúc
integer1         giá trị của cột SID trong view v$session
integer2         giá trị của cột SERIAL# trong view v$session
```

Chú ý: hai giá trị integer1 và integer2 dùng để xác định session

Với lệnh `KILL SESSION` background process `PMON` sẽ thực hiện các công việc sau:

- Rollback transaction hiện thời của user
- Giải phóng tất cả các lock trên các table thực hiện bởi user đó
- Giải phóng các tài nguyên sử dụng bởi user

5.6.5. Trace file và ALERT file

Trace file lưu trữ các thao tác bởi background process. Các thông tin về lỗi trong hệ thống sẽ được lưu vào đây. Điều này là rất hữu ích khi thực hiện dò tìm và khắc phục lỗi xảy ra trong hệ thống.

Trong khi chạy Oracle Instance, tất cả các message phát ra đối với hệ thống đều được lưu vào Alert file. Trong quá trình khởi động database, Oracle sẽ tự tạo ra Alert file nếu nó chưa tồn tại.

Trong trường hợp có lỗi xảy ra, các background process sẽ thực hiện ghi lại các thông tin dump vào trace file.

Ta có thể đặt lại chế độ ghi lỗi ra trace file thông qua lệnh:

```
SQL>ALTER SESSION SET sql_trace=TRUE;
```

Đường dẫn tới các trace file và Alert có thể được chỉ ra bởi các tham số:

```
BACKGROUND_DUMP_DEST
                Xác định nơi đặt của các trace file và ALERT.
USER_DUMP_DEST
                Xác định nơi tạo các trace files.
MAX_DUMP_FILE_SIZE
                Số lượng block của hệ điều hành quy định kích
                thước của trace files.
```

Chương 6. DATA DICTIONARY, VIEWS VÀ PACKAGES

6.1.DATA DICTIONARY VÀ VIEWS

6.1.1. Data Dictionary

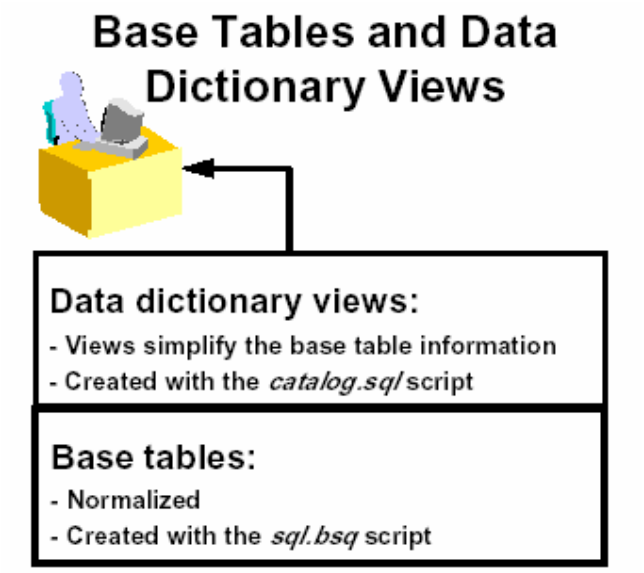
Data dictionary hay từ điển dữ liệu hệ thống là phần rất quan trọng trong Oracle database. Đó là một tập hợp các table và các view sử dụng cho việc tham chiếu đến các thông tin liên quan tới database. Data dictionary được tạo bởi file script *sql.bsq* trong quá trình tạo database.

Data dictionary bao gồm các thông tin trung tâm của Oracle server.

Data dictionary được Oracle server tự động cập nhật mỗi khi thực hiện lệnh định nghĩa dữ liệu (Data Definition Language – DDL).

Data dictionary đặt trong tablespace SYSTEM do User SYS quản lý. Data dictionary bao gồm hai loại sau:

- Base tabes
- Data dictionary Views



Hình vẽ 20. Dictionarytrong database

Base tables

Thông tin trong data dictionary được xác định từ các thông tin có trong các base tables (bảng cơ sở). Nội dung của các bảng này do Oracle server cập nhật. User thuộc database hầu như không thể cập nhật các thông tin này do chúng là các thông tin đã được chuẩn hoá và được mã hoá. Ví dụ: ta chỉ có thể truy xuất tới các thông tin có trong bảng `IND$` để biết được các thông tin về các indexes đã được định nghĩa trong database, hoặc lấy các thông tin trong bảng `OBJ$` để biết được các objects đã được định nghĩa trong database.

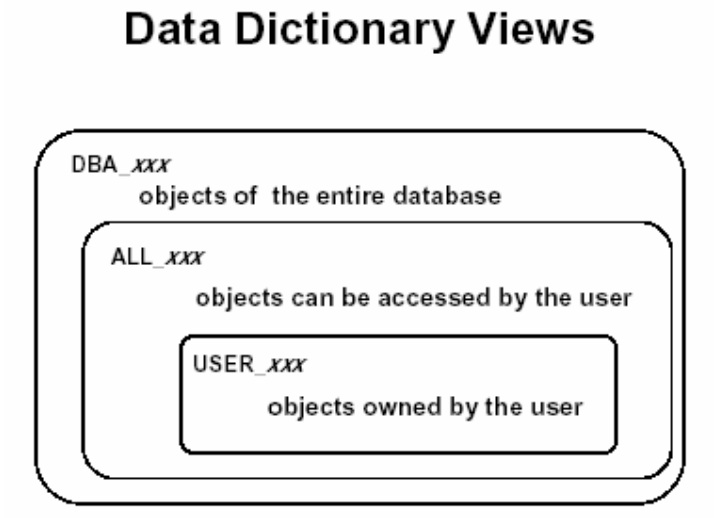
Ta không thể sử dụng các câu lệnh thao tác dữ liệu như `INSERT`, `UPDATE`, hay `DELETE` để thay đổi nội dung thông tin trong các bảng cơ sở một cách trực tiếp ngoại trừ bảng `AUD$` (Xem thêm phần kiểm tra - Auditing).

Data Dictionary Views

Data dictionary views được tạo ra bởi các câu lệnh có trong file script *catalog.sql*. Các views này giải mã và tổng hợp các thông tin có trong các base tables. Để dễ dàng truy xuất các thông tin này, các data dictionary thường được tạo các synonyms tương ứng.

Phần lớn các thông tin hệ thống được User lấy về từ các data dictionary views hơn là lấy trực tiếp từ các base tables.

6.1.2. Data Dictionary views



Hình vẽ 21. Dictionary views

Data dictionary views được phân ra làm ba loại chứa các thông tin tương tự nhau nhưng ở các mức độ khác nhau. Các loại data dictionary views này được phân biệt bởi các tiếp đầu ngữ khác nhau.

Tiếp đầu ngữ USER

Các views có tiếp đầu ngữ `USER` chứa thông tin về các objects do User hiện thời sở hữu. Ví dụ: `USER_TABLES` sẽ chứa thông tin về các bảng dữ liệu của User hiện thời.

Tiếp đầu ngữ ALL

Các views có tiếp đầu ngữ `ALL` chứa thông tin về các objects có thể truy cập bởi User hiện thời, bao gồm cả các đối tượng do User đó sở hữu và cả các đối tượng khác mà User được gán quyền truy nhập. Ví dụ: `ALL_TABLES` sẽ chứa thông tin về các bảng dữ liệu mà User hiện thời có thể truy nhập.

Tiếp đầu ngữ DBA

Các views có tiếp đầu ngữ `DBA` chứa thông tin về các objects có trong database. Các views này là cần thiết cho quản trị viên database. Một User bất kỳ cũng có thể xem được thông tin trong các views `DBA` nếu user đó được cấp quyền `SELECT ANY TABLE`.

Phân loại một số loại views

Tên View	Diễn giải
DICTIONARY DICT_COLUMNS	Thông tin chung
DBA_TABLES DBA_OBJECTS DBA_LOBS DBA_TAB_COLUMNS DBA_CONSTRAINTS	Thông tin liên quan tới các đối tượng của User như: table, Column, Constraint,...
DBA_USERS DBA_SYS_PRIVS DBA_ROLES	Thông tin về mức quyền của User

Tên View	Diễn giải
DBA_EXTENTS DBA_FREE_SPACE DBA_SEGMENTS	Tình hình cấp phát không gian cho các đối tượng trong database.
DBA_ROLLBACK_SEGS DBA_DATA_FILES DBA_TABLESPACES	Thông tin về cấu trúc database
DBA_AUDIT_TRAIL DBA_AUDIT_OBJECTS DBA_AUDIT_OBJ_OPTS	Các thông tin kiểm tra

Ví dụ: Để lấy các thông tin chung trong từ điển dữ liệu, ta có thể truy vấn trong Các views DICTIONARY hoặc DICT_COLUMNS.

```
SVRMGR>SELECT *
2> FROM dictionary
3> WHERE table_name LIKE '%TABLE%';
 TABLE_NAME            COMMENTS
 --------------------  ----------------------------------------
 ALL_ALL_TABLES        Description of all object and relational
                       tables accessible to the user
 ALL_NESTED_TABLES     Description of nested tables in tables
                       accessible to the user
 ALL_OBJECT_TABLES     Description of all object tables
                       accessible to the user
 ALL_PART_TABLES
 ALL_TABLES            Description of relational tables
                       accessible to the user
 ALL_UPDATABLE_COLUMNS Descriptionofallupdatablecolumns
 DBA_ALL_TABLES        Description of all object and relational
                       tables in the database
```

```
DBA_NESTED_TABLES      Description of nested tables contained
                       in all tables
DBA_OBJECT_TABLES      Description of all object tables in the
                       database
...
```

Xây dựng dictionary views

Sau khi tạo database, ta truy cập vào database theo user: `SYS` và chạy các scripts: *catalog.sql* và catprog.sql để tạo các dictionary views. Thông thường, các scripts này nằm trong thư mục: `%ORACLE_HOME%\RDBMS80\ADMIN`

Catalog.sql

`CATALOG.SQL` script dùng để tạo các view dựa trên các base tables (bảng cơ sở) của database. Các view này sẽ được tạo synonym (một tên khác với tên của objects được dùng để truy cập objects) tương ứng để dễ dàng truy vấn các dữ liệu từ đó hơn. Scripts này còn gọi tới các scripts khác để tạo các views và các đối tượng khác phục vụ cho các tiện ích Server Manager, cho việc kiểm tra, cho các tiện ích Export và Import dữ liệu,... Scripts `STANDARD.SQL` được gọi đến trong đó để tạo các môi trường PL/SQL tuân theo chuẩn.

Ví dụ: Scripts tạo mẫu giao tiếp cho 01 hàm built-in có tên `BITAND`:

```
function BITAND (LEFT binary_integer, RIGHT binary_integer)
return binary_integer;
```

Catproc.sql

`CATPROC.SQL` script dùng để tạo các hàm PL/SQL, các packages PL/SQL sử dụng trong `RDBMS`. Ngoài ra, `CATPROC.SQL` script còn tạo Các views mở rộng khác.

6.1.3. Sripts quản trị

Các scripts quản trị được đặt trong thư mục: `%ORACLE_HOME%\RDBMS80\ADMIN`

Các scripts này được phân nhóm và đặt trong từng file riêng biệt.

Các quy định về tên có trong Script quản trị

Quy ước	Diễn giải
Cat*.sql	Các thông tin Catalog và từ điển dữ liệu
Dbms*.sql	Phần khai báo (specification) của các packages trong database
Prvt*.plb	Phần thân cua packages đã được mã hoá và đóng gói
Utl*.sql	Các views và table tiện ích trong database

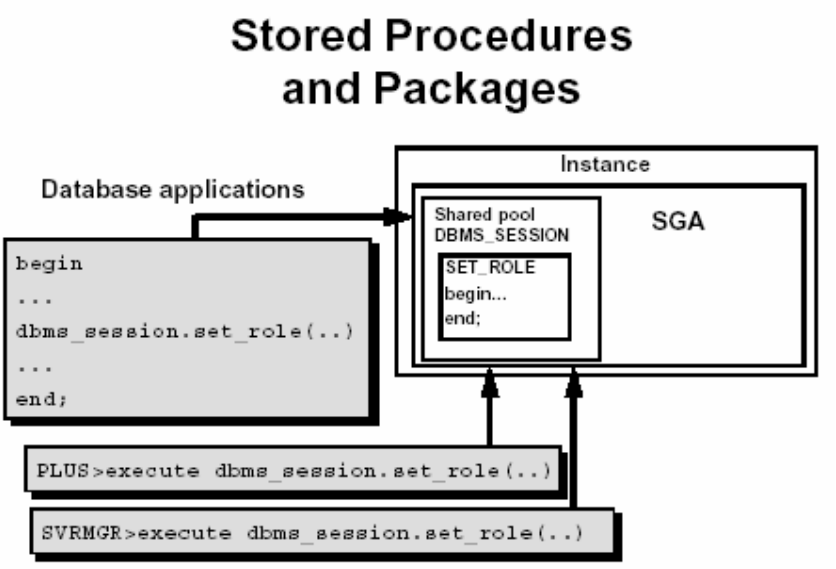
6.2.STORED PROCEDURES VÀ CÁC PACKAGES CHUẨN

6.2.1. Giới thiệu chung

Stored procedures và các packages là các đối tượng trong database, đó là tập hợp các đoạn mã lệnh PL/SQL để thực hiện một chức năng nào đó.

Stored procedures bao gồm cả các procedures (thủ tục), functions (hàm) và các packages được viết gộp thành một program unit (đơn vị chương trình).

Stored procedures có thể được tạo và huỷ bởi các lệnh `CREATE` và `DROP`



Hình vẽ 22. Stored procedures và các Packages chuẩn

Lợi ích của Stored procedures

- Các Stored procedures được nạp vào shared pool, do đó có thể giảm bớt việc truy xuất đĩa khi thực hiện thủ tục.
- Đảm bảo an toàn cho dữ liệu, ngăn không cho các users truy cập trực tiếp vào dữ liệu mà phải thông qua các thủ tục và hàm giao tiếp đã được cung cấp.
- Cho phép nhiều users có thể cùng sử dụng các bản sao của Stored procedures để thực hiện.

6.2.2. Stored procedures

Stored procedures là các functions hay procedures được tạo lập và lưu ngay trong dictionary giống như một schema object. Đây là tập hợp các câu lệnh SQL và PL/SQL. Sau khi Stored procedures được biên dịch, nó sẽ được gán tên và có thể thực hiện trực tiếp mà không cần phải biên dịch lại thêm bất cứ một lần nào nữa.

Sử dụng Stored procedures, ta có thể nạp trực tiếp vào ngay biểu thức thuộc câu lệnh SQL giống như là các hàm built-in có sẵn của Oracle như `UPPER` hay `SUBSTR`.

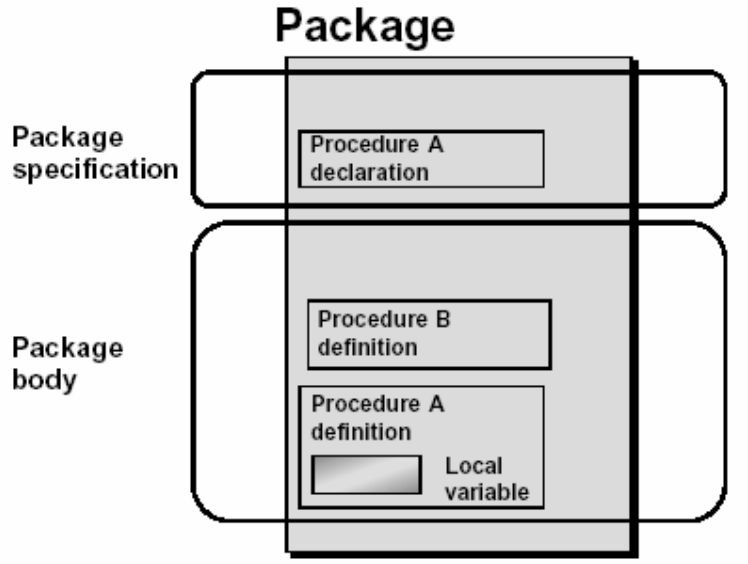
Các functions và procedures cho phép sử dụng tham số dưới dạng tham số vào (`IN`) và tham số ra (`OUT`) hoặc cũng có thể sử dụng tham số vừa vào vừa ra (`IN OUT`). Theo mặc đinh, các tham số được xác định ở chế độ vào `IN`.

6.2.3. Packages chuẩn

Một packages thông thường gồm hai phần: specification (phần đặc tả hay còn gọi là phần khai báo) và body (phần thân). Chúng được lưu riêng biệt trong cùng một database.

- Phần specification là phần giao tiếp với các ứng dụng. Phần này chứa các lời khai báo, các kiểu, biến, hằng, exceptions, cursors, và các khai báo hàm để sử dụng.
- Phần body là phần cài đặt cụ thể (implementation) của các khai báo trong phần specification.

Chức năng của packages cũng tương tự như Stored procedures. Một khi packages được biên dịch, packages đó có thể được sử dụng bởi nhiều ứng dụng khác nhau. Tuy nhiên, có một lợi ích lớn nhất khi sử dụng packages là ngay lần đầu tiên gọi đến packages, toàn bộ packages sẽ được nạp vào trong bộ nhớ.



Hình vẽ 23. Packages trong cơ sở dữ liệu

6.2.4. Giới thiệu một số packages chuẩn do Oracle cung cấp

Oracle cung cấp một số packages chuẩn, ngay sau khi tạo database:

- `DBMS_LOB`: cung cấp các thủ tục cho phép làm việc trên kiểu dữ liệu `BLOB` và `CLOB`, được định nghĩa trong file script catprog.sql.
- `DBMS_SESION`: cung cấp các câu lệnh SQL liên quan đến session như `ALTER SESSION`, `SET ROLE`, ... packages này được định nghĩa trong file *dbmsutil.sql* và *prvtutil.sql*
- `DBMS_UTILITY`: chứa các thủ tục tiện ích, được đặt trong file *dbmsutil.sql* và *prvtutil.sql*
- `DBMS_SPACE`: cung cấp các thông tin về khoảng trống của segment.
- `DBMS_ROWID`: cung cấp các thông tin về `ROWID`
- `DBMS_SHARE_POOL`: lưu trữ và huỷ bỏ các thông tin có trong share pool.

Packages	Thủ tục trong packages	Diễn giải
`DBMS_SESSION`	`SET_ROLE`	Kích hoạt việc thực hiện Roles của user

	SET_SQL_TRACE	Thiết lập chế độ dò tìm thực hiện lệnh
	SET_NLS	Chọn chuẩn hỗ trợ ngôn ngữ
	CLOSE_DATABASE_LINK	Đóng database link.
	UNIQUE_SESSION_ID	Trả về mã duy nhất cả các session hiên đang connect tới database.
	IS_ROLE_ENABLED	Xác đinh xem role có được kích hoạt trong session không.
	IS_SESSION_ALIVE	Xác định xem session có còn hay không.
	SET_CLOSE_CACHED_OPEN_CURSORS	Bật hoặc tắt close_cached_open_cursors
	FREE_UNUSED_USER_MEMORY	Giải phóng vùng bộ nhớ không còn sử dụng
DBMS_UTILITY	ANALYZE_SCHEMA	Phân tích các objects trong schema như: functions, procedures, packages, triggers,..
	COMPILE_SCHEMA	Biên dịch các objects trong schema
	DB_VERSION	Xác định phiên bản của database
DBMS_ROWID	ROWID_INFO	Thông tin về dòng dữ liệu
DBMS_SPACE	UNUSED_SPACE	Vùng không gian không sử dụng
	FREE_BLOCKS	Các blocks rỗi
DBMS_SHARED_POOL	KEEP	Lưu trữ các object trong shared pool
	UNKEEP	Thôi lưu giữ các object
	SIZES	Kích thước bộ nhớ trong shared pool
DBMS_SQL	OPEN_CURSOR	Trả về số hiệu cursor (ID number)
	PARSE	Phân tích câu lệnh
	BIND_VARIABLE	Binds một giá trị biến.
	BIND_ARRAY	Binds một giá trị biến mảng.
	EXECUTE Function	Executes a given cursor.
	EXECUTE_AND_FETCH	Thực hiện lệnh và lấy về các dòng dữ liệu.
	FETCH_ROWS	Lấy về các dòng dữ liệu của một cursor.
	COLUMN_VALUE	Lấy về dữ liệu của cột
	IS_OPEN	Xác định Cursor đã mở hay chưa.

	CLOSE_CURSOR	Đóng cursor và giải phóng bộ nhớ.
	LAST_ERROR_POSITION	Trả về lỗi thực hiện câu lệnh SQL
	LAST_ROW_COUNT	Trả về số lượng dòng dữ liệu lấy về
	LAST_ROW_ID	Trả về mã dòng dữ liệu xử lý ROWID
	LAST_SQL_FUNCTION_CODE	Trả về mã hàm SQL

6.2.5. Package DBMS_METADATA

Một PL/SQL package mới, DBMS_METADATA, được đưa vào Oracle 9i cho phép ta lấy được các siêu dữ liệu (metadata) – Các thông tin tổng hợp về các schema object.

- DBMS_METADATA là package mới bổ sung, nó cho phép thực hiện các thao tác DDL trên objects trong database.
- Package này làm việc được với các tables, indexes, views, packages, functions, procedures, triggers, synonyms, và types.
- DBMS_METADATA có các hàm cơ bản:
  - DBMS_METADATA.GET_DDL(object_type, name, schema)
  - DBMS_METADATA.GET_XML(object_type, name, schema)

Ví dụ:

```
SELECT DBMS_METADATA.GET_DDL('TABLE', 'EMP', 'SCOTT') from dual;
  CREATE TABLE "SCOTT"."EMP"
   (    "EMPNO" NUMBER(4,0),
        "ENAME" VARCHAR2(10),
        "JOB" VARCHAR2(9),
        "MGR" NUMBER(4,0),
        "HIREDATE" DATE,
        "SAL" NUMBER(7,2),
        "COMM" NUMBER(7,2),
        "DEPTNO" NUMBER(2,0),
         CONSTRAINT "PK_EMP" PRIMARY KEY ("EMPNO")
  USING INDEX PCTFREE 10 INITRANS 2 MAXTRANS 255
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645 PCTINCREASE 0
  FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE
"USERS"  ENABLE,
         CONSTRAINT "FK_DEPTNO" FOREIGN KEY ("DEPTNO")
  REFERENCES "SCOTT"."DEPT" ("DEPTNO") ENABLE NOVALIDATE
   ) PCTFREE 10 PCTUSED 40 INITRANS 1 MAXTRANS 255 LOGGING
  STORAGE(INITIAL 65536 NEXT 1048576 MINEXTENTS 1 MAXEXTENTS
2147483645 PCTINCREASE 0
  FREELISTS 1 FREELIST GROUPS 1 BUFFER_POOL DEFAULT) TABLESPACE
"USERS"
```

```
SELECT DBMS_METADATA.GET_XML('TABLE', 'EMP', 'SCOTT') from dual;
<?xml version="1.0"?>
<ROWSET>
 <ROW>
  <TABLE_T>
   <VERS_MAJOR>1</VERS_MAJOR>
   <VERS_MINOR>0</VERS_MINOR>
   <OBJ_NUM>5543</OBJ_NUM>
   <SCHEMA_OBJ>
    <OBJ_NUM>5543</OBJ_NUM>
    <DATAOBJ_NUM>5543</DATAOBJ_NUM>
    <OWNER_NUM>25</OWNER_NUM>
    <OWNER_NAME>SCOTT</OWNER_NAME>
    <NAME>EMP</NAME>
    <NAMESPACE>1</NAMESPACE>
    <MINEXTS>1</MINEXTS>
    <MAXEXTS>2147483645</MAXEXTS>
    <EXTSIZE>128</EXTSIZE>
    <EXTPCT>0</EXTPCT>
    ...
```

6.2.6. Package dbms_redefinition

Package này cung cấp 05 thủ tục cho phép chỉnh sửa các objects online .

- CAN_REDEF_TABLE
- START_REDEF_TABLE
- FINISH_REDEF_TABLE
- ABORT_REDEF_TABLE
- SYNC_INTERIM_TABLE

6.3.THÔNG TIN VỀ CÁC STORED PROCEDURES

Khi lưu trữ các Stored procedures hay packages, Oracle sẽ tự động lưu lại trạng thái của nó là VALID hay INVALID.

- VALID: Stored procedures hay packages có trạng thái là VALID nếu nó đã được biên dịch và không có lỗi xảy ra. Khi này, nó sẵn sàng cho việc sử dụng.
- INVALID: là trạng thái ngược lại với trạng thái VALID. Stored procedures hay Packages vẫn còn lỗi khi biên dịch. Khi này, ta chưa thể sử dụng được ngay.

Cú pháp lệnh yêu cầu biên dịch lại Stored procedures:

```
ALTER PROCEDURE [schema_name].<procedure_name> COMPILE [DEBUG];
```

Với:

```
schema_name     tên schema chứa procedure cần biên dịch lại
```

procedure_name	tên của procedure biên dịch lại.
COMPILE	chỉ định yêu cầu biên dịch lại procedure
DEBUG	chỉ định chương trình biên dịch mã lệnh PL/SQl của procedure sẽ sinh mã lệnh phù hợp để chương trình PL/SQL debugger có thể đọc. User có thể sử dụng chương trình này để dò tìm và gỡ lỗi cho procedure.

Ví dụ:

```
ALTER PROCEDURE henry.close_acct  COMPILE;
```

Tương tự như đối với procedure, cú pháp lệnh yêu cầu biên dịch lại Stored function có dạng:

```
ALTER FUNCTION [schema_name].<function_name> COMPILE [DEBUG];
```

Ví dụ:

```
ALTER FUNCTION merriweather.get_bal COMPILE;
```

Đối với package, lệnh yêu cầu biên dịch lại cũng tương tự nhưng có thêm một bổ sung là user phải khai báo rõ từng phần của package sẽ được biên dịch lại.

Cú pháp:

```
ALTER PACKAGE [schema_name].<package_name>
     COMPILE [DEBUG] <PACKAGE | SPECIFICATION | BODY>;
```

Các khai báo bổ sung cho phép user yêu cầu biên dịch lại phần `SPECIFICATION` hay phần `BODY` hoặc là biên dịch lại cả hai phần trên.

Ví dụ:

```
ALTER PACKAGE blair.accounting
        COMPILE PACKAGE;

Hoặc:
ALTER PACKAGE blair.accounting
        COMPILE BODY;
```

Để xác định được trạng thái của các Stored procedures, ta có thể thực hiện truy vấn dựa trên dictionary `DBA_OBJECTS`.

```
SVRMGR> SELECT object_name, object_type, status
     2> FROM dba_objects WHERE object_name like 'DBMS_%'
OBJECT_NAME          OBJECT_TYPE    STATUS
-------------------- -------------- --------
DBMS_ALERT           PACKAGES       VALID
DBMS_ALERT           PACKAGES       BODY VALID
DBMS_ALERT_INFO      TABLE          VALID
DBMS_APPLICATION_INF PACKAGES       VALID
DBMS_APPLICATION_INF PACKAGES       BODY VALID
DBMS_AQ              PACKAGES       VALID
DBMS_AQ              PACKAGES       BODY VALID
...
```

Hoặc ta cũng có thể sử dụng lệnh `DESCRIBE` để lấy thông tin

```
SVRMGR> DESCRIBE dbms_session.set_role
procedure SET_ROLE (ROLE_CMD VARCHAR2);
```

```
svrmgr> describe dbms_session
packages dbms_session is
------------
-- OVERVIEW
-- This packages provides access to SQL "alter session"
-- statements, and other session information from, stored
-- procedures.
----------------------------
-- PROCEDURES AND FUNCTIONS
procedure set_role(role_cmd varchar2);
-- Equivalent to SQL "SET ROLE ...".
-- Input arguments:
-- role_cmd
-- This text is appended to "set role " and then executed as
-- SQL.
procedure set_sql_trace(sql_trace boolean);
-- Equivalent to SQL "ALTER SESSION SET SQL_TRACE ..."
-- Input arguments:
-- sql_trace
-- TRUE or FALSE. Turns tracing on or off.
procedure set_nls(param varchar2, value varchar2);
```

Stored procedures hay Packages nhận trạng thái `INVALID` khi các câu lệnh trong Stored procedures hay Packages bị lỗi.

Chương 7. QUẢN TRỊ CONTROL FILES

7.1.CONTROL FILES

7.1.1. Giới thiệu control file

Control file là file thông tin dạng nhị phân được sử dụng cho việc khởi tạo và vận hành database một cách hiệu quả.

Mỗi khi instance được MOUNT (gắn) với một Oracle database, các thông tin trong control file sẽ được đọc ra, từ đó xác định các data files và các online redo log files.

Control file được cập nhật liên tục vào database trong suốt quá trình sử dụng và nó luôn ở trạng thái sẵn sàng (available) mỗi khi database được OPEN (mở) hay được MOUNT (gắn) với instance.

Control file cung cấp các thông tin một cách đồng nhất trong database được sử dụng trong quá trình khôi phục (recovery).

Mỗi control file tại một thời điểm chỉ phục vụ cho một database. Khi đã có một database sử dụng control file thì các database khác sẽ không thể truy cập tới control file đó nữa.

7.1.2. Cách thức đặt tên control file

Tên control file được xác định trong tham số CONTROL_FILES của parameter file. Tên của các control files được đặt phân cách bởi dấu phẩy (,). Instance phục vụ database sẽ mở các control file và lấy các thông tin từ đó để có thể điều khiển hoạt động của database. Trong quá trình hoạt động, Instance cũng sẽ ghi lại các tình trạng của database.

Để đảm bảo an toàn, một database cần ít nhất 02 control files và được đặt tại hai chỗ khác nhau. Các control files nên được đặt tên khác nhau sao cho có thể phân biệt dễ dàng.

Tên của Control files nên được đặt kèm với tên của database cho dễ nhớ, như sau:

```
CTL<n><database_name>.ORA
```

Với:

```
n               là số thứ tự của control file
database_name   tên của database
```

Trong parameter file, các tên của control files được đặt phân cách nhau bởi các dấu phẩy.

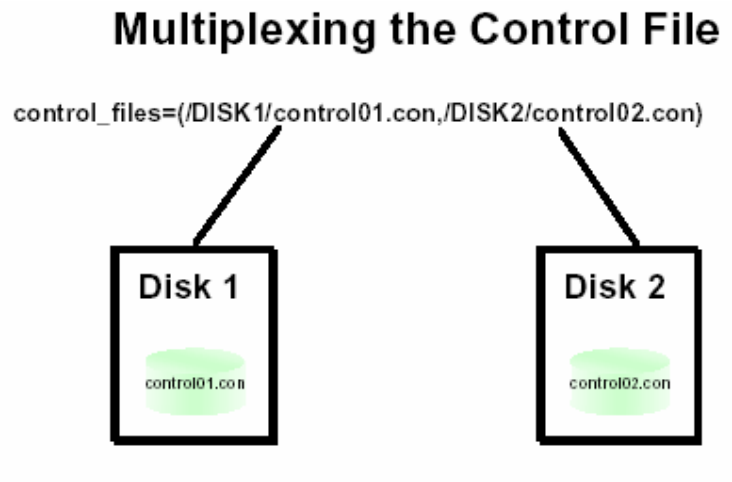
Ví dụ:

```
control_files =  ("C:\ORANT\DATABASE\CTL1KTKB.ORA",
                      "C:\ORANT\DATABASE\CTL2KTKB.ORA")
```

7.1.3. Kết hợp nhiều control files

Khi tạo database, ta có thể sử dụng cùng lúc nhiều control files thông qua việc chỉ rõ tên các control files trong tham số khởi tạo CONTROL_FILES. Oracle server tạo và cập nhật tất cả danh sách các file liên quan mỗi khi tạo database.

Oracle khuyến cáo sử dụng ít nhất 02 control files. Các control files nên được đặt riêng biệt trên các ổ đĩa khác nhau đề phòng sự cố. Nếu một control file bị hỏng, ta có thể sao chép lại file này rồi khởi động lại instance.



Hình vẽ 24. Kết hợp sử dụng nhiều control file

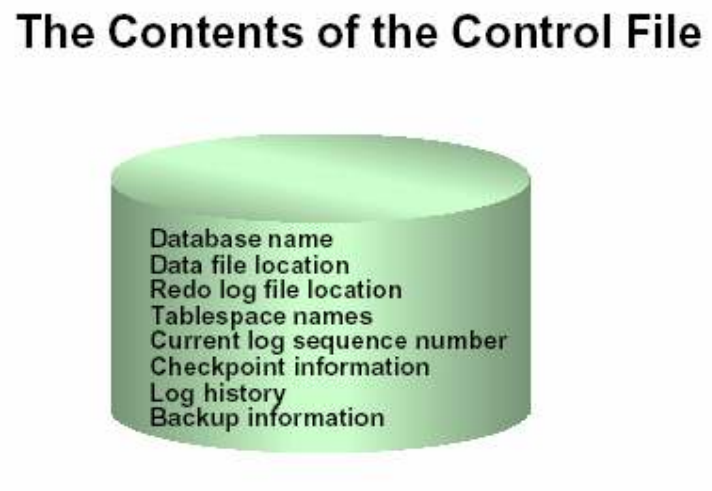
Để thêm mới một control file hoặc thay đổi số lượng cũng như nơi đặt các control file, ta thực hiện theo các bước sau:

1. Shutdown database.
2. Sử dụng lệnh của hệ điều hành để sao chép thêm một bản sao của control file và nên lưu trữ trên một thiết bị khác.
3. Sửa đổi hoặc thêm mới tham số `CONTROL_FILES` và tên (có đường dẫn) tương ứng với các control files.
4. Khởi động lại database.

7.1.4. Nội dung của control file

Các thông tin chứa trong control file bao gồm:

- Tên database và các định danh (identifications)
- Tên và nơi chứa các data files, các redo log files
- Tên các tablespaces trong database
- Nhãn thời gian tương ứng lúc tạo database
- Giá trị số hiệu của log sequence hiện thời
- Thông tin về checkpoint
- Các thông tin lịch sử (log history)
- Các thông tin sao lưu của tiện ích Recovery Manager



Hình vẽ 25. Nội dung control file

Control file có thể được chia làm hai loại chính:

- Có thể tái sử dụng (reused)
- Không thể tái sử dụng (unreused)

7.1.5. Các tham số ảnh hưởng tới kích thước của control file

Có một số tham số hệ thống liên quan tới kích thước của control file

- MAXLOGFILES
- MAXLOGMEMBERS
- MAXLOGHISTORY
- MAXDATAFILES
- MAXINSTANCES

Các control files được xác định tự động dựa theo các tham số khởi tạo tại thời điểm tạo lập database:

```
CONTROL_FILES = ("C:\ORANT\DATABASE\CTL1KTKB.ORA",
                 "C:\ORANT\DATABASE\CTL2KTKB.ORA")
```

Tên file kèm theo đường dẫn được đặt luôn trong tham số tạo database.

Các tham số được chỉ ra trong database có ảnh hưởng tới control file. Quản trị viên database có thể tạo lại các control file hay thay đổi các tham số trong database để có thể tăng, giảm kích thước của control file.

Việc tạo mới control file đòi hỏi phải thay đổi kích thước của control file. Control file lưu trữ các thông tin cần thiết cho Recovery Manager. Vì thế, khi sử dụng Recovery Manager những phần không tái sử dụng được trong control file có thể được mở rộng dựa theo số lượng các thành phần.

7.2.QUẢN TRỊ CONTROL FILE

7.2.1. Tạo mới control file

Việc tạo mới control files đối với database đôi khi là cần thiết. Ta hãy xét các tình huống:

- Tất cả các control files của database hiện thời đều bị lỗi và ta không có bản backup của chúng.

- Ta muốn thay đổi một hay nhiều tham số được thiết lập đối với database mà các tham số này được chỉ ra ngay từ câu lệnh CREATE DATABASE như tên *database*, MAXLOGFILES, MAXLOGMEMBERS, MAXLOGHISTORY, MAXDATAFILES, và MAXINSTANCES.

Ví dụ, ta muốn đổi tên database để khỏi xảy ra xung đột với một database đang có trong hệ thống nhưng trùng tên.

Ta có thể tạo mới control file cho một database thông qua câu lệnh SQL.

Cú pháp:

```
CREATE CONTROLFILE [REUSE]
    [SET] DATABASE database
    LOGFILE [GROUP integer] filespec [, [GROUP integer]
           filespec] ...
    {RESETLOGS | NORESETLOGS}
    DATAFILE filespec [, filespec] ...
    [MAXLOGFILES integer]

    [MAXLOGMEMBERS integer]
    [MAXLOGHISTORY integer]
    [MAXDATAFILES integer]
    [MAXINSTANCES integer]
    [ARCHIVELOG | NOARCHIVELOG]
```

Với:

REUSE	Cho biết CONTROL_FILES có thể được tái sử dụng, ta không cần quan tâm tới các tham số thuộc loại tuỳ chọn.
SET DATABASE	Thay đổi tên của database. Lưu ý: <Tên>
DATABASE	Tên của database.
LOGFILE	danh sách tên của các redo log file groups
MAXLOGFILES	Số lượng tối đa các redo log file groups
MAXLOGMEMBERS	Số lượng tối đa các members trong một redo
MAXLOGHISTORY	Số lượng tối đa các archived redo log file groups
MAXDATAFILES	Số lượng tối đa các datafiles
MAXINSTANCES	Số lượng tối đa các instances có thể kết nối tới database.
ARCHIVELOG	Thiết lập chế độ archiving lưu trữ các redo log files

Ví dụ:

```
CREATE CONTROLFILE
      SET DATABASE prod
      LOGFILE GROUP 1 ('logfile1A', 'logfile1B') SIZE 50K,
      GROUP 2 ('logfile2A', 'logfile2B') SIZE 50K
      NORESETLOGS
      DATAFILE 'datafile1' SIZE 3M, 'datafile2' SIZE 5M
      MAXLOGFILES 50
      MAXLOGMEMBERS 3
```

```
MAXDATAFILES 200
MAXINSTANCES 6
ARCHIVELOG;
```

7.2.2. Tạo mới control file cho một database đã có sẵn

Việc tạo mới control file được thực hiện theo các bước sau:

1. Thiết lập danh sách các datafiles và online redo log files sử dụng trong database. Trong trường hợp backup database, ta có thể dễ dàng xác định được danh sách các file này dựa vào thông tin trong dictionary view: `V$CONTROLFILE`, `V$DATAFILE`, `V$LOGFILE`. Trong trường hợp database bị lỗi, quản trị viên database cần cố gắng xác định đầy đủ các datafiles và online redo log files. Nếu thiếu bất kỳ một trong số các file trên thì tablespace SYSTEM sẽ không thể khôi phục lại được và do đó ta không thể khôi phục lại được database.
2. Shut down (tắt) database nếu nó đang được mở. Thực hiện shut down ở chế độ normal. Trong trường hợp không thể tắt normal được thì hãy tắt database theo chế độ `IMMEDIATE` hoặc `ABORT`.
3. Sao lưu (Backup) tất cả các datafiles và online redo log files của database.
4. Startup instance trở lại ở chế độ nomount.
5. Tạo mới control file thông qua lệnh tạo `CONTROL FILES`. Khi tạo mới control file, sử dụng tuỳ chọn `RESETLOGS` nếu database bị mất bất kỳ một nào online redo log groups. Trong trường hợp này ta cần khôi phục lại các redo logs bị mất. Ngược lại, ta sử dụng tuỳ chọn `NORESETLOGS`.
6. Sao lưu control file mới tạo.
7. Sửa đổi các tham số trong parameter file mà có sử dụng đến trong các control files bao gồm tham số `CONTROL_FILES` và `DB_NAME`.
8. Thực hiện khôi phục database nếu cần. Ta sẽ bỏ qua bước này trong trường hợp không cần phải khôi phục database. Nếu control file mới tạo có sử dụng tuỳ chọn `NORESETLOGS`, thì ta có thể khôi phục lại toàn bộ database. Trong trường hợp tuỳ chọn sử dụng là `RESETLOGS`, ta cần chỉ ra thêm một tuỳ chọn nữa là `USING BACKUP CONTROL FILE`. Thủ tục này sẽ thực hiện khôi phục lại các online hoặc archived redo logs hoặc datafiles.
9. Open database với control file vừa tạo. Nếu không thực hiện recovery thì có thể open database ở chế độ normally.
10. Nếu có sử dụng `RESETLOGS` trong lúc tạo control file, thì cần sử dụng thêm câu lệnh `ALTER DATABASE` , với tuỳ chọn `RESETLOGS`.

7.2.3. Một số lỗi đối với các Control Files

Sau khi thực hiện lệnh `CREATE CONTROLFILE`, ta có thể ta gặp một số lỗi cơ bản sau:

Thiếu file

Sau khi tạo một control file và sử dụng nó để mở database, kiểm tra alert log để biết liệu Oracle có xác định được có thông tin gì không đồng nhất giữa data dictionary và control file hay không? Ví dụ như datafile có kèm theo cả data dictionary nhưng không có danh sách các data dictionary đi kèm.

Nếu một datafile đã tồn tại trong data dictionary nhưng chưa có trong control file mới tạo, Oracle sẽ tạo một placeholder entry trong control file với tên là `MISSING`*nnnn* (trong đó *nnnn* là một con số viết dưới dạng thập phân).

Ta xét hai trường hợp có thế xảy ra như sau:

- Sử dụng tuỳ chọn RESETLOGS trong câu lệnh CREATE CONTROLFILE sẽ cho phép mở database mà không cần tới tuỳ chọn RESETLOGS. Điều này chỉ có thế xảy ra nếu tất cả các online redo logs đang trong tình trạng sẵn sàng.
- Sử dụng tuỳ chọn RESETLOGS trong câu lệnh CREATE CONTROLFILE để bắt buộc phải mở database cùng với tuỳ chọn RESETLOGS, datafile tương ứng với MISSING*nnnn* ở chế độ chỉ đọc hay OFFLINE.

Khi mở database có sử dụng tuỳ chọn RESETLOGS, và MISSING*nnnn* tương ứng với datafile không ở chế độ chỉ đọc hay offline, ta sẽ không thể truy xuất vào datafile đó. Trong trường hợp này, tablespace chứa datafile cần được huỷ bỏ (DROP).

Xử lý lỗi xảy ra đối với lệnh CREATE CONTROLFILE

Oracle gửi trả về mã lỗi(các mã lỗi hay xảy ra là ORA-01173, ORA-01176, ORA-01177, ORA-01215 hoặc ORA-01216) khi ta cố gắng thực hiện mount và open database sau khi tạo mới một control file. Tình huống hay xảy ra nhất là trong câu lệnh CREATE CONTROLFILE mà ta quên một file hoặc có đưa vào tên file nhưng nó vẫn chưa có trong danh sách. Trong trường hợp này, ta cần phải khôi phục (RESTORE) lại các files đã được backup ở bước 3 (phía trên) và lặp lại các thủ tục ở bước 4 (phía trên) lưu ý sử dụng đúng tên các files.

7.2.4. Huỷ bỏ Control Files

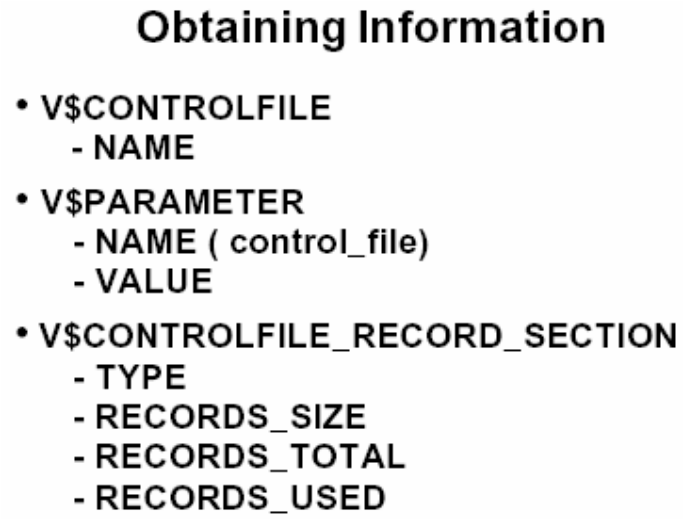
Ta có thể huỷ bỏ các control files khỏi database. Ví dụ, ta thực hiện việc này khi đường dẫn tới các control file không còn phù hợp nữa. Có một điều lưu ý là tại bất kỳ thời điểm nào database cũng cần phải có ít nhất là 2 control files.

Các bước thực hiện

1. Shut down (tắt) database.
2. Sửa lại tham số CONTROL_FILES trong parameter file, xoá tên control file cũ và thay vào đó tên control file mới.
3. Restart (khởi động lại) database.

7.3.THÔNG TIN TRẠNG THÁI CỦA CONTROL FILES

Ta có thể xem được các thông tin về control file dựa trên dictionary views có trong database.



Ví dụ:

```
SVRMGR> SELECT name
2>FROM v$controlfile;
NAME
----------------------
/DISK1/control01.con
/DISK2/control02.con
2 rows selected.

SVRMGR> SELECT value
2>FROM v$parameter WHERE name ='control_files';
VALUE
-----------------------
/DISK1/control01.con
/DISK2/control02.con
2 rows selected.
```

`V$CONTROLFILE_RECORD_SECTION` chứa các thông tin về các section.

Ví dụ:

```
SVRMGR>SELECT type, record_size, records_total, records_used
2> FROM v$controlfile_record_section
3> WHERE type='DATAFILE';
TYPE                RECORD_SIZ  RECORDS_TO RECORDS_US
------------ ----------         ---------- ----------
DATAFILE            180         30          4
1 row selected.
```

Cột dữ liệu `RECORDS_TO` chỉ ra số lượng các bản ghi được cấp phát cho một section.

Chương 8. QUẢN LÝ REDO LOG FILES

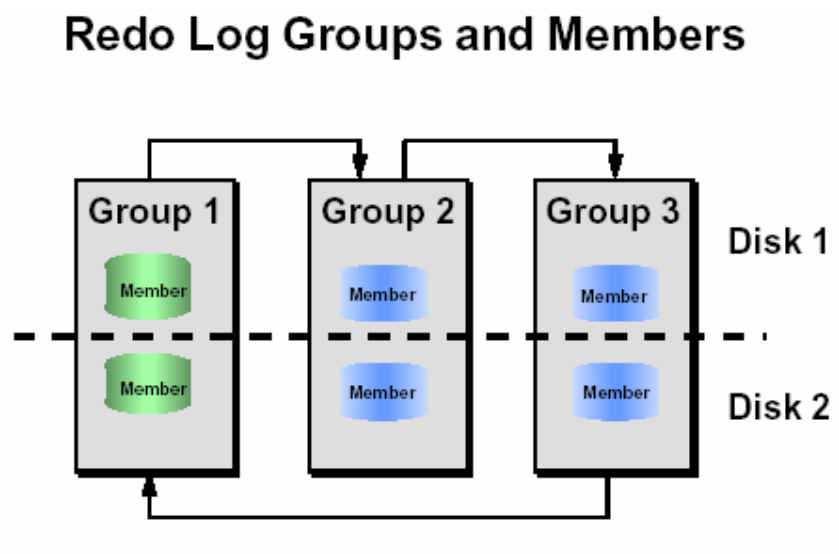
8.1.SỬ DỤNG CÁC REDO LOG FILES

8.1.1. Redo log file

Oracle server sử dụng các online redo log files để giảm thiểu việc mất mát dữ liệu trong database. Redo log files ghi lại tất cả các thay đổi trong database buffer cache trừ một vài ngoại lệ ghi dữ liệu trực tiếp.

Redo log files được sử dụng đến khi instance gặp sự cố và ta muốn khôi phục lại các dữ liệu đã commit nhưng chưa kịp ghi lên data files. Redo log files chỉ được sử dụng trong trường hợp khôi phục dữ liệu.

Quản trị viên cần thiết lập các bản sao các online redo log files của database để tránh việc mất mát thông tin trong database do việc sử dụng một file duy nhất.



Hình vẽ 26. Nhóm các redo log

8.1.2. Online Redo Log Groups

- Là nhóm các bản sao riêng biệt của các online redo log files được gọi là online redo log *group*.
- Background process `LGWR` thực hiện việc ghi đồng thời các thông tin tương tự nhau vào các member thuộc cùng một group. Khi một group đầy sẽ tiếp tục chuyển sang ghi dữ liệu trên group tiếp theo.
- Oracle server, thông thường, cần ít nhất 02 online redo log file groups để có thể vận hành một database.

8.1.3. Online Redo Log Members

- Mỗi một online redo log file trong một group được gọi là một *member (thành viên)*.
- Mỗi member trong một nhóm có một số thứ tự *(log sequence numbers)* phân biệt và các member này có cùng một kích thước. Số thứ tự được gán mỗi khi Oracle server bắt đầu ghi dữ liệu vào log group để có thể phân biệt được các redo log file duy nhất. Số log sequence number được lưu trữ trong control file và trong phần header của tất cả các data files.

8.1.4. Nội dung của Online Redo Log Files (Members)

Online redo log files lưu trữ các *redo records* hay còn được gọi là các *redo entries.* Mỗi *redo record* là một nhóm các *change vectors (vector thay đổi dữ liệu)*, trong đó mỗi vector đặc trưng cho một sự thay đổi trên một block dữ liệu thuộc database. Ví dụ, khi ta thay đổi giá trị lương trong bảng employee, Oracle sẽ tạo ra một redo record lưu trữ lại việc thay đổi dữ liệu của data segment block, rollback segment block và transaction table tương ứng với thay đổi dữ liệu nói trên.

Các redo entries lưu trữ lại các dữ liệu để từ đó ta có thể tái tạo lại các thay đổi dữ liệu trong database, bao gồm cả rollback segments. Khi thực hiện phục hồi (recover) database sử dụng redo data, Oracle sẽ đọc các change vectors có trong các redo records rồi áp các thay đổi này vào các blocks tương ứng.

Các redo records được lưu trữ trong bộ nhớ đệm SGA. Mỗi khi thực hiện commit một transaction, `LGWR` sẽ ghi lại các redo records của transaction đó từ các redo log buffer thuộc SGA vào một online redo log file, và gán một số hiệu *system change number* (`SCN`) cho transaction đã được commit đó. Chi khi các redo records thuộc transaction đã được lưu trữ an toàn trên đĩa thì user process mới được nhận thông báo: transaction has been committed.

Các redo records có thể được ghi vào online redo log file trước khi transaction tương ứng được commit. Khi redo log buffer đầy, hoặc khi transaction commit, `LGWR` sẽ đẩy tất cả các redo log entries trong redo log buffer ra online redo log file, ngay cả khi redo records có thể chưa được commit để khi cần, Oracle có thể khôi phục (roll back) lại các thay đổi này.

8.1.5. Active và Inactive Online Redo Log Files

Tại mỗi một thời điểm, Oracle chỉ sử dụng một trong số các online redo log files để lưu trữ các redo records có trong redo log buffer. Online redo log file đó ở trạng thái sẵn sàng cho việc ghi dữ liệu, nó được gọi là *current* online redo log file.

Các online redo log files cần thiết cho việc khôi phục instance được gọi là *active* online redo log files. Trái lại, các online redo log files không cần thiết cho việc khôi phục instance được gọi là *inactive*.

Khi quản trị viên database đặt chế độ enable archiving, Oracle sẽ không thể tái sử dụng hay ghi đè lên các active online log file cho tới khi ARC*n* lưu trữ hết các nội dung của nó. Trong trường hợp disable archiving, khi online redo log file cuối cùng được điền đầy, việc lưu ra file sẽ được tiếp tục thực hiện đối với active file đầu tiên.

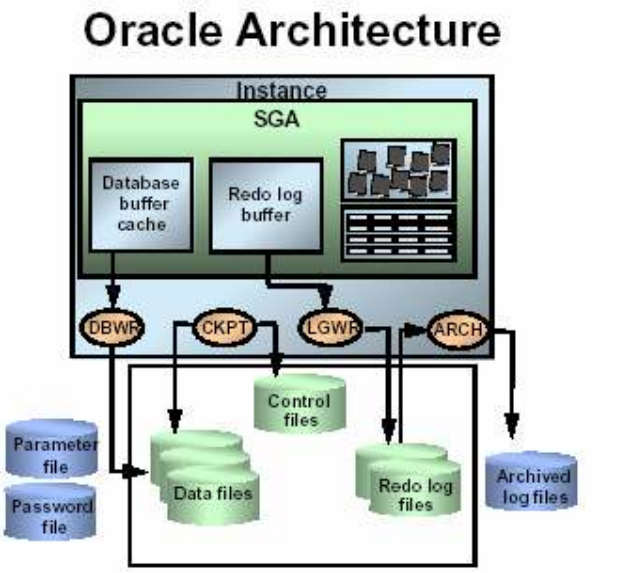
8.1.6. Thiết lập các Redo Log Files khởi tạo

Việc khởi tạo ban đầu tập hợp các online redo log file bao gồm các groups và các members được thực hiện trong quá trình tạo database.

Các tham số dưới đây xác định các giới hạn và số lượng của online redo log files:

- Tham số `MAXLOGFILES` trong lệnh `CREATE DATABASE` xác định số lượng tối đa các online redo log groups. Số lượng tối đa cho `MAXLOGFILES` là `255`.
- Tham số `MAXLOGMEMBERS` trong lệnh `CREATE DATABASE` quy định số lượng tối đa các members có trong mỗi group.
- Tham số khởi tạo `LOG_FILES` xác định số lượng tối đa các log groups có thể được mở trong database tại thời điểm hiện thời. Giá trị này không được vượt quá giá trị `MAXLOGFILES*MAXLOGMEMBERS`.

8.2.LGWR, LOG SWITCHES VÀ CHECKPOINTS



Hình vẽ 27. Tổ chức các redo log files

8.2.1. Redo Log Buffer và Background process LGWR

Oracle Server sẽ tuần tự ghi lại các thay đổi đối với database có trong redo log buffer. Redo *log buffer* được sử dụng theo kiểu xoay vòng. Theo đó, các redo entries sẽ được tiên trình nền `LGWR` ghi vào một trong các online redo log groups gọi là online redo log group hiện thời (current) theo các tình huống sau:

- Khi commit một transaction
- Khi redo log buffer đã đầy
- Khi `LGWR` vượt quá thời gian timeout (3 giây)
- Trước khi `DBWR` ghi các blocks bị thay đổi trong database buffers cache vào trong các data files

Các members trong một redo log group được tiến trình `LGWR` ghi lên đó với cùng một nội dung dữ liệu. Cho nên không có khác biệt giữa các members trong một log group mà chỉ có sự khác nhau giữa các members ở các log group khác nhau.

8.2.2. Log Switches

`LGWR` ghi dữ liệu lên các online redo log files một cách tuần tự, tức là mỗi khi online redo log group được ghi đầy, `LGWR` sẽ lại chuyển sang ghi lên group tiếp theo. Khi online redo log file cuối cùng được ghi đầy, `LGWR` sẽ lại quay trở về online redo log group đầu tiên và lại bắt đầu quá trình ghi.

Log switch là sự kiện xảy ra khi `LGWR` dừng việc ghi trên một online redo log group và chuyển sang ghi trên online redo log group khác. Quản trị viên database cũng có thể thực hiện các log switches bằng tay. Mỗi khi xảy ra log switch, `LGWT` sẽ ghi dữ liệu lên log group mới và nó gán một số hiệu duy nhất để xác định được các redo entries vừa lưu giữ.

Mỗi khi xảy ra sự kiện log switch đồng thời một sự kiện checkpoint cũng sẽ được khởi tạo.

8.2.3. Checkpoints

Khi có checkpoints thì:

- Tất cả các dữ liệu trong database buffers đã bị thay đổi, tính cho đến thời điểm xảy ra checkpoint, sẽ được Background process DBWR ghi lên datafiles.
- Background process CKPT cập nhật phần headers của các data files và các control files.

Checkpoints có thể xảy ra đối với tất cả các data files trong database hoặc cũng có thể xảy ra với một data files cụ thể.

Checkpoint xảy ra theo các tình huống sau:

- Mỗi khi có log switch
- Khi một shut down một instance với các chế độ trừ chế độ abort
- Xảy ra theo như thời gian quy định trong các tham số khởi tạo LOG_CHECKPOINT_INTERVAL và LOG_CHECKPOINT_TIMEOUT
- Khi có yêu cầu trực tiếp của quản trị viên

Thông tin về checkpoint được lưu trữ trong Alert file trong trường hợp các tham số khởi tạo LOG_CHECKPOINTS_TO_ALERT được đặt là TRUE. Và ngược lại với giá trị FALSE.

8.3.LÊN KẾ HOẠCH SỬ DỤNG REDO LOG FILES

8.3.1. Xác định số lượng Online redo log files

Để xác định số lượng các online redo log files sử dụng cho phù hợp với database ta cần phải kiểm tra với nhiều cầu hình khác nhau.

Trong một số trường hợp, một database instance chỉ cần tới 02 groups. Tuy nhiên, trong một số trường hợp khác, một database instance lại có thể cần tới nhiều groups hơn để có thể luôn đảm bảo có các groups sẵn dùng cho LGWR. Ví dụ, khi các thông điệp ghi trong trace file hay Alert file cho biết LGWR thường xuyên phải chờ một group do vẫn chưa kết thúc được checkpoint, hoặc do group vẫn chưa được lưu trữ (archived) thì lúc này là lúc ta cần thêm mới các groups.

Mặc dù Oracle server cho phép sử dụng nhiều groups với số lượng members trong nó là khác nhau, ta vẫn nên cố gắng xây dựng một cấu hình cân đối (số lượng các members trong các group nên là bằng nhau).

8.3.2. Nơi đặt các Online Redo Log Files

Khi sử dụng đồng thời nhiều online redo log files, ta nên đặt các members của một group trên các phần đĩa khác nhau. Một điều lưu ý là khi một member nào đó không sẵn dùng (available) mà các members khác là sẵn dùng thì instance cũng không thể shut down được.

Việc tách biệt các archive log files và online redo log files trên các phần đĩa khác nhau, có thể làm giảm bớt xung đột giữa các background process ARCH và LGWR.

Các data files và online redo log files nên đặt trên các phần đĩa khác nhau để giảm bớt xung đột giữa LGWR và DBWR hạn chế việc mất dữ liệu ở cả data files và online redo log files trong trường hợp hỏng ổ đĩa.

8.3.3. Xác định kích thước cho các Online Redo Log Files

Kích thước tối thiểu của một online redo log file là 50 K còn kích thước tối đa thì tuỳ thuộc vào hệ điều hành. Các members thuộc các groups khác nhau có thể có các kích thước khác nhau; Tuy nhiên ta nên đặt kích thước giống nhau giữa các members này.

Việc sử dụng các groups có kích thước khác nhau chỉ nên thực hiện một cách tạm thời khi ta muốn thay đổi kích thước của các members. Trong trường hợp này, ta cần tạo các online redo log groups mới với kích thước khác, rồi sau đó loại bỏ (remove) các groups cũ đi.

Một số tình huống ảnh hưởng tới cấu hình của các online redo log files:

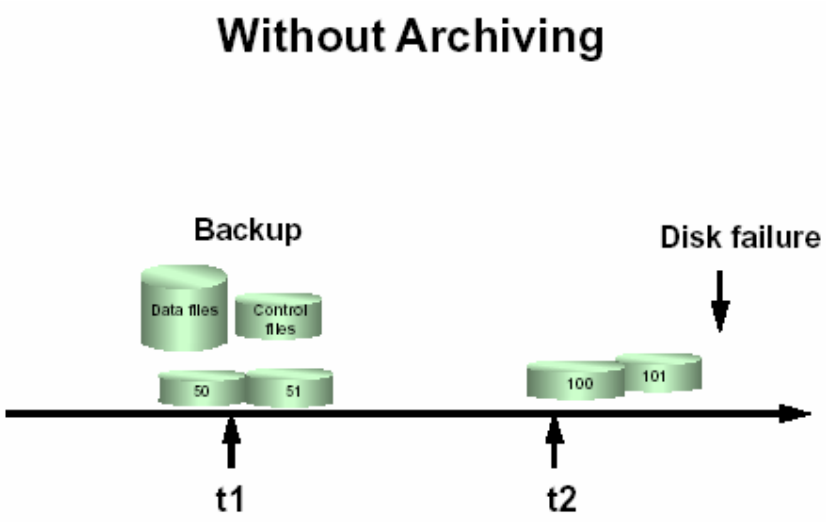
- Số lượng các log switches và checkpoints
- Số lượng và độ lớn của các redo entries
- Độ lớn của vùng không gian lưu trữ thứ cấp

8.3.4. Lưu trữ các redo log files

Quản trị viên database cần phải quyết định đặt chế độ `ARCHIVELOG` hay chế độ `NOARCHIVELOG` cho database.

Chế độ NOARCHIVELOG

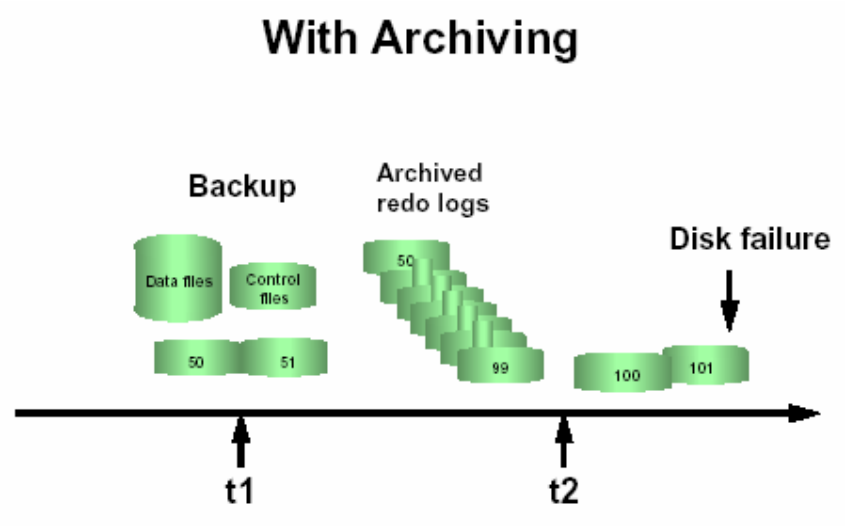
Với chế độ `NOARCHIVELOG`, các online redo log files sẽ bị ghi đè mỗi khi online redo log file đã ghi đầy và xảy ra log switches. `LGWR` sẽ không ghi đè lên redo log group cho tới khi kết thúc checkpoint của group đó



Hình vẽ 28. Lưu trữ dữ liệu ở chế độ NOARCHIVING

Chế độ ARCHIVELOG

Trong trường hợp database được thiết lập ở chế độ `ARCHIVELOG`, các groups đã đầy, mặc dù ở trạng thái inactive sẽ vẫn được lưu giữ. Do tất cả các thay đổi trong database đều được ghi lại trong các online redo log files, quản trị viên database có thể sử dụng phương pháp sao chép vật lý (physical backup) và có thể khôi phục lại các dữ liệu đã commit trong database mà không sợ bị mất dữ liệu.



Hình vẽ 29. Lưu trữ dữ liệu ở chế độ ARCHIVING

Có hai hình thức lưu trữ các online redo log files:

- Thực hiện lưu trữ bằng tay (manually). Lưu trữ các redo log file đã đầy theo lệnh của quản trị viên database.
- Lưu trữ tự động (automatically). Lưu trữ các redo log file đã đầy mỗi khi xảy ra log switch.

Tham số `LOG_ARCHIVE_START` trong parameter file xác định các chế độ lưu trữ này.

- `LOG_ARCHIVE_START = TRUE`, thực hiện lưu trữ ở chế độ tự động
- `LOG_ARCHIVE_START = FALSE`, thực hiện lưu trữ ở chế độ manually

8.4.ĐIỀU KHIỂN LƯU TRỮ SAU ĐỐI VỚI PRIMARY/STANDBY

Oracle cung cấp cơ chế điều khiển switch các online redo log group dựa theo thời gian (`time-based`). Trong cấu hình `primary/standby`, tất cả các noncurrent logs tại primary site sẽ được lưu trữ rồi vận chuyển tới standby database. Việc này sẽ hiệu quả khi hạn chế số lượng các redo records.

Việc thực hiện lưu trữ sau là vì standby database cho tất cả các thay đổi trên online redo log tại primary database được lưu trữ sau. Để điều khiển việc lưu trữ sau này, ta cần sử dụng tham số `ARCHIVE_LAG_TARGET`. Việc thiết lập tham số này cho phép ta hạn chế, cũng như xác định được khoảng thời gian được sử dụng cho lưu trữ sau.

8.4.1. Thiết lập tham số ARCHIVE_LAG_TARGET

Khi thiết lập tham số khởi tạo `ARCHIVE_LAG_TARGET`, Oracle sẽ kiểm tra theo định kỳ thời gian các online redo log của instance hiện thời và phát sinh các log switch theo các điều kiện sau:

- Giả sử ban đầu, current log được tạo sau *n* giây và sau đó lại mất *m* giây để lưu current log ra đĩa. Khi này khoảng thời gian `n + m` sẽ tương ứng với giá trị của tham số `ARCHIVE_LAG_TARGET`.
- Current log chứa các redo records.

Tham số ARCHIVE_LAG_TARGET cho biết giới hạn trên về thời gian (tính theo đơn vị giây) mà current log cần sử dụng. Do thời gian lưu trữ không chính xác bằng khoảng thời gian log switch.

Tham số khởi taon này nên được thiết lập với giá trị khoảng 30 giây.

ARCHIVE_LAG_TARGET = 1800

Giá trị 0 tương ứng với việc không thực hiện chức năng log switching. Đây là giá trị thiết lập mặc định.

Ta có thể đặt giá trị cho tham số ARCHIVE_LAG_TARGET ngay cả khi database không ở trong chế độ sao lưu (standby database). Ví dụ, tham số ARCHIVE_LAG_TARGET có thể được thiết lập để bắt buộc các logs phải thực hiện thao tác switch và lưu trữ lên ổ đĩa.

ARCHIVE_LAG_TARGET là một tham số động và ta có thể thay đổi giá trị của tham số này thông qua câu lệnh ALTER SYSTEM SET.

8.4.2. Các yếu tố ảnh hưởng tới tham số ARCHIVE_LAG_TARGET

Có một số yếu tố cần được xem xét khi ta thiết lập giá trị cho tham số ARCHIVE_LAG_TARGET.

- Tổng thời gian switch (xem như là thời gian lưu trữ) các logs
- Tần suất thực hiện switch các log khi nó đầy
- Lượng dữ liệu có thể redo bị mất khi database làm việc ở chế độ standby

Tham số ARCHIVE_LAG_TARGET sẽ trở nên không hữu dụng khi log được switch trong một khoảng thời gian quá ngắn. Tuy nhiên, trong trường hợp các redo được tạo ra với tốc độ không đều như nhau, thì khoảng thời gian ngắt quãng (interval) sẽ đưa ra giới hạn trên đối với current log. Khi database ở trong trạng thái nghỉ (idle) và redo records không được tạo ra thì, sau khoảng thời gian interval, log switch sẽ xảy ra và đẩy và ghi tất cả các redo records lên standby database.

Trong trường hợp ARCHIVE_LAG_TARGET được thiết lập với giá trị quá thấp thì cũng không tốt cho hệ thống về mặt hiệu suất. Là vì hệ thống liên tục phải thực hiện các log switches. Do vậy ta nên chọn giá trị hợp lý để nâng cao hiệu suất hệ thống.

8.5.XÁC ĐỊNH CHẾ ĐỘ LƯU TRỮ

Để biết được các thông tin về việc lưu trữ, ta có thể sử dụng một số cách sau:

8.5.1. Sử dụng lệnh Server Manager

Câu lệnh này cho biết chế độ log của database.

Ví dụ:

```
SVRMGR> ARCHIVE LOG LIST
Database log mode No Archive Mode
Automatic archival Disabled
Archive destination ?/dbs/arch
Oldest online log sequence 688
Current log sequence 689
```

8.5.2. Sử dụng thông tin trong data dictionary

Ta cũng có thể sử dụng thông tin trong các data dictionary views: V$DATABASE và V$INSTANCE.

Ví dụ:

```
SVRMGR> SELECT name, log_mode
     2> FROM v$database;
NAME            LOG_MODE
---------       ---------
U15             NOARCHIVELOG
1 row selected.

SVRMGR> SELECT archiver
     2> FROM v$instance;
ARCHIVE
---------
STOPPED
1 row selected.
```

Ta cũng có thể xem các thông tin liên quan đến các groups và các members thông qua views data dictionary V$THREAD, V$LOG.

Các thông tin cần quan tâm:

- V$THREAD: GROUPS, CURRENT_GROUP#, SEQUENCE#
- V$LOG: GROUP#, MEMBERS, STATUS, SEQUENCE#, BYTES

Ví dụ:

```
SVRMGR>SELECT groups, current_group#,sequence#
     2>FROM v$thread;
GROUPS          CURRENT_GR  SEQUENCE#
----------      ----------  ----------
2               1           689
1 row selected.


SVRMGR>SELECT group#,sequence#,bytes,members,status
     2>FROM v$log;
GROUP#    SEQUENCE#   BYTES    MEMBERS   STATUS
--------- ----------  -------- --------- -------
1         688         1048576  1         CURRENT
2         689         1048576  1         INACTIVE
2 rows selected.
```

Trong câu lệnh ở trên, giá trị của cột STATUS được biểu hiện như sau:

- UNUSED chỉ ra online redo log group vẫn chưa được sử dụng. Trạng thái này tương ứng với việc online redo log file mới được thêm vào.
- CURRENT chỉ ra rằng online redo log group đang được sử dụng. Nó cũng ngầm đinh luôn trạng thái active đối với các online redo log group này.
- ACTIVE: trạng thái này ứng với the online redo log group vẫn đang được sử dụng nhưng không phải là online redo log group hiện thời.
- INACTIVE chỉ ra online redo log group không còn cần thiết cho việc khôi phục instance.

Để xác định tên của tất cả các member trong một group, ta có thể tra cứu thông tin trong V$LOGFILE: GROUP#, STATUS, MEMBER

Ví dụ:

```
SVRMGR>SELECT *
2>FROM v$logfile;
GROUP#           STATUS   MEMBER
----------       -------  ------------------------------
1                         /DISK3/log1a.rdo
2                         /DISK4/log2a.rdo
```

8.6.ĐIỀU KHIỂN CÁC LOG SWITCHS VÀ CHECKPOINTS

8.6.1. Thực hiện log switches

Log switches và checkpoint là các sự kiện xảy ra một cách tự động mỗi khi online redo log group đầy. Tuy nhiên, ta vẫn có thể phát sinh các Log switchs thông qua lệnh của Server Manager.

```
SVRMGR>ALTER SYSTEM SWITCH LOGFILE;
```

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

1. Sử dụng Backup Manager
2. Chọn Subsystem
3. Chọn Logfile --> Switch logfile

8.6.2. Thực hiện checkpoint

Ta cũng có thể phát sinh các Checkpoints thông qua lệnh:

```
SVRMGR>ALTER SYSTEM CHECKPOINT;
```

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

1. Sử dụng Backup Manager
2. Chọn Subsystem
3. Chọn Logfile --> Force checkpoint

8.6.3. Điều chỉnh các ngắt quãng checkpoints

Trong trường hợp database sử dụng các online redo log files lớn, ta có thể điều chỉnh lại các ngắt quãng đối với online redo log file đó thông qua các tham số:

- LOG_CHECKPOINT_INTERVAL: Số lượng blocks (tính theo số block của hệ điều hành) lớn nhất để thực hiện một checkpoint
- LOG_CHECKPOINT_TIMEOUT: Khoảng thời gian lớn nhất (tính theo đơn vị giây) để thực hiện một checkpoint.

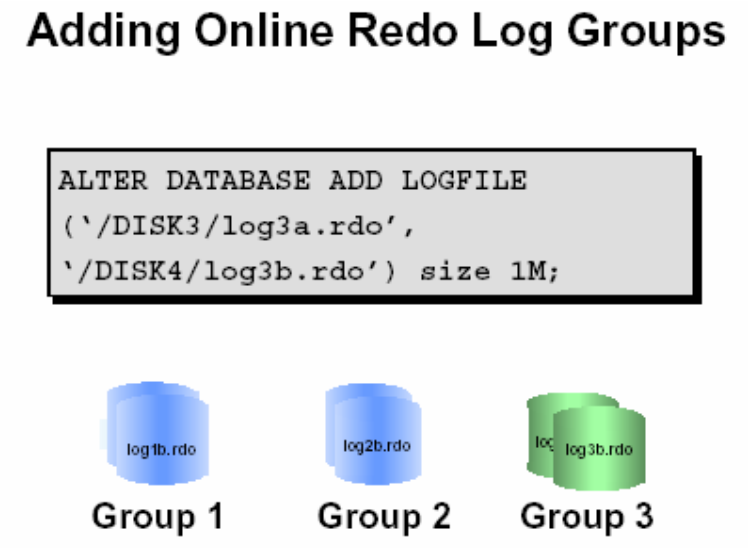
8.7.QUẢN TRỊ CÁC REDO LOG FILES

8.7.1. Bổ sung các online redo log groups

Trong một vài trường hợp, ta có thể cần tới việc nạp thêm các log groups hay các log members.

Cú pháp:

```
ALTER DATABASE [database]
ADD LOGFILE [GROUP integer] filespec
[, [GROUP integer] filespec]...]
```



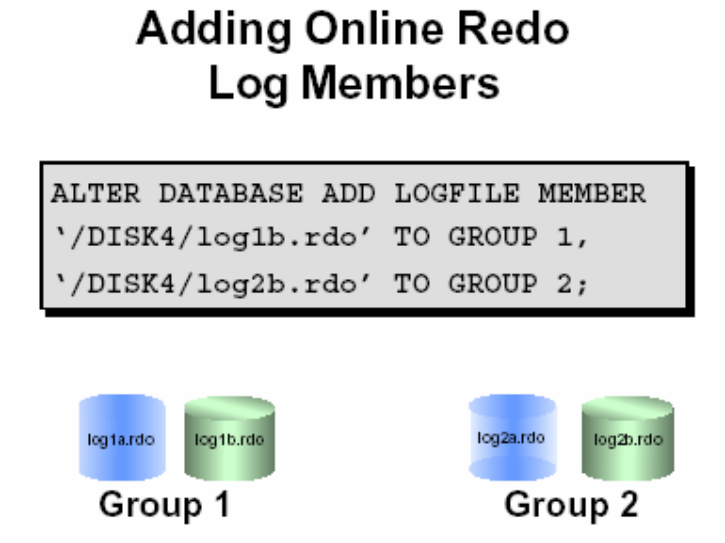
Hình vẽ 30. Bổ sung online redo log groups

Với câu lệnh trên, ta cần chỉ ra tên và đường dẫn của các members trong từng group cụ thể. Giá trị của tham số `GROUP` được chọn tương ứng với mỗi redo log file group. Trong trường hợp bỏ qua tham số này, Oracle server sẽ tự động sinh ra các giá trị thích hợp.

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

1. Sử dụng Backup Manager
2. Chọn Subsystem
3. Chọn Logfile --> Add Logfile Group

8.7.2. Bổ sung các online redo log members



Hình vẽ 31. Bổ sung online redo log members

Tương tự như các group, ta cũng có thể thêm mới các member cho từng group bằng câu lệnh SQL

```
ALTER DATABASE [database]
      ADD LOGFILE MEMBER
      [ 'filename' [REUSE]
      [,'filename' [REUSE]]...
      TO {GROUP integer
      |('filename'[, 'filename']...)
      }
      ]...
```

Lưu ý: tên file được chỉ ra cần kèm theo đường dẫn đầy đủ. Trong trường hợp không có đường dẫn, file sẽ được xem như được đặt trong thư mục mặc định. Nếu file thêm mới đã tồn tại, ta cần thêm vào tuỳ chọn `REUSE`.

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

1. Sử dụng Backup Manager
2. Chọn Subsystem
3. Chọn Logfile --> Add Logfile Member

8.7.3. Định lại chỗ cho các redo log file

Trong một vài trường hợp, ta cần phải dịch chuyển các file redo log tới một vị trí khác, để đảm bảo an toàn chẳng hạn. Khi này, ta cần thực hiện theo các bước sau:

1. Tắt database.
2. Sao chép các online redo log files tới một địa điểm mới.
3. Restart database ở chế độ mount.
4. Thực hiện lệnh `ALTER DATABASE RENAME FILE` để thay đổi con trỏ trong control file, trỏ tới một đường dẫn file mới.
5. Mở lại database (Lệnh: `ALTER DATABASE OPEN`).

Câu lệnh đổi tên file:

```
ALTER DATABASE [database]
      RENAME FILE 'filename'[, 'filename']...
      TO 'filename'[, 'filename']...
```

Lưu ý: Phải tồn tại file ở đường dẫn mới chỉ ra.

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

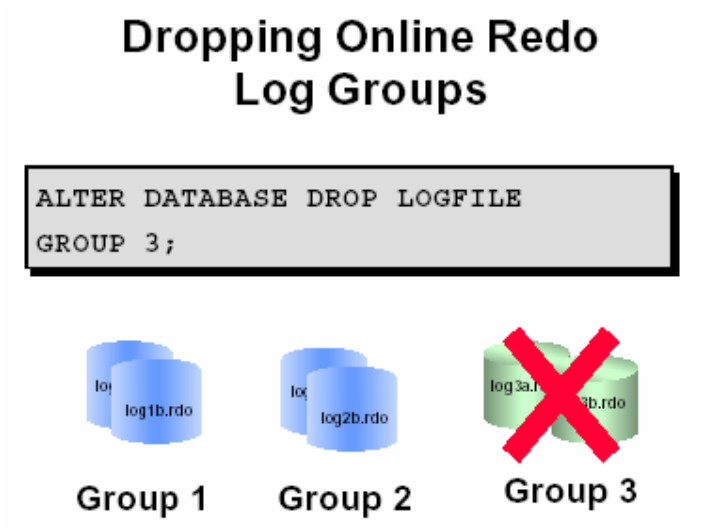
1. Sử dụng Backup Manager
2. Chuyển tới nút Logfile Group
3. Chọn log file group tương ứng
4. Thay đổi tên file trong trường thuộc tính.

8.7.4. Ngừng sử dụng các Online redo log groups

Để có thể thay đổi kích thước các online redo log groups, ta có thể thêm mới các online redo log group và xoá bỏ các online redo log group đã có.

Sử dụng lệnh của Server Manager để ngưng sử dụng online redo log group:

```
ALTER DATABASE [database]
      DROP LOGFILE
      {GROUP integer|('filename'[, 'filename']...)}
      [,{GROUP integer|('filename'[, 'filename']...)}]...
```



Hình vẽ 32. Ngừng sử dụng Online redo log groups

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

1. Sử dụng Backup Manager
2. Chuyển tới nút Logfile Group
3. Chọn log file group tương ứng
4. Chọn Logfile --> Drop Logfile Group
5. Bấm nút OK.

Một số điểm cần lưu ý khi xoá log groups

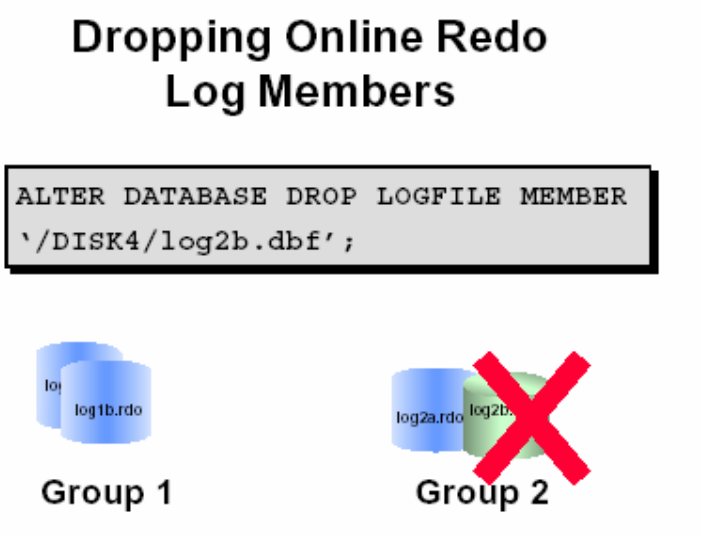
- Một instance cần ít nhất hai nhóm (group) các online redo log files.
- Không thể huỷ (drop) group đang ở trạng thái active.
- Khi huỷ một online redo log group, thực chất ta chỉ huỷ về mặt logic mà thôi. Oracle sẽ không tiếp tục quản lý nó nữa. Tuy nhiên, các file sẽ vẫn còn và không bị xoá bởi hệ điều hành.

8.7.5. Ngừng sử dụng các Online redo log members

Tương tự như các log group, đối với các log members ta cũng có thể ngưng sử dụng.

Sử dụng lệnh của Server Manager để ngưng sử dụng online redo log member:

```
ALTER DATABASE [database]
      DROP LOGFILE MEMBER 'filename'[, 'filename']...
```



Hình vẽ 33. Ngừng sử dụng Online redo log members

Trong Oracle Enterprise Manager – OEM, ta làm theo các bước sau:

1. Sử dụng Backup Manager
2. Chuyển tới nút Logfile Group
3. Chọn log file group tương ứng
4. Chọn Logfile --> Drop Logfile Member
5. Bấm nút OK.

Một số điểm cần lưu ý khi xoá log members

- Không thể ngừng sử dụng member của group mà có trạng thái là `VALID`.
- Nếu group đang trong trạng thái active, ta cần phải thực hiện log switch để chuyển sử dụng sang một log group khác trước khi ngưng sử dụng các member của group hiện thời.
- Khi huỷ một online redo log member, thực chất ta chỉ huỷ về mặt logic các file vẫn không bị xoá bởi hệ điều hành.

8.7.6. Xoá rỗng Online redo log file

Trong một vài trường hợp các members bị lỗi, quản trị viên database có thế xử lý bằng cách khởi tạo lại các log file thông qua lệnh SQL để khởi tạo lại:

```
ALTER DATABASE CLEAR LOGFILE
```

Cú pháp:

```
ALTER DATABASE [database]
      CLEAR [UNARCHIVED] LOGFILE
      {GROUP integer|('filename'[, 'filename']...)}
      [,{GROUP integer|('filename'[, 'filename']...)}]...
```

Sử dụng lệnh này cũng tương đương với việc thêm mới các online redo log file và xoá bỏ các redo log file hiện thời.

Lưu ý:

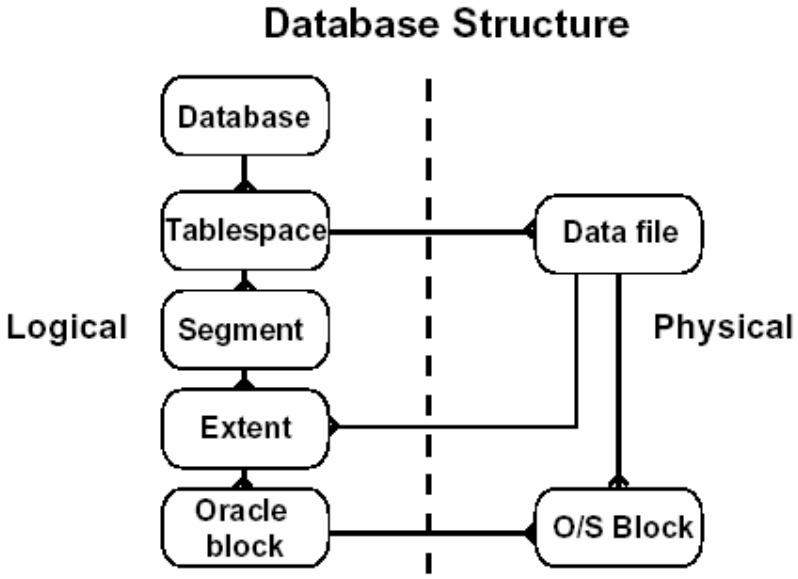
Khi xoá rỗng logfile mà nó không dùng để lưu trữ, ta cần bổ sung từ khoá `UNARCHIVED`.

Chương 9. QUẢN TRỊ TABLESPACES VÀ DATA FILES

9.1.CẤU TRÚC CỦA DATABASE

Cấu trúc database bao gồm cấu trúc logic và cấu trúc vật lý.

Cấu trúc vật lý bao gồm tập hợp các control files, online redo log files và các data files. Cấu trúc logic bao gồm các schema objects tablespaces, segments, extents và data blocks.



Hình vẽ 34. Cấu trúc database

9.1.1. Quan hệ giữa database với các tablespaces và data files

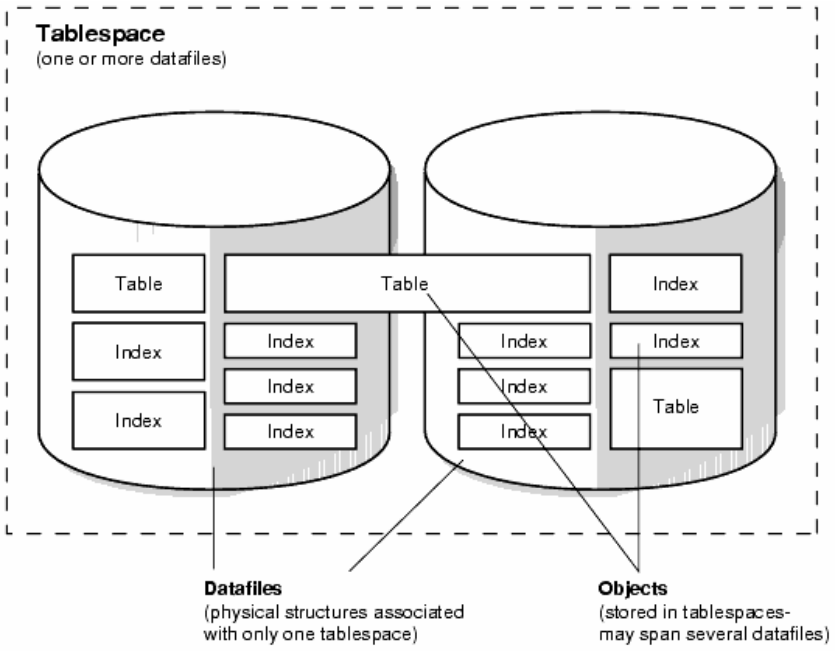
Về mặt logic, một database có thể phân nhỏ thành nhiều phần gọi là các tablespaces.

Tablespace

- Một tablespace chỉ thuộc một database.
- Mỗi tablespace có thể chứa một hay nhiều data file thuộc hệ điều hành.
- Tablespaces có thể đặt ở trạng thái online hay offline trong lúc database đang chạy.
- Ngoại trừ tablespace `SYSTEM` hay tablespace chứa rollback segments đang có trạng thái `ACTIVE`, các tablespaces đều có thể chuyển về trạng thái offline trong lúc database đang chạy.
- Các tablespaces cũng có thể chuyển đổi trạng thái read-write hay read-only.

Sử dụng tablespace

- Để điều khiển vùng không gian cấp phát và gán cho mỗi users
- Với việc đặt chế độ online hay offline cho các tablespace, ta có thể thay đổi tính sẵn dùng (availability) của các dữ liệu trong các tablespace
- Ta cũng có thể phân biệt các dữ liệu lưu trữ giữa các thiết bị để tăng hiệu suất sử dụng database.
- Thực hiện sao lưu và phục hồi dữ liệu từng phần, nâng cao hiệu suất hệ thống



Hình vẽ 35. Quan hệ giữa tablespace và datafile

Data files

Mỗi một tablespace có thể bao gồm một hay nhiều data files, là các file thuộc hệ điều hành dùng để lưu trữ dữ liệu trong tablespace. Các data files có một số tính chất chính sau:

- Một data file chỉ thuộc về một tablespace.
- Quản trị viên database có thể thay đổi kích thước của data file ngay cả khi nó đã được tạo lập, làm tăng tính năng động cho các đối tượng có trong tablespace.

9.1.2. Quan hệ giữa segment với các extent và các blocks

Oracle cho phép điều chỉnh không gian đĩa thông qua việc thay đổi kích thước của các cấu trúc lưu trữ logic như: tablespaces, segments, extents và blocks.

Setgments

Một segment là vùng không gian cấp phát tương ứng với một kiểu cấu trúc logic có trong một tablespace. Ta có thể phân ra làm một số loại segment chính sau:

- Data segments
- Index segments
- Temporary segments
- Rollback segments

Một segment cụ thể là một data segment có thể được trải rộng trên nhiều datafiles thuộc một tablespace.

Extents

Extent là một cấp độ phân chia về mặt logic tiếp theo của databse. Một extent là tập hợp liên tiếp các blocks dữ liệu. Mỗi kiểu segment được quy đinh bao gồm một hay nhiều extents. Khác với segments, một extent chỉ được nằm duy nhất trên một data file.

Data Blocks

Đây là đơn vị lưu trữ *(lưu ý không phải là đơn vị quản lý)* dữ liệu nhỏ nhất trong database Oracle. Một block dữ liệu sẽ tương ứng với một hay nhiều blocks của hệ điều hành. (Ví dụ: hệ điều hành Windows 32, 1 block hệ điều hành = 32 kbytes = 32*1024 bytes). Kích thước của block dữ liệu được xác định bởi tham số khởi tạo `DB_BLOCK_SIZE` ngay khi database được tạo. Block trong database cũng là đơn vị vào ra nhỏ nhất.

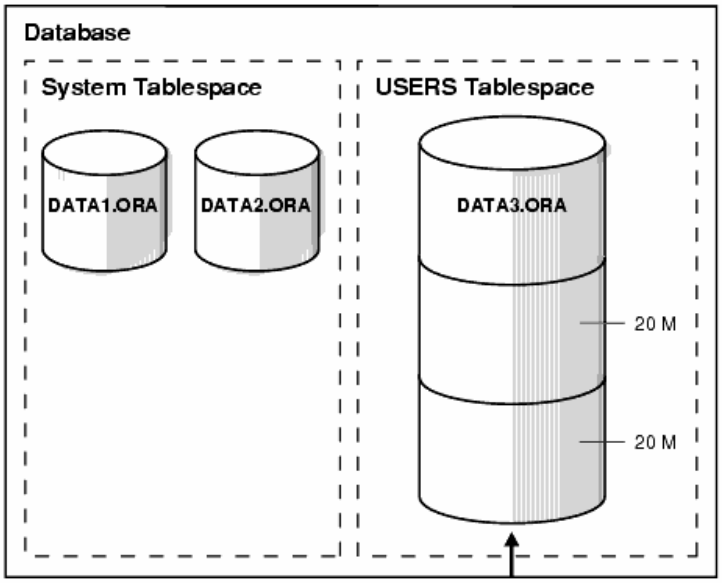
9.2.PHÂN LOẠI CÁC TABLESPACES

9.2.1. Tablespace SYSTEM và non-SYSTEM

Một database gồm có ít nhất một tablespace là tablespace SYSTEM, là nơi lưu trữ các thông tin của hệ thống. Ngoài ra, database còn có thể thêm vào các tablespace khác, đó là các non-SYSTEM tablespaces, chứa dữ liệu của các user.

Tablespace SYSTEM

- Có trong tất cả các database
- Chứa thông tin về các data dictionary views, các định nghĩa của stored procedures, packages, và các database triggers dưới dạng PL/SQL program units.
- Chứa SYSTEM rollback segment
- Không nên chứa dữ liệu người dùng trong tablespace này mặc dù có thể.



Hình vẽ 36. Dữ liệu người dùng nên đặt trong tablespace riêng

Non-SYSTEM Tablespace

- Chứa các rollback segments
- Chứa các temporary segments
- Chứa các data segments
- Chứa các index segments

9.2.2. Tablespaces read-only / read-write

Tablespaces read-only

Mục đích chính của việc sử dụng các tablespaces read-only (chỉ đọc) là hạn chế các thủ tục cần thiết khi thực hiện sao lưu và phục hồi một phần lớn dữ liệu không bị thay đổi (static) của database. Oracle không thực hiện cập nhật các files nằm trong tablespace read-only, vì thế các files có thể được đặt trong thiết bị chỉ đọc như CD ROMs hay ổ đĩa WORM drives (Write Once-Read Many).

Mỗi khi tạo mới một tablespace, hệ thống sẽ tạo cho ta một tablespace có đủ cả quyền đọc và quyền ghi. Ta có thể thay đổi lại thuộc tính tablespace thành read-only thông qua mệnh đề `READ ONLY` trong câu lệnh `ALTER TABLESPACE`. Việc này sẽ dẫn tới tất cả các datafiles thuộc tablespace đó sẽ được đặt lại thuộc tính là read-only.

Câu lệnh `ALTER TABLESPACE ... READ ONLY` sẽ đặt tablespace vào chế độ chuyển tiếp (*transitional read-only)* và chờ cho tất cả các transactions trên đó kết thúc (commit hoặc roll back). Chế độ chuyển tiếp này sẽ không cho phép bất kỳ một thao tác ghi mới nào được thực hiện trên tablespace ngoại trừ việc rollback các transactions hiện thời và thay đổi dữ liệu trong các blocks trong tablespace. Do đó, chế độ chuyển tiếp của tablespace cũng hệt như tablespace read-only đối với các câu lệnh mới của người dùng ngoại trừ lệnh `ROLLBACK`. Sau khi tất cả các transactions hiện thời đã kết thúc thì câu lệnh `ALTER TABLESPACE ... READ ONLY` mới được xem là kết thúc và tablespace được đặt ở chế độ read-only.

Đặt chế độ read-only cho tablespace không làm ảnh hưởng tới trạng thái offline hay online của tablespace đó. Các Offline datafiles vẫn không thể truy xuất được. Việc đưa một datafile trong tablespace read-only vào chế độ online sẽ cho phép user có thể đọc dữ liệu trong file đó. File này vẫn không thể viết dữ liệu vào trừ phi tablespace tương ứng được đặt lại ở chế độ cho phép đọc và ghi.

Read-only tablespaces không thể bị sửa đổi. Để cập nhật dữ liệu trong một read-only tablespace, trước tiên ta cần đặt lại chế độ cho tablespace là read-write. Sau đó, thực hiện cập nhật dữ liệu trong tablespace rồi đặt lại chế độ read-only cho tablespace đó.

Do các read-only tablespaces không bị sửa đổi nên ta cũng không cần thiết phải thực hiện việc backup dữ liệu trên nó nhiều lần. Và ta cũng không cần thiết phải phục hồi lại các read-only tablespaces, do dữ liệu trong đó không bị thay đổi.

Ta không thể bổ sung các datafiles vào tablespace read-only, ngay cả khi đã đặt chế độ cho tablespace là offline. Bởi vì, khi bổ sung một datafile, Oracle sẽ phải cập nhật phần thông tin header trong khi đó thao tác ghi lên tablespace này là không được phép.

Tablespace read-write

Trái với tablespace read-only, với các tablespace read-wirte, ta có thể thực hiện các thao tác đọc và ghi trên đó.

Ta cũng có thể sử dụng mệnh đề `READ WRITE` trong câu lệnh `ALTER TABLESPACE` để thay đổi trạng thái tablespace read-only thành trạng thái read-write.

9.2.3. Temporary tablespace / permanent tablespace

Temporary tablespaces được sử dụng để dành riêng cho các thao tác sắp xếp dữ liệu. Trong temporary tablespace không có bất cứ segments dữ liệu nào nằm trong đó.

Sort segments có thể cùng được chia sẻ sử dụng khi nhiều thao tác sắp xếp cùng được thực hiện. Một sort segment được sử dụng cho tất cả các instance có thực hiện thao tác sắp xếp trên một tablespace.

Việc sử dụng các temporary tablespaces cho phép nâng cao hiệu suất thực hiện mỗi khi có nhiều thao tác sắp xếp được thực hiện trên một vùng nhớ lớn và không phù hợp với kích thước của bộ nhớ trong của máy tính. Sort segment thuộc temporary tablespace được tạo ra vào ngay thời điểm đầu của thao tác sắp xếp. Sort segment sẽ được cấp thêm vùng nhớ và mở rộng dần cho tới khi kích thước của segment ngang bằng hoặc lớn hơn tổng số kích thước lưu trữ cần thiết cho việc thực hiện tất cả các thao tác sắp xếp của instance.

Các tablespaces không phải là *temporary tablespaces* được gọi là các permanent tablespaces. Các permanent tablespace được sử dụng để lưu trữ dữ liệu trong database.

9.3.QUẢN LÝ KHÔNG GIAN TRONG TABLESPACES

Tablespaces cấp phát vùng không gian theo các *extents*. Tablespaces sử dụng hai phương pháp khác nhau để cấp phát và giải phóng vùng không gian lưu giữ:

- Quản lý các extents qua data dictionary (dictionary-managed tablespaces)
- Quản lý các extents qua tablespace (locally-managed tablespaces)

Ngay khi tạo tablespace, ta cần lựa chọn luôn phương pháp quản lý vùng không gian sẽ được áp dụng cho tablespace đó. Khi đã chọn rồi, ta không thể thay đổi phương pháp quản lý không gian nữa.

9.3.1. Dictionary-Managed Tablespaces

Trong phương pháp này tablespace sử dụng data dictionary để quản lý các extents của nó. Oracle cập nhật từng tables trong data dictionary mỗi khi cấp phát, giải phóng hay sử dụng lại một extent. Oracle cũng lưu lại các thông tin rollback của việc cập nhật các dictionary tables.

Theo mặc định, phương pháp quản lý này sẽ được áp dụng cho các tablespaces có trong database. Trong các phiên bản Oracle 8.0 hoặc sớm hơn, chỉ có một phương pháp đó chính là phương pháp này.

9.3.2. Locally-Managed Tablespaces

Bên cạnh đó, tablespace cũng có thể quản lý các extents của nó thông qua một bitmap (ánh xạ bit) trong từng datafile từ đó xác định được trạng thái của các blocks trong datafile là đang sử dụng hay đã được giải phóng. Mỗi một bit trong bitmap sẽ tương ứng với một block hay một nhóm các blocks. Mỗi khi có một extent được cấp phát, giải phóng hay tái sử dụng, Oracle sẽ thay đổi giá trị của bitmap theo đúng như trạng thái mới của các blocks. Việc thay đổi này sẽ không làm phát sinh các thông tin trong rollback do không có thao tác cập nhật dữ liệu nào trong các tables của data dictionary (Ngoại trừ trường hợp đặc biệt liên quan đến các thông tin hạn mức (quota) của tablespace).

Locally-managed tablespaces có một số ưu điểm hơn so với dictionary-managed tablespaces là:

- Quản lý cục bộ các extents tránh các thao tác quản lý không gian theo kiểu đệ quy. Việc này có thể xảy ra khi sử dụng phương pháp dictionary-managed tablespaces nếu việc sử dụng hay giải phóng không gian là kết quả của các thao tác sử dụng hay giải phóng không gian trong rollback segment hay data dictionary table.

- Quản lý cục bộ các extents một cách tự động các vùng không gian giải phóng liền kề với nhau. Điều này là cần thiết khi thực hiện công việc hợp nhất các extents rồi.

Kích thước của các extents được quản lý cục bộ có thể được xác định tự động bởi hệ thống. Mặt khác, tất cả các extents có thể có cùng một kích cỡ như nhau trong phương pháp locally-managed tablespace.

Mệnh đề `LOCAL` trong phần `EXTENT MANAGEMENT` của câu lệnh `CREATE TABLESPACE` sẽ chỉ rõ phương thức quản lý không gian:

- Với các permanent tablespaces và temporary tablespaces, ta có thể sử dụng mệnh đề `EXTENT MANAGEMENT LOCAL`.
- Trong phiên bản 8i, phương pháp quản lý này vẫn chưa được áp dụng cho tablespace SYSTEM. Nếu áp dụng, hệ thống sẽ phát sinh lỗi 809225.

9.4.THIẾT LẬP TRẠNG THÁI CHO TABLESPACES

Quản trị viên database có thể thiết lập trạng thái cho các tablespaces là *online* (có thể sử dụng) hay *offline* (không thể sử dụng) ngoại trừ tablespace SYSTEM mỗi khi mở database. Tablespace SYSTEM luôn ở trạng thái online mỗi khi database được mở bởi vì Oracle luôn phải sử dụng các dữ liệu trong dictionary.

Một tablespace thông thường ở chế độ online khi đó, các dữ liệu trong nó là sẵn sàng đối với các database users. Tuy nhiên, quản trị viên database có thể đặt chế độ offline cho tablespace:

- Khi này một phần của database sẽ không thể truy xuất được, trong khi phần còn lại vẫn có thể truy xuất bình thường.
- Thực hiện offline tablespace khi backup dữ liệu (mặc dù ta vẫn có thể backup dữ liệu ngay khi database đang chạy và các tablespace ở trạng thái online).

Lưu ý: ta không thể đặt chế độ offline cho tablespace nếu nó có chứa các rollback segments đang được sử dụng.

Đặt Offline cho tablespace

Khi một tablespace được đưa ra offline, Oracle sẽ không cho phép thực hiện các câu lệnh SQL có tham chiếu tới các objects lưu trữ trong tablespace này. Oracle lưu lại các dữ liệu rollback tương ứng khi thực hiện câu lệnh SQL trong một rollback segment khác có trong tablespace SYSTEM thay vì là rollback segment có trong tablespace được offline nếu có. Tablespace được đưa về online trở lại, Oracle sẽ áp lại các dữ liệu rollback đang có trong tablespace SYSTEM vào tablespace đó.

Ta chỉ có thể đưa một tablespace thành online trong chính database mà nó được tạo, không thể đặt online cho tabalespace trong một database khác được. Việc này được giám sát bởi các thông tin có trong dictionary.

Oracle tự động thực hiện chuyển chế độ từ online thành offline đối với tablespaces mỗi khi xảy ra sự cố hệ thống. Ví dụ như: tiến trình DBW*n* gặp lỗi.

9.5.TRAO ĐỔI CÁC TABLESPACES GIỮA DATABASES

Ta có thể sử dụng chức năng *transportable tablespaces* để dịch chuyển một phần của một database sang một database Oracle khác. Việc trao đổi các tablespaces giữa các database là rất hữu ích cho:

- Việc dịch chuyể dữ liệu từ hệ thống xử lý trực tuyến (OLTP – online transaction processing systems) sang thành dữ liệu của hệ thống kho dữ liệu (data warehouse staging systems).
- Cập nhật kho dữ liệu (data warehouses) và các dữ liệu thuộc hệ thống.
- Nạp các dữ liệu từ các kho cơ sở dữ liệu trung tâm (central data warehouses).
- Lưu trữ các dữ liệu của hệ thống OLTP and data warehouse systems efficiently.
- Cung cấp dữ liệu cho các khách hàng hoặc người sử dụng nội bộ.

Dịch chuyển dữ liệu thông qua việc trao đổi các tablespaces cho phép di chuyển dữ liệu nhanh chóng và hiệu quả hơn các cách dịch chuyển dữ liệu khác như export/import hay unload/load đối với cùng một dữ liệu, Do việc trao đổi các tablespace chỉ đòi hỏi phải sao chép các datafiles rồi tích hợp thông tin về cấu trúc của tablespace vào database mới. Có thể sử dụng phương pháp trao đổi các tablespaces để dịch chuyển các index data, do đó, để tránh việc tái tạo lại (rebuilds) các index, ta có thể thực hiện công việc này để nạp dữ liệu trong các bảng.

9.5.1. Một số hạn chế trong việc trao đổi các tablespace:

- Database nguồn và đích phải được chạy trên cùng một nền phần cứng (hardware platform). Ví dụ, có thể trao đổi các tablespaces giữa database Oracle chạy trên hệ điều hành Sun Solari, hoặc trao đổi các tablespaces giữa các databases Oracle chạy trên hệ điều hành NT. Tuy vậy, ta không thể trao đổi các tablespace giữa database Oracle chay trên SUN Solaris với các database Oracle chạy trên NT.
- Database nguồn và đích phải có cùng một kích thước của data block.
- Database nguồn và đích phải sử dụng cùng một tập ký tự sử dụng trong database (national character set).
- Không thể chuyển đổi tablespace sang database đích khi database này đã có một tablespace có cùng tên.
- Việc chuyển đổi tablespaces không được hỗ trợ:
    - Snapshot/replication
    - Function-based indexes
    - Scoped REFs
    - Domain indexes (Một kiểu index mới, cho phép mở rộng việc đánh chỉ số)

9.5.2. Các bước thực hiện chuyển đổi một tablespace giữa các database

1. Chỉ có thể thực hiện trao đổi các tablespaces mà nó không chứa các tham chiếu tới tablespace khác.

2. Tạo một transportable tablespace set.

   Transportable tablespace set chứa các datafiles ứng với tập các tablespaces được sử dụng để chuyển đổi các file có chứa thông tin cấu trúc của các tablespaces dịch chuyển.

   (Xem minh hoạ việc tạo một transportable tablespace set ở phia dưới).

3. Chuyển đổi tablespace.

   Sao chép các datafiles và export file sang database đích. Có thể sử dụng các công cụ sao chép file thông thường của hệ điều hành để thực hiện công việc này

4. Đưa tablespace vào sử dụng (plug-in).

   Thực hiện công việc Import để đưa các tablespaces vào database đích.

Minh hoạ việc trao đổi tablespace

1. Để biết tablespace SALES_1 và SALES_2 có chứa các tham chiếu trong nó không, ta thực hiện câu lệnh:

```
EXECUTE dbms_tts.transport_set_check('sales_1,sales_2', TRUE);
```

Câu lệnh này sinh ra kết quả và lưu trong view có tên là: TRANSPORT_SET_VIOLATIONS. Sử dụng câu lệnh truy vấn để xem kết quả:

```
SELECT * FROM transport_set_violations;
```

Lệnh truy vấn kết xuất kết quả rỗng cho biết tablespace không chứa các tham chiếu tới tablespace bên ngoài.

2. Tạo transportable tablespace set

Phát lệnh thay đổi trạng thái của tablespace về trạng thái read-only để không cho phép cập nhật dữ liệu vào tablespace này, chuẩn bị cho việc trao đổi tablespace.

```
ALTER TABLESPACE sales_1 READ ONLY;
```

Sử dụng công cụ tiện ích Export của Oracle để kết xuất các tablespace này:

```
EXP TRANSPORT_TABLESPACE=y TABLESPACES=(sales_1,sales_2)
     TRIGGERS=y/n CONSTRAINTS=y/n GRANTS=y/n FILE=expdat.dmp
```

TRIGGERS=Y – cho phép kết xuất; N – không cho phép kết xuất.

GRANTS=Y – kết xuất cả các quyền trên mỗi bảng thuộc tablespace đó; N – không kết xuất.

CONSTRANTS=Y – các ràng buộc tham chiếu sẽ được kết xuất; N – không kết xuất các ràng buộc tham chiếu.

3. Thực hiện sao chép các datafile của tablespace vừa được kết xuất ra một vị trí khác.

4. Đặt lại trạng thái bình thường cho tablespace vừa được xem xét.

```
ALTER TABLESPACE sales_1 READ WRITE;
```

5. Đưa bản sao của các datafile vừa được sao chép vào vị trí tương ứng với database đích.

6. Connect vào database mới với mức quyền SYSDBA.

7. Đưa các tablespaces đã được kết xuất vào database mới

```
IMP      TRANSPORT_TABLESPACE=y
  DATAFILES=('c:\db\sales_jan','c:\db\sales_feb',...)
  TABLESPACES=(sales_1,sales_2) TTS_OWNERS=(dcranney,jfee)
  FROMUSER=(dcranney,jfee) TOUSER=(smith,williams) FILE=expdat.dmp
```

9.6.TẠO TABLESPACE

9.6.1. Lệnh tạo tablespace

Ta có thể sử dụng câu lệnh SQL để tạo một tablespace.

Cú pháp:

```
CREATE TABLESPACE tablespace
```

```
        DATAFILE filespec [autoextend_clause]
        [, filespec [autoextend_clause]]...
        [MINIMUM EXTENT integer[K|M]]
        [DEFAULT storage_clause]
        [PERMANENT|TEMPORARY]
        [ONLINE|OFFLINE]

    storage_clause:= =
    STORAGE ( [INITIAL integer[K|M]]
              [NEXT integer[K|M]]
              [MINEXTENTS integer]
              [MAXEXTENTS {integer|UNLIMITED}]
              [PCTINCREASE integer]
        )
```

Với:

tablespace	tên của tablespace được tạo
DATAFILE	tên data files của tablespace được tạo
DEFAULT STORAGE	tham số lưu trữ mặc định cho tất cả các đối tượng được tạo lập trong tablespace
MINIMUM EXTENT	kích thước tối thiểu của extent được sử dụng value
ONLINE	đặt chế độ sử dụng (Online) cho tablespace ngay từ khi tạo lập
OFFLINE	đặt chế độ chưa sử dụng (Offline) cho tablespace ngay từ khi tạo lập
PERMANENT	tablespace có thể sử dụng để lưu trữ các đối tượng thường trú
TEMPORARY	tablespace chỉ sử dụng để lưu trữ các đối tượng trung gian (temporary objects). Ví dụ: sử dụng để lưu trữ dữ liệu khi sắp xếp theo câu lệnh ORDER BY

Ví dụ:

```
CREATE TABLESPACE app_data
     DATAFILE '/DISK4/app01.dbf' SIZE 100M,
     '/DISK5/app02.dbf' SIZE 100M
     MINIMUM EXTENT 500K
     DEFAULT STORAGE (INITIAL 500K NEXT 500K
     MAXEXTENTS 500 PCTINCREASE 0);
```

Cũng tương tự, ta có thể thực hiện trong Oracle Enterprise Manager – OEM:

1. Chạy Oracle Storage Manager.
2. Chọn Tablespace—>Create.
3. Trong General page của bảng thuộc tính, nhập vào tên tablespace rồi chọn ADD.
4. Trong bảng thuộc tính Create Datafile, chỉ ra các data file.
5. Trong phần Extents page, nhập vào các thông tin lưu giữ

6. Chọn mục Create.

Hạn chế

Số lượng tối đa các tablespaces trên mỗi database là 64.

Số lượng tối đa các data files trong mỗi tablespace là 1023.

9.6.2. Chế độ quản lý các tablespaces

Với câu lệnh tạo tablespace thông thường như ở trên, Oracle server sẽ tạo tablespace với chế độ quản lý là Dictionary-Managed Tablespaces

Để thực hiện quản lý tablespace theo phương pháp Locally-Managed Tablespaces ta cần đưa thêm vào câu lệnh mệnh đề: `MANAGEMENT LOCAL AUTOLOCATE`.

Ví dụ:

```
CREATE TABLESPACE lmtbsb
      DATAFILE 'c:\data\lmtbsb01.dbf' SIZE 50M
      EXTENT MANAGEMENT LOCAL AUTOALLOCATE;
```

9.6.3. Tạo temporary tablespace

Quản trị viên database có thể tạo một temporary tablespace sử dụng cho việc sắp xếp các dữ liệu không dùng để lưu trữ thường trú các dữ liệu.

Để tạo temporary tablespace, ta có thể sử dụng lệnh SQL giống như lệnh tạo tablespace thông thường, nhưng có thêm từ khoá `TEMPORARY` ở cuối.

Ví dụ:

```
CREATE TABLESPACE sort
      DATAFILE '/DISK2/sort01.dbf' SIZE 50M
      MINIMUM EXTENT 1M
      DEFAULT STORAGE (INITIAL 2M NEXT 2M
      MAXEXTENTS 500 PCTINCREASE 0)
      TEMPORARY;
```

Với Oracle Enterprise Manager, ta làm theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chọn Tablespace—>Create.
3. Trong General page, nhập vào tên tương ứng rồi chọn ADD để hiển thị mục Create Datafile.
4. Trong Create Datafile chỉ ra từng data file cụ thể.
5. Chọn TEMPORARY trong nhóm chọn radio button.
6. Bấm nút Create.

9.6.4. Các tham số lưu trữ

Lượng không gian dùng cho một tablespace được xác định trong mệnh đề lưu trữ (storage clause). Các tham số này được xác định ngay tại thời điểm tạo tablespace. Trong trường hợp không chỉ rõ các tham số này trong lệnh tạo lập (`CREATE`), các tham số sẽ được sử dụng các giá trị theo mặc định.

Có một số tham số lưu trữ cần quan tâm sau:

- `INITIAL` quy định kích thước của extent đầu tiên. Kích thước nhỏ nhất của extent đầu tiên là 02 block = (`2*DB_BLOCK_SIZE`). Mặc định, kích thước này là 5 blocks = (`5* DB_BLOCK_SIZE`).
- `NEXT` ứng với kích thước của extent thứ hai. Kích thước tối thiểu là 01 block. Mặc định, kích thước này là 5 blocks = (`5* DB_BLOCK_SIZE`).
- `MINEXTENTS` số lượng extent được tạo lập mỗi khi segment được tạo lập. Mặc định giá trị này là 1.
- `PCTINCREASE` phần trăm tăng kích thước extent. Kích thước của một extent được xác định theo kích thước:

$$Size_n = NEXT \times \left(1 + \frac{PCTINCREASE}{100}\right)^{(n-2)}$$

Với: $Size_n$ kích thước của extent thứ n

Ví dụ: `NEXT = 200K, PCTINCREASE = 50`. Ta tính được extent thứ hai = 200K, extent thư ba = 300K, extent thứ tư = 450K

- `MAXEXTENTS` xác định số lượng tối đa các extents có trong một segment. Giá trị nhỏ nhất là 1. Giá trị lớn nhất theo mặc định phụ thuộc vào kích thước của block dữ liệu. Giá trị này cũng có thể được xác định thông qua giá trị `UNLIMITED`, tương đương với giá trị là `2147483645`.

9.7.CÁC THAY ĐỔI ĐỐI VỚI TABLESPACE

9.7.1. Chuyển đổi một tablespace thành một temporary tablespace

Ta có thể thay đổi các tablespaces đang tồn tại để biến nó thành một temporary tablespace.

Ví dụ:

```
ALTER TABLESPACE tbsa TEMPORARY;
```

9.7.2. Thêm mới các tablespace

Để mở rộng không gian của tablespace ta có thể thực hiện theo hai cách sau:

- Thêm mới các data file vào tablespace
- Thay đổi dung lượng các data files

Hoặc ta cũng có thể sử dụng câu lệnh SQL can thiệp như sau:

```
ALTER TABLESPACE tablespace
      ADD DATAFILE filespec [autoextend_clause]
      [, filespec [autoextend_clause]]...
```

Với Oracle Enterprise Manager, ta làm theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chọn Tablespace—>Create.
3. Trong General page, nhập vào tên tương ứng rồi chọn ADD để hiển thị mục Create Datafile.
4. Trong Create Datafile chỉ ra từng data file cụ thể.

9.7.3. Mở rộng data files

Ta có thể thực hiện mở rộng (thay đổi) kích thước data file theo hai cách:

- Mở rộng theo chế độ tự động. Sử dụng từ khoá: `AUTOEXTENTED`
- Mở rộng theo chế độ can thiệp trực tiếp (manually). Sử dụng lệnh `ALTER TABLESPACE, ALTER DATABASE`

Thiết lập chế độ AUTOEXTENT trong khi tạo file

Cú pháp:

```
ALTER TABLESPACE tablespace
      ADD DATAFILE filespec [autoextend_clause]
      [, filespec [autoextend_clause]]...
```

Ví dụ:

```
ALTER TABLESPACE app_data
      ADD DATAFILE
      '/DISK6/app04.dbf' SIZE 200M
      AUTOEXTEND ON NEXT 10M
      MAXSIZE 500M;
```

Trong OEM ta thực hiện các bước sau:

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Tablespace.
3. ChọnTablespace—>Add Datafile.
4. Trong General page nhập vào các thông tin của file.
5. Trong Autoextend page nhập vào các thông tin tương ứng.
6. Bấm nút Create.

Thiết lập chế độ AUTOEXTENT khi data file đã tồn tại

Cú pháp:

```
ALTER DATABASE [database]
      DATAFILE 'filename'[, 'filename']...
      autoextend_clause
```

Trong OEM ta thực hiện các bước sau:

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Tablespace node.
3. Chọn data file.
4. Trong phần Autoextend page, bấm vào nút Enable Auto Extend.
5. Bấm nút Apply.

9.7.4. Thay đổi kích thước data file

Thay vì mở rộng kích thước của database bằng cách thêm vào các data file, quản trị viên cũng có thể mở rộng bằng cách điều chỉnh tăng kích thước của data file.

Sử dụng câu lệnh SQL sau để thay đổi kích thước của data file

```
ALTER DATABASE [database]
```

```
        DATAFILE 'filename'[, 'filename']...
        RESIZE integer[K|M]
```

Với:

```
integer          Kích thước tuyệt đối của file data file
```

Sử dụng câu lệnh SQL sau để thay đổi nơi lưu trữ mặc định:

```
ALTER TABLESPACE tablespace
      {MINIMUM EXTENT integer[K|M]
      |DEFAULT storage_clause
      }
```

Ví dụ:

```
ALTER TABLESPACE app_data
      MINIMUM EXTENT 2M;

ALTER TABLESPACE app_data
      DEFAULT STORAGE
      (INITIAL 2M NEXT 2M
      MAXEXTENTS 999);
```

9.7.5. Chuyển đổi chế độ ONLINE và OFFLINE

User chỉ có thể truy xuất vào tablespace nếu nó đang ở trạng thái online. Trong một vài trường hợp, quản trị viên database có thể thay đổi trạng thái database thành offline với mục đích:

- Di chuyển các data files tới vị trí khác
- Chỉ cho phép user truy xuất phần dữ liệu còn lại trong database.

Để chuyển đổi chế độ `ONLINE` và `OFFLINE`, ta có thể thực hiện câu lệnh SQL sau:

```
ALTER TABLESPACE tablespace
      {ONLINE
      |OFFLINE [NORMAL|TEMPORARY|IMMEDIATE]
      }
```

Chế độ OFFLINE

Oracle server không cho phép thực hiện câu lệnh SQL đối với các đối tượng có trong tablespace đã được `OFFLINE`.

Oracle server thực hiện checkpoint đối với tất cả các data files thuộc tablespace trước khi chuyển sang chế độ `OFFLINE`.

Mỗi khi database được mở, quản trị viên database có thể chuyển chế độ offline cho tất cả các tablespace ngoại trừ SYSTEM và các tablespace tương ứng với các active rollback segments hay temporary segments.

Trong OEM ta có thể thực hiện theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Tablespace.
3. Chọn tablespace tương ứng.

4. Trong General page, đặt chế độ trong Offline radio button.
5. Bấm nút Apply

9.7.6. Di chuyển các data file

Tuỳ thuộc kiểu tablespace, ta có thể di chuyển các data files theo các phương thức khác nhau.

Lệnh ALTER TABLESPACE

Lệnh này chỉ áp dụng cho các tablespace không phải là SYSTEM tablespace, và không chứa rollback segments hay temporary segments.

Câu lệnh:

```
ALTER TABESPACE tablespace
      RENAME DATAFILE 'filename'[, 'filename']...
      TO 'filename'[, 'filename']...
```

Ví dụ:

```
ALTER TABLESPACE app_data RENAME
      DATAFILE '/DISK4/app01.dbf' TO
      '/DISK5/app01.dbf';
```

Ta thực hiện theo các bước sau:

1. Chuyển chế độ offline cho tablespace.
2. Di chuyển các data files tương ứng bằng lệnh của hệ điều hành.
3. Thực hiện lệnh `ALTER TABLESPACE RENAME DATAFILE`.
4. Chuyển lại chế độ online cho tablespace đó.
5. Sử dụng lệnh của hệ điều hành để xoá data file cũ nếu cần thiết.

Lệnh ALTER DATABASE

Lệnh này chỉ áp dụng cho các tablespace không là SYSTEM và không chứa rollback segments hay temporary segments.

Câu lệnh:

```
ALTER DATABASE [database]
      RENAME FILE 'filename'[, 'filename']...
      TO 'filename'[, 'filename']...
```

Ví dụ:

```
ALTER DATABASE RENAME FILE
      '/DISK1/system01.dbf' TO
      '/DISK2/system01.dbf';
```

Ta thực hiện theo các bước sau:

1. Shutdown database.
2. Di chuyển data files bằng lệnh của hệ điều hành.
3. Mount lại database.
4. Thực hiện lệnh `ALTER DATABASE RENAME FILE`.
5. Mở lại database.

Trong OEM ta làm như sau

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Tablespace rồi chọn data file tương ứng.
3. Trong phần General page, thay đổi lại các thông tin thích hợp.
4. Bấm nút Apply.

9.7.7. Tablespace chỉ đọc

Sử dụng lệnh SQL để thiết lập các chế độ này.

Cú pháp:

```
ALTER TABLESPACE tablespace
     READ{ONLY|WRITE}
```

Ví dụ:

```
ALTER TABLESPACE app_data READ ONLY;
```

Trong OEM ta thực hiện theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Tablespace.
3. Chọn tablespace.
4. Chọn Tablespace—>Make Read Only.
5. Bấm nút OK.

Thiết lập chế độ chỉ đọc cho tablespace

Ta có thể thiết lập chế độ chỉ đọc cho tablespace khi nó đảm bảo một số điều kiện sau:

- Tablespace phải đang online
- Không có transaction nào xảy ra đối với tablespace đó
- Tablespace không chứa các rollback segments
- Hiện thời không có online backup trên tablespace

9.7.8. Huỷ tablespace

Trong một vài trường hợp ta có thể huỷ tablespace khỏi database.

Việc này có thể thực hiện bởi câu lệnh SQL sau:

```
DROP TABLESPACE tablespace
     [INCLUDING CONTENTS [CASCADE CONSTRAINTS]]
```

Với

`tablespace` tên của tablespace được huỷ

`INCLUDING CONTENTS`

huỷ luôn các segment có trong tablespace

`CASCADE CONSTRAINTS`

Huỷ luôn cả các ràng buộc liên quan tới các bảng bên ngoài có tham chiếu duy nhất tới các bảng thuộc tablespace bị huỷ

Ví dụ:

```
DROP TABLESPACE app_data
     INCLUDING CONTENTS;
```

Trong OEM ta thực hiện theo các bước sau

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Tablespace chọn tablespace tương ứng.
3. Chọn Tablespace—>Remove.
4. Bấm nút OK.

9.8.THÔNG TIN VỀ CÁC TABLESPACES

Một số views thông tin chung

Tên tham số	Diễn giải
DBA_TABLESPACES, USER TABLESPACES	Diễn giải của các tablespaces.
DBA_SEGMENTS, USER_SEGMENTS	Thông tin về segment có trong các tablespaces.
DBA_EXTENTS, USER_EXTENTS	Thông tin về data extents có trong các tablespaces.
DBA_FREE_SPACE, USER_FREE_SPACE	Thông tin về free extents có trong các tablespaces.
V$DATAFILE	Thông tin về tất cả các datafiles, bao gồm cả số hiệu tablespace và user sở hữu tablespace.
V$TEMPFILE	Thông tin về các tempfiles, bao gồm cả số hiệu tablespace và user sở hữu tablespace.
DBA_DATA_FILES	Hiển thị các datafiles thuộc các tablespaces.
DBA_TEMP_FILES	Hiển thị các tempfiles thuộc các temporary tablespaces.
V$TEMP_EXTENT_MAP	Thông tin của các extents trong các locally managed temporary tablespaces.
V$TEMP_EXTENT_POOL	Thông tin của các locally managed temporary tablespaces bao gồm: trạng thái của temporary space cached (vùng không gian đệm trung gian) được sử dụng bởi mỗi instance.
V$TEMP_SPACE_HEADER	Hiển thị vùng không gian used/free của mỗi tempfile.
DBA_USERS	Các tablespaces mặc định và temporary tablespaces của các users.
DBA_TS_QUOTAS	Hạn mức sử dụng tablespace của các users.
V$SORT SEGMENT	Thông tin về sort segment đối với mỗi instance.
V$SORT_USER	Vùng không gian sắp xếp trung gian được sử dụng bởi user và temporary/permanent tablespace.

9.8.1. Xem thông tin tablespace

Để xem thông tin về tablespace, ta có thể lấy trong data dictionary views. View DBA_TABLESPACES lưu trữ các thông tin này.

Một số thông tin quan tâm:

Tên tham số	Diễn giải
TABLESPACE_NAME	Tên tablespace
NEXT_EXTENT	Kích thước của các extent mở rộng tính theo bytes
MAX_EXTENTS	Số lượng tối đa các extents trong một segment
PCT_INCREASE	Phần trăm tăng trưởng kích thước của các extents
MIN_EXTENTS	Số lượng tối thiểu các extents trong một segment
STATUS	Trạng thái của tablespace là Online hay Offline
CONTENTS	Phân loại tablespace là permanent hay temporary

Ví dụ:

```
SVRMGR> SELECT tablespace_name, initial_extent, next_extent,
2 > max_extents, pct_increase, min_extlen
3 > FROM dba_tablespaces;

TABLESPACE_NAME  INITIAL_EX NEXT_EXT  MIN_EXTENT MAX_EXTENT PCT_I  MIN_EXTLEN
---------------  ---------- --------  ---------- ---------- -----  ---------
SYSTEM           1240       10240     1          121        50     0
RBS              10240      10240     1          121        50     0
TEMP             262144     262144    1          999        50     131072
DATA01           204800     204800    1          999        50     51200
4 rows selected.


SVRMGR> SELECT tablespace_name, contents,status
     2> FROM dba_tablespaces;

TABLESPACE_NAME   CONTENTS   STATUS
---------------   ---------  ------
SYSTEM            PERMANENT  ONLINE
RBS               PERMANENT  ONLINE
TEMP              TEMPORARY  ONLINE
DATA01            PERMANENT  ONLINE
4 rows selected.
```

9.8.2. Xem thông tin data files

Để xem thông tin về data files, ta có thể lấy trong dictionary views. View DBA_DATA_FILES lưu trữ các thông tin này.

Một số thông tin quan tâm:

Tên tham số	Diễn giải
FILE_NAME	Tên file (có kèm đường dẫn) tương ứng với datafile
TABLESPACE_NAME	Tên của tablespace ứng với datafile đó

BYTES	Dung lượng tính theo bytes của tablespace hiện thời
AUTOEXTENSIBLE	Chế độ tự động mở rộng dung lượng của datafile
MAXBYTES	Dung lượng tối đa
INCREMENT_BY	Chỉ số tăng tự động trong hệ thống

Ví dụ:

```
    SVRMGR> SELECT file_name, tablespace_name, bytes,
    2> autoextensible, maxbytes, increment_by
    3> FROM dba_data_files;

FILE_NAME            TABLESPACE_NAME  BYTES     AUT   MAXBYTES  INCREMENT_BY
---------            ---------------  --------  ----  --------  ----------
/DISK1/system01.dbf  SYSTEM           31457280  NO    0         0
/DISK2/rbs01.dbf     RBS              5242880   NO    0         0
/DISK3/temp01.dbf    TEMP             5242880   NO    0         0
/DISK4/data01.dbf    DATA01           5242880   NO    0         0
/DISK5/data02.dbf    DATA01           512000    YES   15728640  512
5 rows selected.
```

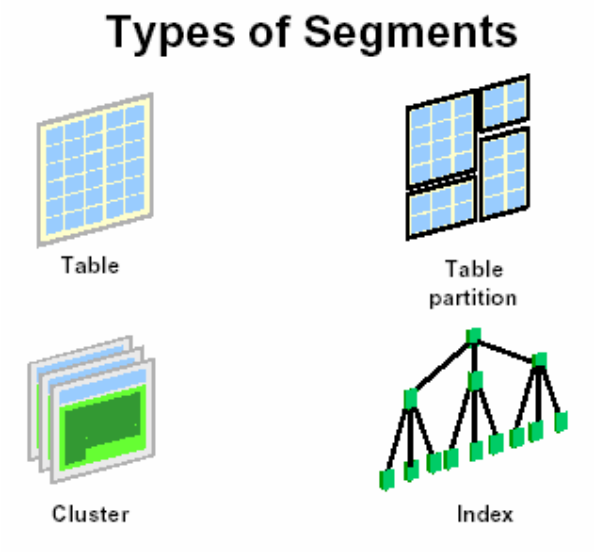
Chương 10. CẤU TRÚC LƯU TRỮ

10.1.CÁC LOẠI SEGMENTS

Segments là các vùng không gian của các objects (đối tượng) trong database. Dưới đây, ta sẽ xem xét một số loại segments cụ thể.

10.1.1. Table

Table (bảng), là nơi lưu giữ dữ liệu trong database. Dữ liệu trong một table được lưu giữ không theo một thứ tự bắt buộc. Các dữ liệu trong một table thuộc loại nonpartitioned (không phân khu) sẽ phải lưu giữ trong cùng một tablespace.



Hình vẽ 37. Các loại segments

10.1.2. Table partition

Có thể có một số table trong database có số lượng truy cập lớn và đồng thời. Khi đó, dữ liệu trong table đó sẽ được lưu thành nhiều partition (phân khu), mỗi partition có thể nằm trên các tablespace khác nhau. Oracle server hỗ trợ việc phân chia này bằng các giá trị khoá. Khi một table được phân khu, mỗi partition đó được xem như một segment.

10.1.3. Cluster

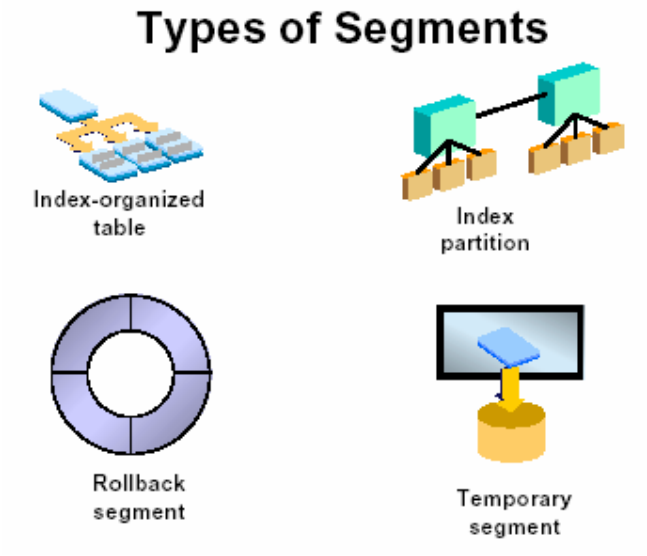
Các dòng dữ liệu trong một cluster được lưu trữ theo các giá trị của trường khoá (key column). Một cluster có thể chứa một hay nhiều tables và nó được xem là một kiểu đoạn dữ liệu (type of data segment). Các tables trong một cluster thuộc về cùng một đoạn và có chung các tính chất lưu trữ.

10.1.4. Index

Tất cả các đầu mục (entries) ứng với một index cụ thể được lưu trữ trong một index segment. Một table có tới bao nhiêu indexes, thì sẽ có bấy nhiêu index segments được sử dụng. Mục đích của segment này là tìm kiếm và định vị các dòng dữ liệu trong một table dựa trên một khoá được chỉ ra.

10.1.5. Index-Organized Table

Trong một index-organized table, các dữ liệu trong một index được lưu trữ dựa vào giá trị khoá. Một index-organized table không cần thiết đến một table dùng để tìm kiếm (lookup), các dữ liệu có thể được trả về ngay trực tiếp từ cây index (index tree).



Hình vẽ 38. Các loại segments (tiếp theo)

10.1.6. Index Partition

Một index có thể được partitioned (phân khu) và trải rộng trên nhiều tablespaces khác nhau. Khi đó, mỗi partition của một index sẽ tương ứng với segment (đoạn) và không được phép nằm dài trên nhiều tablespaces. Mục đích chính của việc sử dụng index partition là để giảm thiểu những tranh chấp vào ra I/O.

10.1.7. Rollback Segment

Rollback segment được sử dụng trong transaction (giao dịch) để tạo các thay đổi trong database. Trước khi thay đổi các dữ liệu hay các index blocks, các giá trị cũ sẽ được lưu giữ vào rollback segment. Việc làm này cho phép user có thể phục hồi lại các thay đổi.

10.1.8. Temporary Segment

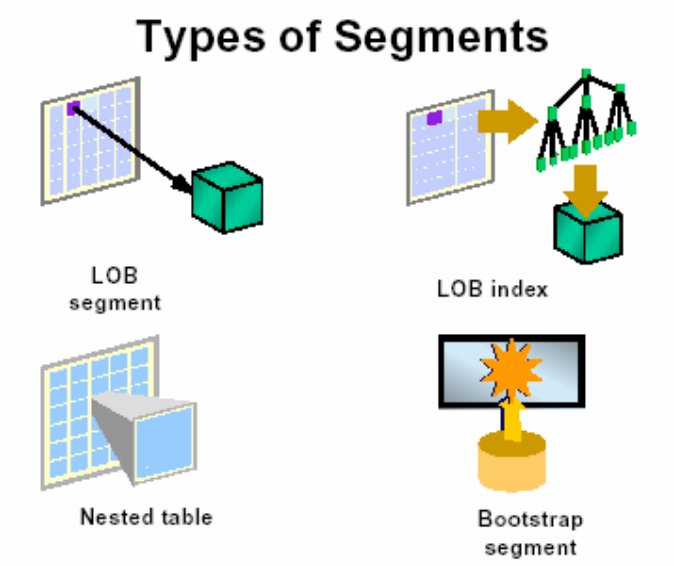
Khi một user thực hiện các lệnh như `CREATE INDEX`, `SELECT DISTINCT`, và `SELECT GROUP BY`, Oracle sẽ cố gắng thực hiện công việc sắp xếp ngay trong bộ nhớ. Khi công việc sắp xếp cần đến nhiều không gian hơn, các kết quả này sẽ được ghi trực tiếp lên đĩa. Temporary segments sẽ được dùng đến trong trường hợp này.

10.1.9. LOB Segment

Khi một hay nhiều cột trong table lưu giữ các đối tượng lớn (large objects - LOBs) như các văn bản tài liệu, hình ảnh, hay videos. Các cột chứa dữ liệu lớn này sẽ được Oracle server lưu giữ trong các segments riêng được biết đến như là LOB segments. Table sẽ chỉ lưu giữ các giá trị dùng để định vị, xác định nơi lưu giữ các dữ liệu LOB tương ứng.

10.1.10. LOB Index

Một LOB index segment được tạo ngầm định mỗi khi LOB segment được tạo lập. Các tính chất lưu giữ của LOB index có thể được quy định bởi quản trị viên database. Mục đích của việc sử dụng LOB index segment là cho phép tìm kiếm các giá trị cụ thể trong cột dữ liệu loại LOB.



Hình vẽ 39. Các loại segments (tiếp theo)

10.1.11. Nested Table

Cột dữ liệu trong table có thể được tạo lập từ một user-defined table (bảng do người dùng định nghĩa). Trong trường hợp này, bảng dữ liệu tương ứng với phần tử thuộc cột dữ liệu (inner table), được biết đến như một nested table và được lưu giữ trong một segment riêng biệt.

10.1.12. Bootstrap Segment

Bootstrap segment, được biết đến như một cache segment, được tạo bởi file script *sql.bsq* sau mỗi khi database được tạo. Segment giúp cho việc khởi tạo data dictionary cache mỗi khi database được mở bởi một instance. Dữ liệu trong bootstrap segment không thể xem hay sửa chữa, cập nhật được. Quản trị database cũng không cần thiết phải quan tâm tới segment này.

10.2.QUẢN LÝ EXTENTS

10.2.1. Cấp phát và thu hồi các extents

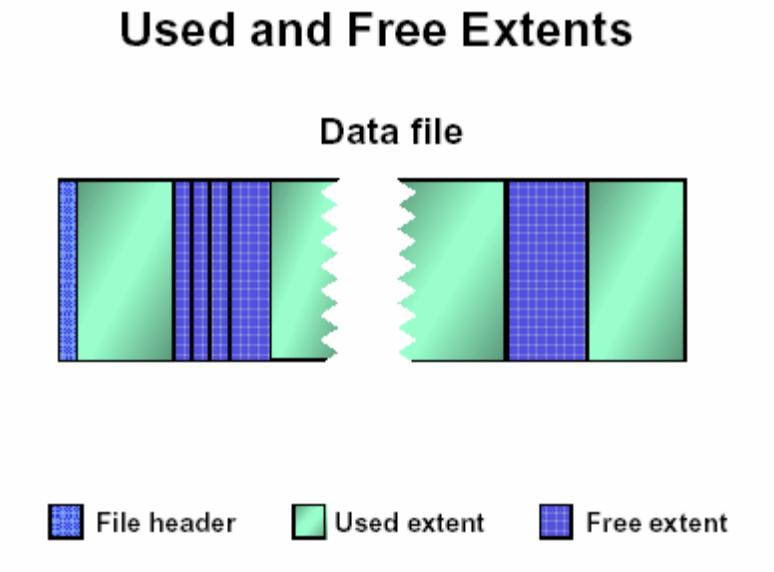
Việc cấp phát các extent xảy ra mỗi khi segment được tạo mới, được mở rộng hay bị thay đổi (altered).

Và nó sẽ bị thu hồi khi segment bị huỷ, bị thay đổi, bị cắt bớt (truncated). Riêng đối với các rollback segments, các extent có thể bị tự động thu hồi.

10.2.2. Sử dụng và giải phóng các extent

Khi một tablespace được tạo, các data files thuộc tablespace sẽ chứa các phần thông tin sau:

- Header block, tương ứng với block đầu tiên của file
- Phần còn lại của data file là các phần còn trống



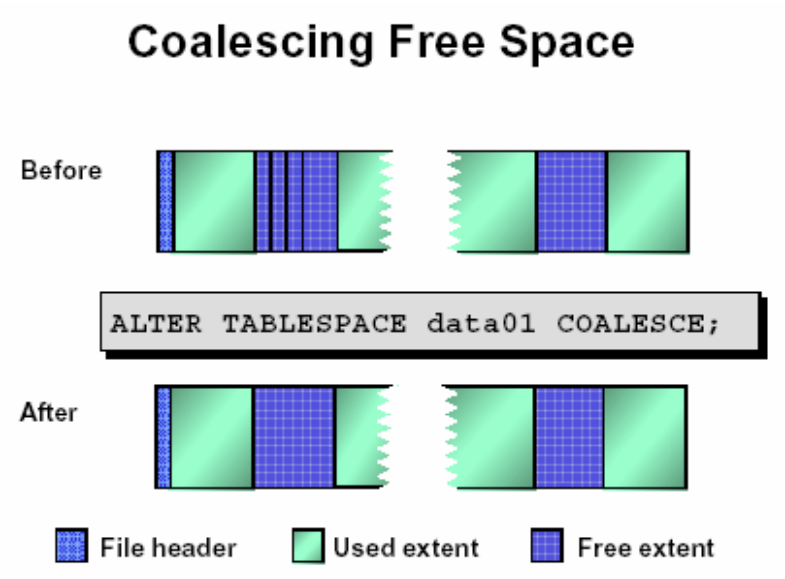
Hình vẽ 40. Sử dụng và giải phóng các extents

Mỗi khi segments được tạo lập, nó sẽ được cấp phát một vùng không thích hợp từ những extents còn trống trong tablespace. Segment sẽ cố gắng sử dụng nhiều nhất các vùng không gian liên tiếp nhau. Sau khi cấp phát, extent đó sẽ được xem là *used extent* (extent đã được sử dụng). Khi các segments giải phóng vùng không gian, các extents tương ứng với nó sẽ được giải phóng và đưa vào vùng free extents (extents rỗi) của tablespace. Với việc cấp phát và giải phóng các extents có thể gây nên hiện tượng phân đoạn vùng dữ liệu trong các data files của tablespace.

10.2.3. Kết hợp các vùng không gian trống

Ta có thể thực hiện việc kết hợp các vùng không gian trống liên tiếp nhau mỗi khi các extents trong cùng một tablespace được giải phóng. Điều này rất dễ xảy ra, ví dụ: khi có hai table bị huỷ (dropped). Các extents trống này có thể được kết hợp lại thành một extent trong các điều kiện:

- Khi tiến trình SMON khởi tạo một *space transaction* để kết hợp các extents trống.
- Khi Oracle server cần phải cấp phát vùng trống mà nó cần tới lượng không gian trống lớn hơn không gian của một extent.
- Kết hợp theo yêu cầu của quản trị viên database.



Hình vẽ 41. Kết hợp các vùng không gian trống

Lưu ý

Tiến trình `SMON` sẽ chỉ kết hợp các extent trong cùng tablespaces khi mà `PCTINCREASE` lớn hơn 0. Trong storage clause mặc định của tablespaces, đặt `PCTINCREASE=1` khi đó các user objects có thể được tự động kết hợp các vùng trống mỗi khi nó được giải phóng.

Yêu cầu kết hợp vùng trống

View `DBA_FREE_SPACE_COALESCED` được dùng để xem tablespace nào có các extents rỗng có thể kết hợp được với nhau. Sử dụng câu lênh truy vấn sau đây để lấy các thông tin:

```
SVRMGR> SELECT tablespace_name, total_extents,
2> percent_extents_coalesced
3> FROM dba_free_space_coalesced
4> WHERE percent_extents_coalesced <> 100;
TABLESPACE_NAME TOTAL_EXTE PERCENT_EX
--------------- ---------- ----------
RBS                      3         33
DATA01                   9         22
2 rows selected.
```

Thực hiện kết hợp các vùng không gian trống trong tablespace bằng lệnh dưới đây:

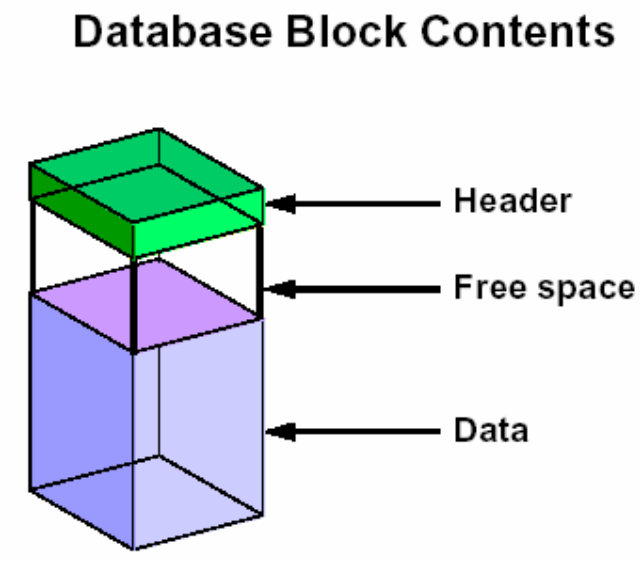
```
ALTER TABLESPACE tablespace COALESCE;
```

Trong OEM, ta thực hiện theo các bước sau

1. Sử dụng công cụ Oracle Tablespace Manager.
2. Chuyển tới nút Expand Tablespaces.
3. Chọn tablespace tương ứng.
4. Chọn mục Tools—>Coalesce Free Extents.

10.3.BLOCK DỮ LIỆU

10.3.1. Cấu trúc của block dữ liệu



Hình vẽ 42. Cấu trúc của Block dữ liệu

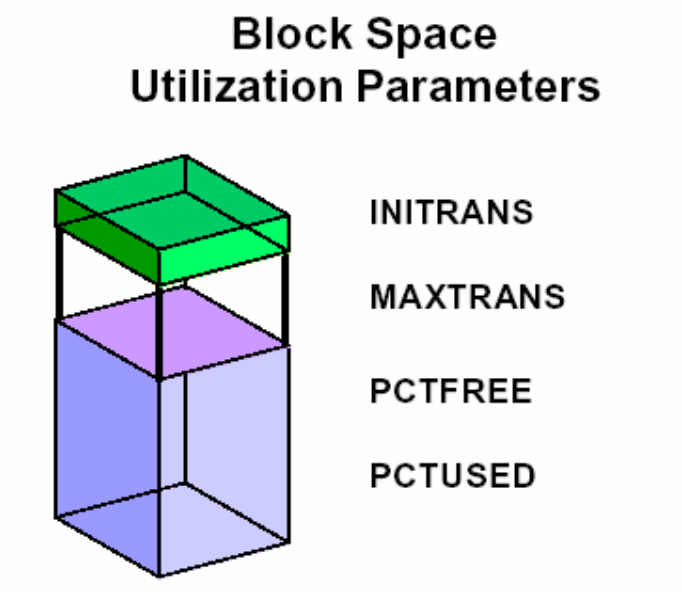
Các Blocks dữ liệu của Oracle được cấu thành từ các phần sau:

- Block header (vùng đầu): Header chứa địa chỉ của block dữ liệu, thông tin về table directory, row directory, và các transaction slots. Thông tin trong Block headers sẽ tăng dần theo hướng từ trên xuống dưới.
- Data space (vùng dữ liệu): Các dòng dữ liệu được nạp vào block theo hướng từ dưới lên.
- Free space (vùng trống): Vùng trống trong block là vùng nằm giữa vùng header và vùng không gian lưu trữ dòng dữ liệu. Ban đầu, vùng không gian trống là liên tiếp với nhau.Tuy nhiên sau một thời gian sử dụng, vùng không gian trống trong một block có thể bị phân đoạn do việc xoá và cập nhật, thay đổi các dòng dữ liệu. Để giải quyết vấn đề này, Oracle server cho phép thực hiện kết hợp các phân đoạn dữ liệu.

10.3.2. Các tham số sử dụng không gian trong block

Các tham số sử dụng không gian trong block được dùng để điều khiển việc sử dụng vùng không gian dữ liệu và index trong các segments.

Các tham số điều khiển song song



Hình vẽ 43. Các tham số sử dụng không gian trong block

Các tham số `INITRANS` và `MAXTRANS` chỉ ra số lượng khởi tạo, số lượng lớn nhất các transaction slots, được tạo trong mỗi index block hay data block. Các transaction slots được sử dụng để lưu giữ các thông tin về các transactions làm thay đổi các block tại cùng một thời điểm. Mỗi transaction chỉ sử dụng một transaction slot.

`INITRANS` được gán giá trị mặc định bằng 1 cho data segment, và 2 cho index segment.

`MAXTRANS` được gán giá trị mặc định là 255, dùng để tạo ngưỡng đối với các transactions đồng thời có làm thay đổi các block dữ liệu hay index block. Khi thiết lập giá trị này, vùng không gian cho các transaction slots sẽ được đảm bảo để có thể thực hiện các transaction một cách hiệu quả.

Tham số điều khiển vùng lưu trữ dữ liệu

`PCTFREE` trong một data segment chỉ lượng phần trăm vùng trống trong mỗi data block để dành cho việc tăng lên của dữ liệu do việc cập nhật các dòng dữ liệu trong block. Theo mặc đinh, `PCTFREE` là 10 phần trăm.

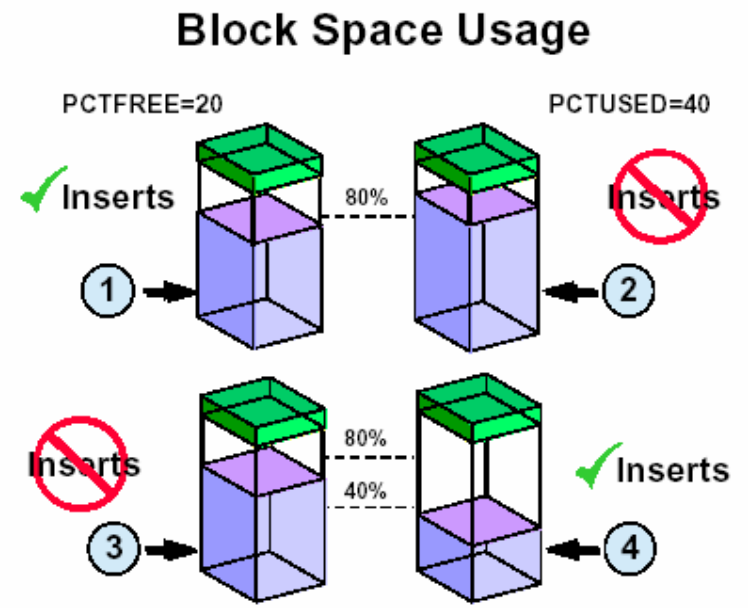
`PCTUSED` trong một data segment chỉ lượng phần trăm tối thiểu của vùng không gian sử dụng, theo đó Oracle Server lưu giữ các block dữ liệu của table. Một block sẽ được nạp lại vào free list (danh sách trống) mỗi khi `PCTUSED` giảm xuống. Free list của một segment là danh sách các blocks sẵn dùng cho việc cấp phát mỗi khi có dòng dữ liệu được insert. Theo mặc định mỗi free list sẽ được tạo tương ứng với mỗi segment. Tham số `FREELISTS` xác định số lượng free list. Mặc định, `PCTUSED` bằng 40 phần trăm.

`PCTFREE` và `PCTUSED` được tính toán theo phần trăm vùng không gian của dữ liệu, tức là vùng không gian của Block còn lại trừ đi vùng không gian header.

10.3.3. Sử dụng không gian trong block

Để cụ thể, ta theo dõi các bước việc sử dụng các vùng không gian trong block đối với một table có `PCTFREE=20` và `PCTUSED=40`:

- **Phase 1**: Các dòng dữ liệu được nạp vào block cho tới khi đủ 80% (`100-PCTFREE`). Lúc này, ta không thể insert thêm dữ liệu vào Block.



Hình vẽ 44. Sử dụng vùng không gian trong block

- **Phase 2**: 20% không gian còn lại sử dụng cho việc tăng kích thước của các dòng dữ liệu do việc cập nhật lại các dòng dữ liệu này.
- **Phase 3**: Khi xoá dòng dữ liệu trong block, vùng không gian trống trong block sẽ tăng lên. Tuy nhiên, lúc này ta vẫn chưa thể insert dữ liệu vào block được.
- **Phase 4**: Khi vùng trống trong block đạt tới mức `PCTUSED`, ta lại có thể insert dữ liệu vào Block. Ta lại bắt đầu từ bước 01.

10.3.4. Phân loại mức độ phân đoạn đối với từng loại segment

Tablespace	Phân loại sử dụng	Mức độ phân đoạn
SYSTEM	Data dictionary	Không xảy ra
TOOLS	Applications	Rất ít
DATAn	Data segments	Ít
INDEXn	Index segments	Ít
RBSn	Rollback segments	Nhiều
TEMPn	Temporary segments	Rất nhiều*

Ký hiệu * có nghĩa là chỉ đúng với các tablespaces thuộc loại `PERMANENT`

Hiện tượng phân đoạn dữ liệu xảy ra với mức độ khác nhau đối với các loại segments khác nhau. Oracle khuyến cáo nên lưu trữ dữ liệu trên nhiều tablespaces khác nhau để giảm thiểu việc sử dụng lãng phí các vùng không gian.

Phân loại các Objects và phân đoạn

Các loại objects khác nhau được liệt kê dưới đây theo mức độ tăng dần về phân đoạn:

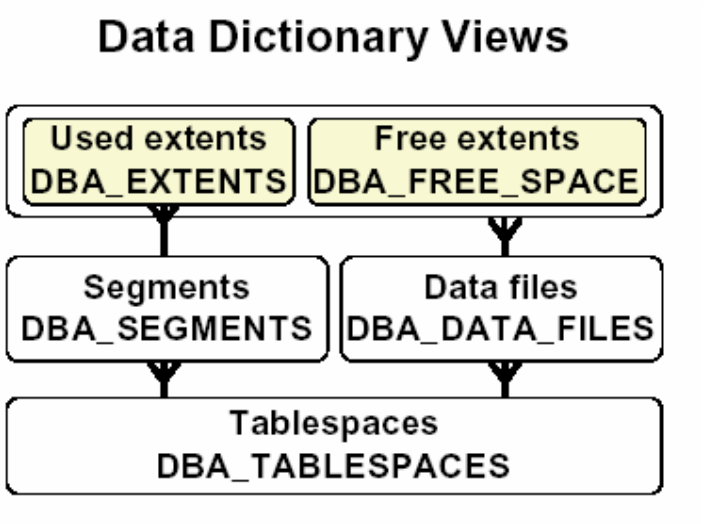
- Các data dictionary objects, ngoại trừ các audit table (bảng kiểm tra), đều không bao giờ bị dropped hay truncated. Vì thế chúng không bị phân đoạn trong tablespace.
- Vùng không gian sử dụng cho việc lưu trữ các ứng dụng luôn được cấp phát và thu hồi trong quá trình tái cấu trúc lại bộ nhớ. Vì thế, các tables lưu trữ này có mức độ phân đoạn là thấp.
- Data segment và index segements được sử dụng cho việc lưu trữ dữ liệu người dùng thuộc các ứng dụng. Các đối tượng này thường có mức độ phân đoạn cao.
- Do các rollback segments được phân bổ lại extents một cách tự động, chúng dễ gây ra hiện tượng phân đoạn dữ liệu trong hệ thống.
- Temporary segments trong các permanent tablespaces thường xuyên được bị xảy ra hiện tượng phân đoạn.

10.4.THÔNG TIN VỀ CẤU TRÚC LƯU TRỮ

10.4.1. Các view lưu trữ thông tin

Thông tin về các tablespaces, data files, segments, và extents (thông tin về cả phần sử dụng lần phần còn trống) đều có thể lấy từ các từ điển dữ liệu.

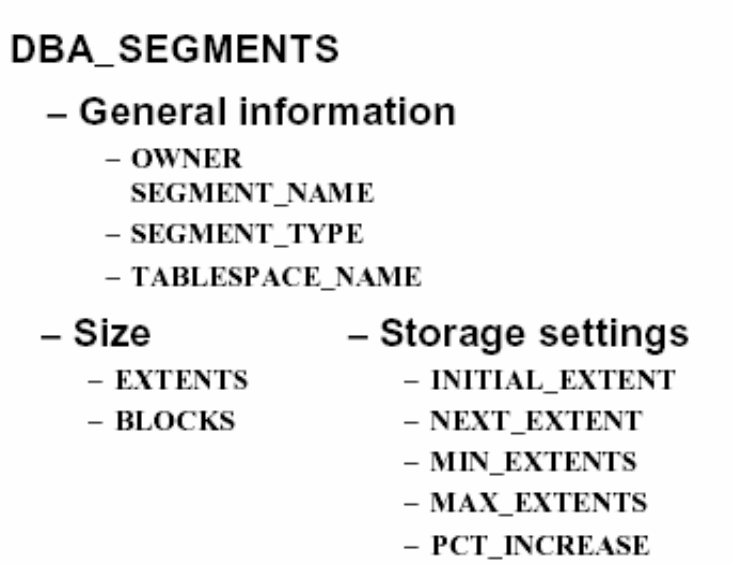
Thông tin về tablespace có thể được lưu trong `DBA_TABLESPACES`. Thông tin về các file dữ liệu của database được lưu trong `DBA_DATA_FILES`. Thông tin về các vùng trống trong các data file, vùng trống của extent được lưu trong `DBA_FREE_SPACE`. View `DBA_SEGMENTS` lưu giữ thông tin về các segment. Tương tự như vậy, `DBA_EXTENTS` lưu giữ thông tin về các extent.



Hình vẽ 45. Các views chứa thông tin về cấu trúc lưu trữ

10.4.2. Xem thông tin về các segments

Thông tin được lưu trong DBA_SEGMENTS.



Hình vẽ 46. Phân loại các thông tin chính có trong DBA_SEGMENTS

Ta có thể lấy thông tin về các segments theo các loại sau:

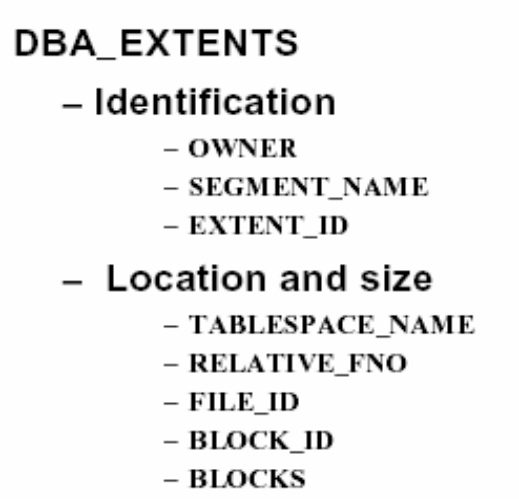
- Thông tin tổng hợp: User sở hữu, tên segment, loại segment, tên tablespace.
- Thông tin về kích cỡ: extents, blocks.
- Thông tin lưu trữ: INITIAL_EXTENT, NEXT_EXTENT, MIN_EXTENT, MAX_EXTENT, PCT_INCREASE

Ví dụ: Xem số lượng các extents và blocks được cấp phát cho từng segment do user SCOTT sở hữu.

```
SVRMGR> SELECT segment_name,tablespace_name,extents,blocks
2> FROM dba_segments
3> WHERE owner='SCOTT';

SEGMENT_NAME      TABLESPACE_NAME EXTENTS    BLOCKS
------------      --------------- --------- --------
EMP               DATA01          5         55
DEPT              DATA01          1         5
BONUS             DATA01          1         5
SALGRADE          DATA01          1         5
DUMMY             DATA01          1         5
5 rows selected.
```

10.4.3. Thông tin về các extents

Thông tin được lưu trong `DBA_EXTENTS`.



Hình vẽ 47. Phân loại các thông tin chính có trong DBA_EXTENTS

Ta có thể lấy thông tin về các extents theo các loại sau:

- Thông tin nhận dạng: User sở hữu, tên segment, mã hiệu extent
- Thông tin về kích cỡ và nơi đặt: `TABLESPACE_NAME, RELATIVE_FNO, FILE_ID, BLOCK_ID, BLOCKS`

Ví dụ: Xem thông tin chi tiết về các extents có trong một segment cho trước

```
SVRMGR> SELECT extent_id,file_id,block_id,blocks
2> FROM dba_extents
3> WHERE owner='SCOTT'
4> AND segment_name='EMP';

EXTENT_ID      FILE_ID BLOCK_ID BLOCKS
---------      ------- -------- ------
0              4       2        5
1              4       27       5
2              4       32       10
3              4       42       15
4              4       57       20
5 rows selected.
```

10.4.4. Thông tin về các vùng trống

Thông tin về các vùng trống được lưu trong DBA_FREE_SPACE.

Hình vẽ 48. Phân loại các thông tin chính có trong DBA_FREE_SPACE

View này chứa các thông tin về

Ví dụ:

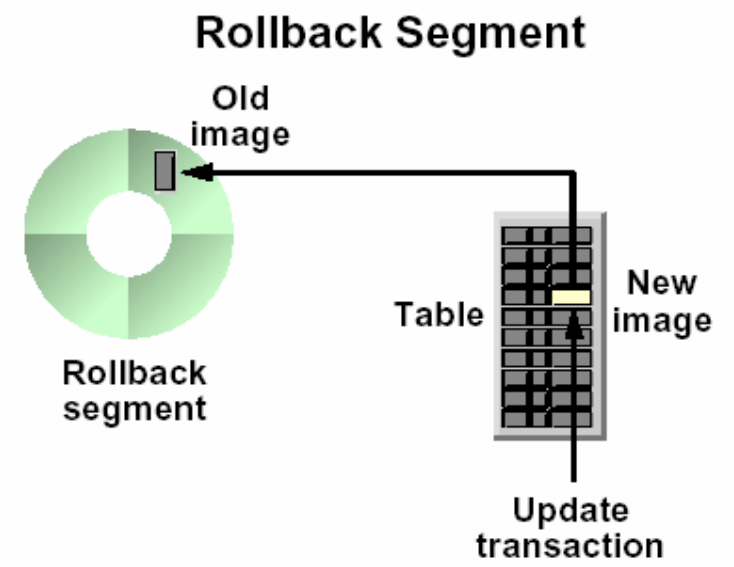
```
SVRMGR> SELECT tablespace_name, count(*),
2> max(blocks), sum(blocks)
3> FROM dba_free_space
4> GROUP BY tablespace_name;
TABLESPACE_NAME COUNT(*) MAX(BLOCKS SUM(BLOCKS
--------------- -------- ---------- ----------
DATA01                 2       1284       1533
RBS                    3       2329       2419
SORT                   1       1023       1023
SYSTEM                 1       5626       5626
TEMP                   1       2431       2431
5 rows selected.
```

Chương 11. QUẢN LÝ ROLLBACK SEGMENTS

11.1.GIỚI THIỆU ROLLBACK SEGMENTS

11.1.1. Khái niệm

Mỗi khi có sự thay đổi dữ liệu trong database, các dữ liệu cũ đều được lưu lại để có thể khôi phục lại trạng thái của dữ liệu trước khi thay đổi. Rollback segment được dùng để lưu trữ các giá trị cũ đó. Rollback segment lưu giữ các thông tin về block như block ID, và các dữ liệu đã sửa đổi của block.



Hình vẽ 49. Rollback segment

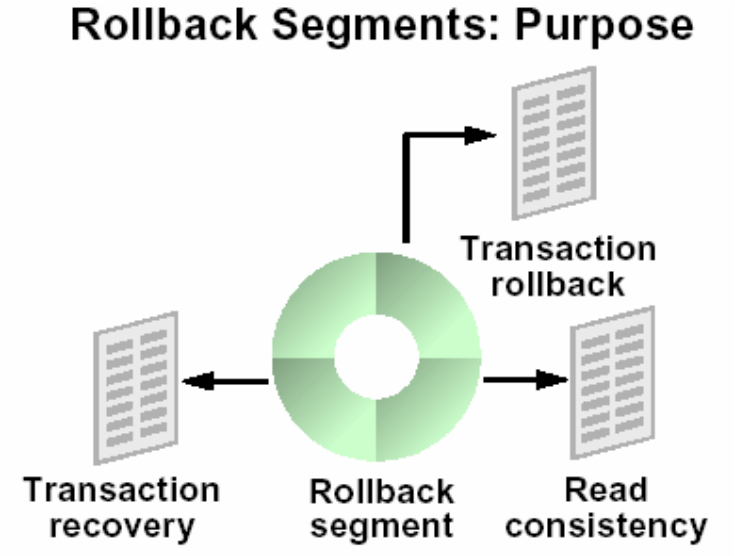
Phần đầu (header) của một rollback segment chứa một transaction table là nơi lưu giữ thông tin về các giao dịch hiện thời có sử dụng tới rollback segment đang xem xét. Mỗi transaction chỉ có thể sử dụng duy nhất một rollback segment để lưu giữ các dữ liệu dùng để khôi phục.

Nhiều transactions có thể đồng thời ghi dữ liệu lên rollback segment.

11.1.2. Mục đích sử dụng segment

Transaction rollback

Khi một transaction thực hiện các thay đổi đối với các dòng dữ liệu trong table, các giá trị ban đầu (old image) sẽ được lưu giữ vào rollback segment. Khi transaction đó được rolled back (lấy lại), các dữ liệu cũ lưu trong rollback segment sẽ được lấy ra và đè lên dữ liệu hiện tại trong block, phục hồi lại các giá trị nguyên thuỷ.



Hình vẽ 50. Mục đích của rollback segment

Phục hồi các Transaction

Trong trường hợp một instance gặp lỗi khi các transactions đang thực hiện, Oracle server cần phải khôi phục lại các dữ liệu chưa commit. Rollback trong trường hợp này được gọi là phục hồi dữ liệu. Việc này chỉ thực hiện được khi các thay đổi đối với các rollback segments đã được kết hợp bảo vệ bởi các redo log files.

Nhất quán trong việc đọc dữ liệu

Khi một thực hiện các transactions, các users trong database sẽ không thể thấy được các dữ liệu đã bị thay đổi mà chưa được commit bởi transactions. Các dữ liệu cũ lưu trong rollback segments sẽ vẫn được sử dụng để cung cấp cho các users khác nhằm đảm bảo nhất quán dữ liệu cho các user đó.

11.1.3. Phân loại rollback segment

SYSTEM Rollback Segment

SYSTEM rollback segment được tạo ngay trong SYSTEM tablespace mỗi khi một database được tạo lập. Rollback segment này chỉ được sử dụng đối với các thay đổi dữ liệu của các đối tượng nằm trong SYSTEM tablespace.

Non-SYSTEM Rollback Segments

Một database có thể có nhiều tablespaces và nên có ít nhất một non-SYSTEM rollback segment. Các non-SYSTEM rollback segment do quản trị viên database tạo lập có thể được sử dụng để lưu giữ các thay đổi trên các đối tượng có trong các non-SYSTEM tablespace khác. Có hai loại non-SYSTEM rollback segments.

- Private: Private rollback segments là các segments được sử dụng riêng cho mỗi instance.
- Public: Public rollback segments là một phần của rollback segments có trong database. Public rollback segments có thể được sử dụng bởi Oracle Parallel Server.

11.2.SỬ DỤNG ROLLBACK SEGMENT

11.2.1. Sử dụng rollback segment trong các transaction

Cấp phát các Rollback Segment

Đối với các transaction phải xử lý một khối lượng lớn các dữ liệu, ta cần gán transaction này với một rollback segment riêng chuyên làm nhiệm vụ lưu giữ các trạng thái ban đầu của dữ liệu.

Chú ý gán rollback segments cho một transaction:

- Lường trước khối lượng thông tin trong transaction cần rollback phù hợp (fit) với kích thước của vùng trống (extents) hiện thời của rollback segment.
- Cấp phát vừa đủ các vùng trống và không cần cấp phát bổ các vùng trống (extents) cho rollback segments đã được gán cho transaction vì điều này có thể dẫn đến việc giảm hiệu suất thực hiện của hệ thống.

Để gán một transaction cho một rollback segment một cách tường minh thì rollback segment đó cần phải đang ở trạng thái online. Cần thực lệnh lệnh `SET TRANSACTION USE ROLLBACK SEGMENT` trước khi thực hiện các lệnh trong transaction đó. Nếu trạng thái của rollback segment là offline hoặc câu lệnh `SET TRANSACTION USE ROLLBACK SEGMENT` không được đặt ở vị trí đầu tiên của transaction thì hệ thống sẽ phát sinh một lỗi.

Ví dụ: sử dụng lệnh gán rollback segment cho transaction tại thời điểm bắt đầu transaction:

```
SET TRANSACTION USE ROLLBACK SEGMENT large_rs1;
```

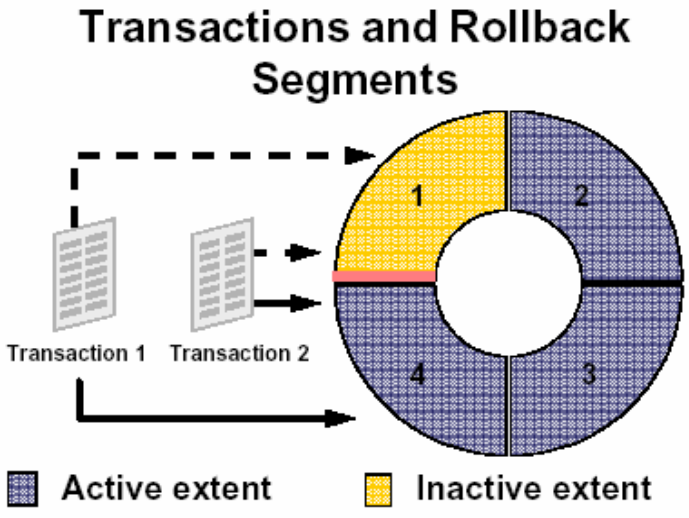
Sau khi transaction được commit, rollback segment này lại được Oracle đưa về trạng thái sẵn sàng sử dụng. Oracle sẽ tự động gán transaction tiếp theo cho một rollback segment bất kỳ nào đang còn rỗi (available) trừ phi transaction này lại được tiếp tục gán cho rollback segment bằng tay bởi user.

Sử dụng các extents

Các transactions sử dụng extents trong rollback segment theo một trình tự xoay vòng. Theo đó, transaction sẽ ghi dữ liệu thay đổi vào extent hiện thời, rồi tiếp tục chuyển tới các extent kế tiếp. Khi extent cuối cùng được sử dụng đầy, nó lại quay trở về extent 1.

Để rõ hơn, ta xem xét một ví dụ sau:

Có hai transaction cùng sử dụng một rollback segments có 04 extents.

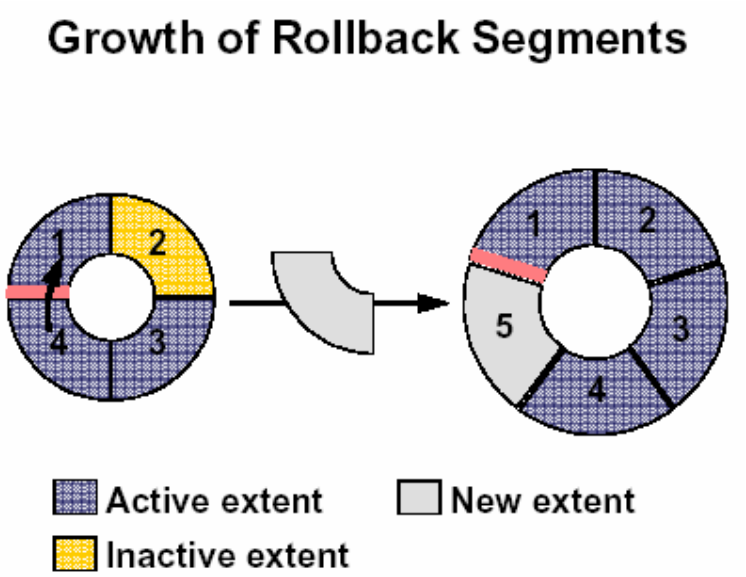


Hình vẽ 51. Sử dụng dữ liệu trong rollback segment

1. Tại thời điểm bắt đầu, giao dịch bắt đầu ghi dữ liệu vào Extent 3
2. Trong khi thực hiện, các transaction sẽ ghi dữ liệu vào Extent 3 cho tới khi đầy rồi tiếp tục chuyển sang ghi dữ liệu lên Extent 4.
3. Khi Extent 4 cũng đầy, nó tiếp tục lại quay trở lại ghi dữ liệu vào extent 1 nếu extent này ở trạng thái rỗi hoặc inactive. Một extent là rỗi hoặc inactive nếu hiện thời nó không bị sử dụng bởi bất kỳ một transaction nào.

11.2.2. Tăng trưởng đối với các rollback segments

Rollback segment có con trỏ để xác định extent đang làm việc. Khi extent làm việc đầy, con trỏ sẽ chuyển sang extent kế tiếp để thực hiện việc ghi dữ liệu. Cứ như vậy cho đến extent cuối cùng rồi lại quay trở về extent đầu tiên nếu extent này đang rỗi. Tuy nhiên, có nhiều khả năng extent đầu tiên này cũng đang không rỗi. Khi đó, con trỏ không thể nhảy cách mà bỏ qua extent 1 để chuyển sang extent 2 được. Để tiếp tục duy trì hoạt động cho transaction, cần phải bổ sung thêm một extent nữa vào sau extent cuối cùng. Việc này tạo nên sự tăng trưởng đối với các rolback segments. Việc tăng trưởng đối với các rolback segments sẽ tiếp tục xảy ra cho tới khi số lượng các extents tăng kịch khung quy định trong tham số `MAXEXTENTS`.



Hình vẽ 52. Tăng kích thước Rollback Segment

Sau khi rollback segments đã được tạo lập, quản trị viên database vẫn có thể thay đổi tham số lưu trữ của rollback segments. Để thay đổi, quản trị viên chỉ cần điều chỉnh các tham số `OPTIMAL` hay `MAXEXTENTS` cho phù hợp.

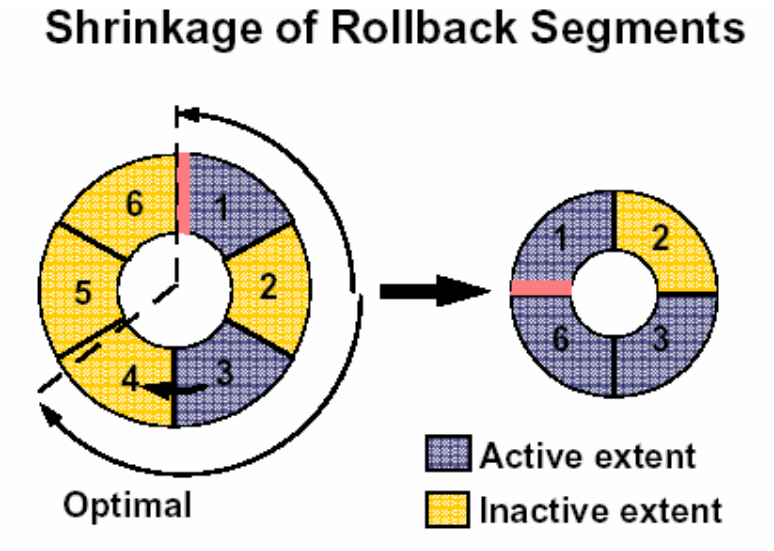
Ví dụ: Câu lệnh sau thay đổi số lượng tối đa các extents cấp phát cho rollback segments `RBS_01`.

```
ALTER ROLLBACK SEGMENT rbs_01
    STORAGE (MAXEXTENTS 120);
```

Với câu lệnh thay đổi này, ta cũng có thể điều chỉnh với rollback segment SYSTEM , bao gồm cả tham số `OPTIMAL`.

11.2.3. Tối ưu các rollback segments

Khi kết thúc hoặc commit các transaction, nó sẽ giải phóng vùng không gian đã sử dụng để lưu các dữ liệu dùng để phục hồi. Các extent trong rollback được đưa trở lại trạng thái inactive. Để tiết kiệm không gian lưu trữ trong rollback segment, ta có thể tối ưu lại rollback segment đó thông qua tham số `OPTIMAL`.



Hình vẽ 53. Giảm kích thước của Rollback segment

Oracle server sẽ thu hồi lại các extent đã cấp phát khi:

- Kích thước của rollback segment hiện tại được điều chỉnh tới giá trị của tham số `OPTIMAL`.
- Khi có nhiều hơn 02 extent rỗi liên tiếp cạnh nhau.

Một điều lưu ý là khi thu hồi lại các extent, Oracle server sẽ thu hồi extent chứa dữ liệu lâu nhất trước đó.

Ta có thể thực hiện giảm bớt kích thước của rollback segments thông qua việc sử dụng câu lệnh `ALTER ROLLBACK SEGMENT`. Lưu ý, khi này rollback segment được thu nhỏ nhất thiết phải đang ở trạng thái online.

Ví dụ: Thu nhỏ kích thước rollback segment `RBS1` bằng `100K`:

```
ALTER ROLLBACK SEGMENT rbs1 SHRINK TO 100K;
```

Câu lệnh phía trên thực hiện rút bớt kích thước của rollback segment tới kích thước như đã được chỉ ra. Tuy nhiên, việc rút gọn sẽ dừng lại khi có một extent không thể bị thu hồi do bất kỳ nguyên nhân nào.

11.3.QUẢN LÝ ROLLBACK SEGMENTS

11.3.1. Sử dụng rollback segment

Kích thước của rollback segment

Kích thước của rollback được xác định tuỳ thuộc vào hai yếu tố sau:

- Loại transaction được thực hiện (insert, update, delete, ...)
- Lượng dữ liệu được xử lý

Thông thường, việc thêm mới bản ghi vào bảng cần ít không gian lưu giữ thông tin phục hồi hơn là việc xoá dữ liệu khỏi bảng. Với thao tác thêm mới, chỉ cần lưu giữ ROWID vào rollback, trong khi thao tác delete lại cần phải lưu giữ toàn bộ dòng dữ liệu.

Đánh giá kích thước của rollback segment căn cứ theo transaction dài nhất có sử dụng rollback segment.

Số lượng các Extents

Với các rollback segment có quá nhiều các extents sẽ gây ra lãng phí không gian lưu trữ dữ liệu, để giảm bớt lãng phí, ta có thể điều chỉnh tham số `MINEXTENTS` cho phù hợp.

Oracle khuyến nghị, thông thường, `MINEXTENTS` nên đặt giá trị là 20.

11.3.2. Tạo rollback segment

Ta có thể tạo rollback segment thông qua câu lệnh SQL:

Cú pháp:

```
CREATE [PUBLIC] ROLLBACK SEGMENT rollback_segment
      [TABLESPACE tablespace]
      [STORAGE ([INITIAL integer[K|M]]
            [NEXT integer[K|M]]
            [MINEXTENTS integer]
            [MAXEXTENTS {integer|UNLIMITED}]
            [OPTIMAL {integer[K|M]|NULL}]
            )
      ]
```

Lưu ý:

- Một rollback segment có thể là `PUBLIC` hoặc `PRIVATE` (mặc định) việc gán này được thực hiện ngay lúc tạo và không thể thay đổi sau này.
- `MINEXTENTS>=2` đối với các rollback segment.
- `PCTINCREASE` được bỏ qua đối với các rollback segment và được gán bằng 0.
- `OPTIMAL`, nếu có chỉ ra thì không được nhỏ hơn giá trị kích thước khởi tạo của rollback segment được xác định trong tham số `MINEXTENTS`.
- `INITIAL=NEXT` để đảm bảo các extent trong rollback segment có cùng một kích thước.
- Không nên gán giá trị cho `MAXEXTENTS` là `UNLIMITIED` vì như vậy sẽ dẫn đến việc mở rộng các extent một cách không cần thiết.
- Nên đặt rollback segment trong một tablespace riêng biệt để giảm bớt hiện tượng phân đoạn dữ liệu trong database.

Ví dụ:

```
CREATE ROLLBACK SEGMENT rbs01
      TABLESPACE rbs
      STORAGE (
      INITIAL 100K NEXT 100K OPTIMAL 4M
      MINEXTENTS 20 MAXEXTENTS 100);
```

Trong Oracle Enterprise ta thực hiện theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chọn Rollback—>Create.
3. Trong phần General page, nhập vào tên, tablespace, và kiểu tương ứng. Chọn mục Online trong radio button.
4. Trong phần Extents, nhập vào các thông tin về rollback segments.
5. Bấm nút Create.

11.3.3. Thay đổi trạng thái của Rollback segments

Rollback segments có thể nhận một trong hai trạng thái `ONLINE/OFFLINE`

Khi rollback segment có trạng thái *online* thì nó sẵn sàng sử dụng cho các transactions, ngược lại, trạng thái *offline* cho biết nó không sẵn sàng cho các transactions. Thông thường, rollback segments là online và sẵn dùng cho các transactions.

Trong một số tình huống nhất định, ta cần đặt trạng thái online hay ofline đối với các rollback segments:

- Khi trạng thái của tablespace là online, nếu tablespace có chứa các rollback segments, ta sẽ không thể đặt trạng thái tablespace thành offline nếu có bất kỳ một transaction nào vẫn còn đang sử dụng các rollback segments thuộc tablespace đó. Để xử lý được tình huống này, ta cần thay đổi trạng thái của rollback segments thành offline để ngăn không cho sử dụng các rollback segments trước khi thay đổi trạng thái của tablespace là offline.
- Khi ta muốn drop (huỷ) các rollback segments, nhưng không thể thực hiện được do vẫn còn transactions đang sử dụng nó. Để xử lý được tình huống này, ta cần ngăn không cho sử dụng rollback segment thông qua việc đặt lại trạng thái rollback segments là offline.

Sau khi tạo mới một rollback segment, nó sẽ có trạng thái offline và chưa thể sử dụng ngay được. Để có thể sẵn dùng cho các transaction, rollback segment cần được chuyển trạng thái thành online thông qua câu lệnh `ALTER ROLLBACK SEGMENT`

Cú pháp:

```
ALTER ROLLBACK SEGMENT rollback_segment ONLINE | OFFLINE
```

Rollback segment sẽ có trạng thái online cho tới khi instance bị tắt (shutdown).

Đặt trạng thái online cho rollback segments ngay khi startup database

Để đảm bảo cho các rollback segments luôn nhận trạng thái online ngay khi khởi động (startup) database, ta cần chỉ rõ tên của rollback segments trong tham số `ROLLBACK_SEGMENTS` của parameter file.

Ví dụ:

```
ROLLBACK_SEGMENTS=(rbs01, rbs02, rbs03)
```

Lưu ý: Số lượng tối đa các rollback segment online đối với một instance được xác định bởi tham số `MAX_ROLLBACK_SEGMENT`.

Trong OEM ta có thể thực hiện theo các bước sau:

1. Chạy Oracle Storage Manager.

2. Chuyển tới nút Rollback Segments.
3. Chọn rollback segment tương ứng.
4. Chọn Rollback—>Place Online/ Take Offline.
5. Trong hộp thoại xác nhận, bấm nút Yes.

11.3.4. Instance sử dụng rollback segment

Để cụ thể, ta xem xét các bước thực hiện khi một instance sử dụng rollback segment

1. Instance sử dụng tất cả các rollback segments có tên trong phần tham số `ROLLBACK_SEGMENTS`.
2. Tham số `TRANSACTIONS` và `TRANSACTIONS_PER_ROLLBACK_SEGMENT` được sử dụng để tính toán số lượng rollback segments cần thiết cho một instance:

$$N = \frac{T}{TPR}$$

Với:

N	Số lượng rollback segment cần thiết
T	Giá trị tham số TRANSACTIONS
TRP	Giá trị tham số TRANSACTIONS_PER_ROLLBACK_SEGMENT

3. Trong trường hợp N nhỏ hơn hay bằng số lượng non-SYSTEM rollback segments có được, instance cũng sẽ không cần tới nhiều rollback segments hơn.
4. Khi giá trị của N lớn hơn hay bằng số các non-SYSTEM rollback segments dành cho instance, khi đó đòi hỏi phải sử dụng thêm cả các public rollback segments.

11.3.5. Điều chỉnh khả năng lưu trữ của rollback segment

Ta có thể điều chỉnh các tính chất lưu trữ của từng rollback segment thông qua lệnh `ALTER ROLLBACK SEGMENT`

Cú pháp:

```
ALTER ROLLBACK SEGMENT rollback_segment
[STORAGE ( [NEXT integer[K|M]]
      [MINEXTENTS integer]
      [MAXEXTENTS {integer|UNLIMITED}]
      [OPTIMAL {integer[K|M]|NULL}]
            )
]
```

Trong OEM ta thực hiện theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Rollback Segments.
3. Chọn rollback segment tương ứng.
4. Trong phần Extents page, đặt lại các tham số tương ứng.
5. Bấm nút Apply.

11.3.6. Giảm bớt độ rộng của rollback segment

Trong trường hợp tham số OPTIMAL được chỉ rõ, Oracle server sẽ có gắng thực hiện cấp phát và giải phóng vùng không gian dựa theo giá trị của tham số OPTIMAL. Ngược lại, ta có thể thực hiện cấp phát không gian thông qua lệnh trực tiếp:

```
ALTER ROLLBACK SEGMENT rollback_segment
      SHRINK [ TO integer [ K|M ]];
```

Trong trường hợp tham số integer không được chỉ rõ, Oracle sẽ giảm lượng không gian rollback segment về tới giá trị OPTIMAL

Trong OEM ta có thể thực hiện theo các bước sau:

1. Chạy Oracle Storage Manager.
2. Chuyển tới nút Rollback Segments.
3. Chọn rollback segment tương ứng.
4. Chọn Rollback—>Shrink.
5. Trong hộp thoại Shrink Rollback Segment, chọn Optimal Size rút gọn kích thước rollback segment theo kích thước đã được tối ưu. Hoặc chọn Size rồi nhập vào giá trị kích thước cho vùng không gian tương ứng.
6. Bấm nút OK.

11.3.7. Hủy bỏ rollback segment

Trong một số trường hợp không cần sử dụng các rollback segment, ta có thể hủy các rollback segment thông qua câu lệnh SQL:

```
DROP ROLLBACK SEGMENT rollback_segment;
```

Trong OEM, ta làm theo các bước sau:

1. Chọn Oracle Storage Manager.
2. Chuyển tới nút Rollback Segments.
3. Chọn rollback segment tương ứng.
4. Chọn Rollback—>Remove. Ta chỉ có thể hủy các rollback segment đã ở trạng thái offline.
5. Bấm nút Yes trong hộp thoại xác nhận.

11.3.8. Quản lý undo tự động

Khả năng quản lý undo tự động (Automatic Undo Management - AUM) là một đặc điểm khá mới của Oracle 9i. Cung cấp cơ chế tin cậy hơn cho DBA khi tạo, thay đổi kích thước và điều chỉnh rollback segments trong database. Theo đó, Rollback segments có thể được tạo, xoá hay điều chỉnh kích thước một cách tự động bởi instance.

Dữ liệu rollback data được quản lý nhờ vào undo tablespace.

Ví dụ: tạo undo tablespace

```
CREATE UNDO TABLESPACE "UNDO_TBS"
   DATAFILE '/u01/oradata/freeney9/undo_tbs01.ora' SIZE 100M
   AUTOEXTEND ON NEXT 10M MAXSIZE 700M
```

Một số tham số khởi tạo chính:

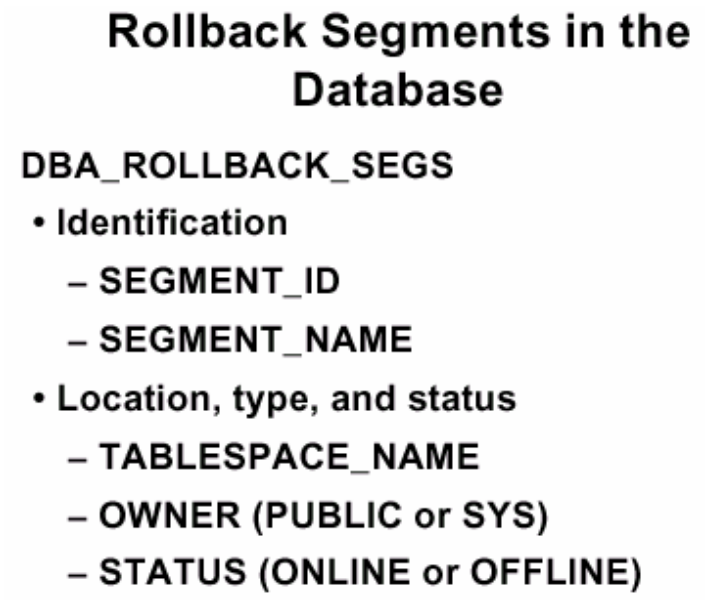
- `UNDO_MANAGEMENT (MANUAL / AUTO)`: Cho biết database có sử dụng cơ chế AUM hay không. Default = MANUAL
- `UNDO_TABLESPACE (valid tablespace)`: Chỉ rõ tên undo tablespace sử dụng.
- `UNDO_RETENTION (in seconds default=30)`: Cho biết thời gian trễ để thực hiện committed undo.
- `UNDO_SUPPRESS_ERRORS (TRUE / FALSE)`: Cho biết hệ thống có trả về exception hay không khi "SET TRANSACTION USE ROLLBACK SEGMENT" phát lỗi. Default = TRUE

11.4. THÔNG TIN VỀ CÁC ROLLBACK SEGMENT

Thông tin về các rollback segment được lưu giữ trong từ điển dữ liệu.

11.4.1. Xem thông tin chung về các rollback segment

Thông tin chung về rollback segment được lưu trong view `DBA_ROLLBACK_SEGS`.



Hình vẽ 54. Các thông tin chính về rollback segments

Các thông tin bao gồm:

- `SEGMENT_ID`: Mã hiệu của segment
- `SEGMENT_NAME`: Tên segment
- `TABLESPACE_NAME`: Tên tablespace chứa segment
- `OWNER (PUBLIC/SYS)`: Tên user sở hữu segment
- `STATUS (ONLINE/OFFLINE)`: Trạng thái của segment

Ví dụ: Xem thông tin chung về segment

```
SVRMGR> SELECT segment_name, tablespace_name, owner, status
2> FROM dba_rollback_segs;
SEGMENT_NAME                TABLESPACE_NAME  OWNER STATUS
```

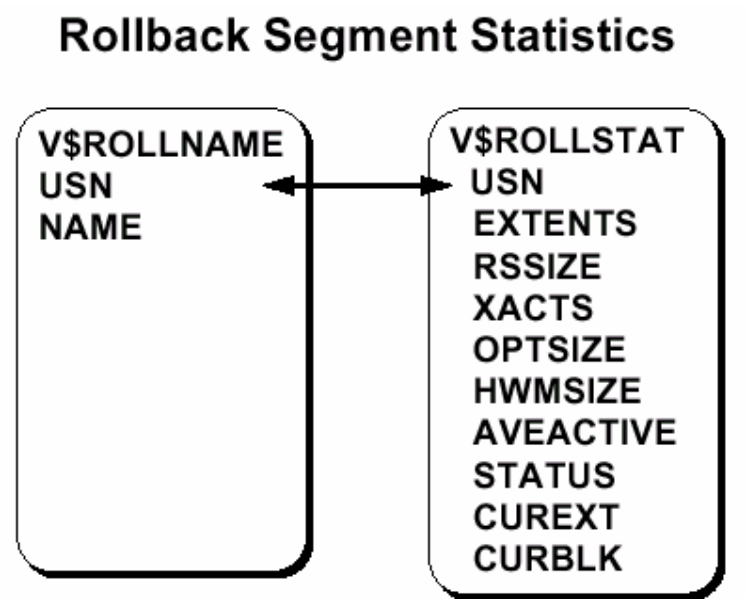
```
---------------              ---------------  ------     --------
-------
SYSTEM      SYSTEM           SYS         ONLINE
RBS1        RBS              SYS         ONLINE
RBS2        RBS              SYS         ONLINE
RBS3        RBS              SYS         OFFLINE
4 rows selected.
```

Dữ liệu trong cột `OWNER` nhận các giá trị:

- `SYS`: Rollback đó thuộc loại private
- `PUBLIC`: Rollback đó thuộc loại public

11.4.2. Xem thông tin thống kê về rollback segment

Ta lấy được các thông tin này từ các view `V$ROLLSTAT` và `V$ROLLNAME`.



Hình vẽ 55. Các thông tin thống kê về segments

Ví dụ: Xem các thông tin thông kê về segments

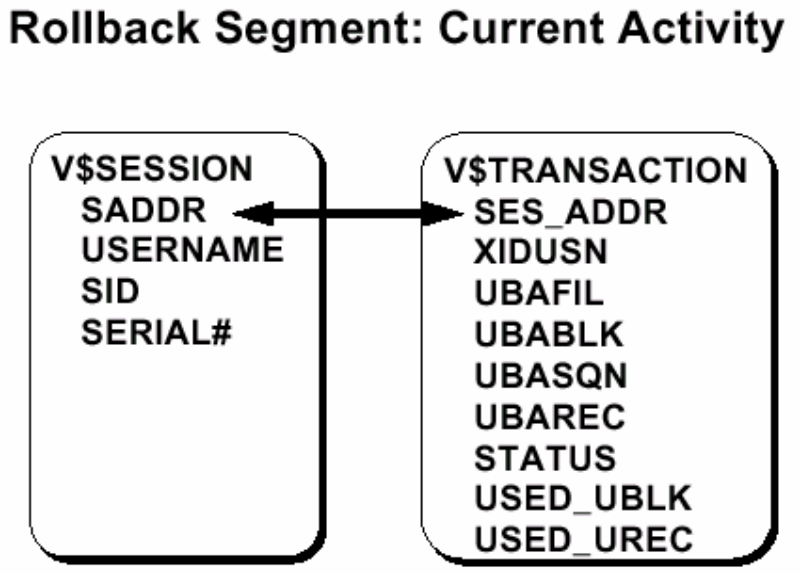
```
SVRMGR> SELECT n.name, s.extents, s.rssize, s.optsize,
2> s.hwmsize, s.xacts, s.status
3> FROM v$rollname n, v$rollstat s
4> WHERE n.usn = s.usn;

NAME   EXTENTS RSSIZE    OPTSIZE   HWMSIZE   XACTS  STATUS
-----  ------- --------  --------  --------  ------ ---------
SYSTEM 43      2199552             2199552   0      ONLINE
RBS1   20      202752    204800    417792    0      ONLINE
RBS2   4       38912               38912     0      PENDING
                                               OFFLINE
3 rows selected.
```

Diễn giải một số cột dữ liệu trong view V$ROLLSTAT

Tên cột	Diễn giải
USN	Là số hiệu của rollback segment (Rollback segment number)
EXTENTS	Số lượng các extents có trong rollback segment
RSSIZE	Kích thước của segment hiện thời tính theo đơn vị bytes
XACTS	Số lượng các transaction sử dụng rollback segment
OTPSIZE	Giá trị OPTIMAL của rollback segment
HWMSIZE	Hight warter mark; kích thước tối đa tính theo bytes, khi rollback segment tăng
AVEACTIVE	Kích thước của extent hiện thời,
STATUS	Trạng thái của rollback segment

11.4.3. Thông tin về rollback segment đang active

Ta có thể kết hợp thông tin trong hai bảng V$TRANSACTION và V$SESSION.



Hình vẽ 56. Thông tin về các thao tác trên các segments

Ví dụ:

```
SVRMGR> SELECT s.username, t.xidusn, t.ubafil,
2> t.ubablk, t.used_ublk
3> FROM v$session s, v$transaction t
4> WHERE s.saddr = t.ses_addr;

USERNAME     XIDUSN     UBAFIL     UBABLK     USED_UBLK
-----------  ---------- ---------  ---------- ----------
SYSTEM       2          2          7          1
SCOTT        1          2          163        1
```

```
2 rows selected.
```

Diễn giải một số cột dữ liệu

Tên cột	Diễn giải
SES_ADDR	Địa chỉ của session, lấy được từ V$SESION.SADDR
XIDUSN	Số hiệu của Rollback segment được sử dụng bởi transaction
UBAFIL, UBABLK, UBASQN,UBAREC	Vị trí hiện thời của rollback segment mà transaction sẽ ghi vào
USED_UBLK	Số hiệu block undo được tạo ra bởi transaction
START_UEXT, START_UBAFIL, START_UBABLK	Số hiệu của extent (file, block) thuộc rollback segment mà transaction bắt đầu ghi dữ liệu

11.5.CÁC VẤN ĐỀ LIÊN QUAN TỚI ROLLBACK SEGMENT

11.5.1. Thiếu không gian cho các transactions

Nguyên nhân

Do một transaction không được sử dụng nhiều rollback segments nên có thể xảy ra tình trạng thiếu vùng không gian cho các rollback segment và gây ra lỗi (ORA-01562). Nguyên nhân có thể là một trong các trường hợp sau:

- Không có đủ không gian trong tablespace (ORA-01560)
- Số lương các extents trong rollback segment đã đạt tới giá trị MAXEXTENTS và không thể bổ sung thêm các extent vào rollback segment (ORA-01628)

Giải pháp

Với lỗi ORA-01560:

- Mở rộng thêm các data files trong tablespace
- Đặt chế độ cho các data files là AUTOEXTEND
- Bổ sung mới data file vào tablespace

Với lỗi ORA-01628:

- Tăng tham số MAXEXTENTS của rollback segment
- Huỷ và tạo lại rollback segment với kích thước của extent lớn hơn

11.5.2. Lỗi đọc dữ liệu không đồng nhất

Nguyên nhân

Oracle server cố gắng đảm bảo các câu lệnh sẽ chỉ xử lý trên các dữ liệu đã được commit. Vì thế, các dữ liệu chưa commit sẽ không được sử dụng. Trong trường hợp Oracle server không tạo được các bản lưu giá trị cũ các dữ liệu (read-consistent image of data), user sẽ nhận được lỗi ORA-01555 snapshot too old. Lỗi này xảy ra khi transaction thay đổi các dữ liệu đã được commit và:

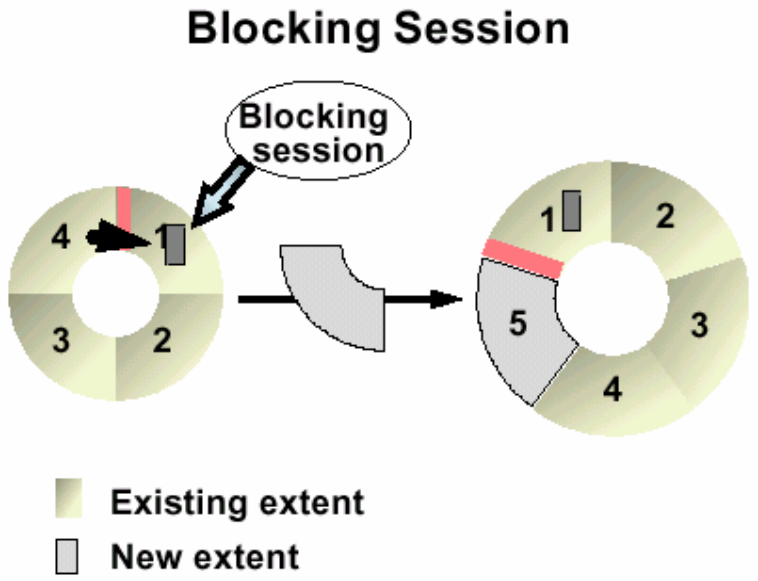
- Transaction slot có trong phần rollback header đang được sử dụng

- Giá trị ban đầu (before-image) trong rollback segment được ghi đè lên bởi một transaction khác

Giải pháp

- Tăng chỉ số `MINEXTENTS`
- Mở rộng kích thước extent
- Tăng giá trị `OPTIMAL`

11.5.3. Chặn session



Hình vẽ 57. Chặn session

Vấn đề

Khi một extent trong rollback segment được ghi đầy, Oracle server sẽ tiếp tục sử dụng extent kế tiếp theo cơ chế xoay vòng. Trong trường hợp extent kế tiếp vẫn đang trong tình trạng active, transaction sẽ không sử dụng được nó. Mặt khác, nó cũng không thể bỏ qua extent kế tiếp để chuyển tới extent sau nữa nếu nó rỗi. Khi đó, rollback segment sẽ được bổ sung thêm các extent. Việc làm này làm cho rollback segment ngày một mở rộng và quản trị viên cần phải can thiệp để hạn chế việc mở rộng này.

Giải pháp

Quản trị viên database cần thực hiện kiểm tra thông tin của các transaction đang được thực hiện thông qua việc lấy thông tin từ các view `V$ROLLSTAT`, `V$TRANSACTION`, `V$SESSION` để phát hiện các transaction đang bị cản trở, từ đó thực hiện việc điều chỉnh cho phù hợp. Công việc kiểm tra và giám sát này được thực hiện bằng tay bởi người quản trị database.

Ví dụ: Xem thông tin về các transactions đang được thực hiện

```
SVRMGR> SELECT s.sid, s.serial#, t.start_time, t.xidusn,
s.username
2> FROM v$session s, v$transaction t, v$rollstat r
3> WHERE s.saddr = t.ses_addr
4> AND t.xidusn = r.usn
```

```
5> AND ((r.curext = t.start_uext-1) OR
6> ((r.curext = r.extents-1) AND t.start_uext=0));

SID   SERIAL# START_TIME             XIDUSN USERNAME
---   ------- -------------          ------ --------
9     27      10/30/97 21:10:41      2      SYSTEM
1 row selected.
```

Chương 12. QUẢN LÝ TEMPORARY SEGMENTS

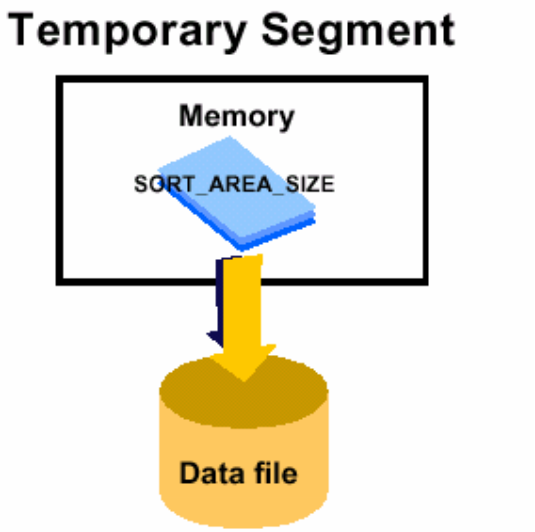
12.1.TEMPORARY SEGMENTS

Temporary segments được sử dụng khi Oracle server thực các hiện câu lệnh sắp xếp mà không thể sử dụng vùng không gian trong bộ nhớ do không đủ, ví dụ như:

- `SELECT. . . ORDER BY`
- `CREATE INDEX`
- `SELECT DISTINCT`
- `SELECT. . . GROUP BY`
- `SELECT. . . UNION`

Dung lượng bộ nhớ cần thiết cho tiến trình sắp xếp được xác định dựa trên tham số khởi tạo `SORT_AREA_SIZE`. Trong một số trường hợp, nhiều thao tác sắp xếp cùng được sử dụng và cần nhiều bộ nhớ hơn. Khi này bộ nhớ trong của máy là không thể đáp ứng được và kết quả của việc sắp xếp đó cần phải được tạm thời lưu lên đĩa. Vùng đĩa lưu trữ các dữ liệu trung gian này chính là temporary segments.

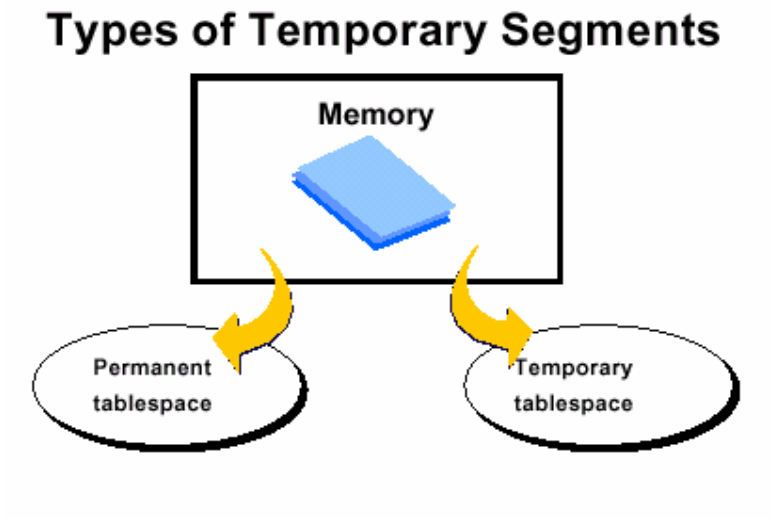
Temporary segments trong tablespace được Oracle server tạo lập với mục đích sử dụng làm vùng nhớ trung gian hỗ trợ thao tác sắp xếp.



Hình vẽ 58. Temporary segment

12.1.1. Phân loại temporary segments

Các temporary segments có thể được tạo trên một permanent tablespace hoặc trên một temporary tablespace. User có thể sử dụng một trong các kiểu tablespaces này để sắp xếp.



Hình vẽ 59. Phân loại temporary segment

Temporary Tablespace

Một temporary tablespace được sử cho các temporary segments tương ứng và không chứa bất kỳ segment nào có kiểu khác. Ta có thể tạo các temporary tablespace theo câu lệnh SQL sau:

```
CREATE TABLESPACE tablespace_name TEMPORARY
     DATAFILE filespec [autoextend_clause]
     [ , filespec [autoextend_clause]] ...
```

Một permanent tablespace có thể chuyển đổi thành dạng temporary tablespace bằng cách sử dụng câu lệnh:

```
ALTER TABLESPACE tablespace_name TEMPORARY
```

Lưu ý: với câu lệnh trên, tablespace không được phép chứa bất kỳ một đối tượng thường trú nào (như: table, store procedure, ...). Một temporary tablespace có thể chuyển đổi lại thành permanent tablespace thông qua câu lệnh SQL dưới đây:

```
ALTER TABLESPACE tablespace_name PERMANENT
```

Oracle server có thể tạo một temporary segment trong một permanent tablespace với số điều kiện sau:

- User thực hiện câu lệnh sắp xếp cần đến vùng không gian trên đĩa.
- User chạy câu lệnh mà nó đã được gán cho một permanent tablespace để thực hiện sắp xếp.

Khi một permanent tablespace được sử dụng cho việc sắp xếp, một instance có thể có một hoặc nhiều temporary segment trong tablespace.

Một temporary segment sẽ được hủy bởi tiến trình nền SMON khi kết thúc câu lệnh sắp xếp và vùng không gian đã cấp phát sẽ được giải phóng để cho các đối tượng khác của database

sử dụng. Permanent tablespaces được sử dụng cho việc sắp xếp, có ba vùng không gian trong tablespace có thể được phân vùng khác nhau. Thông thường, mỗi tablespace nên được sử dụng cho từng tiến trình sắp xếp khác nhau.

Khi một temporary tablespaces được sử dụng cho các temporary segments, Instance chỉ tạo một segment dùng để sắp xếp cho tablespace. Một vài transactions cần đến sắp xếp trên ổ đĩa có thể sử dụng cùng segment. Tuy nhiên, một extent thì không thể cùng chia sẻ đồng thời cho nhiều transactions khác nhau.

12.1.2. Sử dụng các Sort Segments

Sort segment được tạo bởi câu lệnh sắp xếp đầu tiên sử dụng tới temporary tablespace cho việc sắp xếp. Và sort segment chỉ bị hủy khi tắt (shutdown) database. Việc này làm giảm bớt số lần cấp phát và thu hồi các sort segments phục vụ cho công việc sắp xếp, làm tăng năng suất hệ thống. Oracle không hạn chế số lượng các extents cấp phát cho mỗi sort segment thuộc một temporary tablespace.

12.1.3. Sort Extent Pool

Oracle server lưu lại chi tiết sort segment trong vùng Sort Extent Pool của vùng nhớ SGA, mỗi câu lệnh cần tới các vùng trống để thực hiện sắp xếp có thể tìm các extent rỗi trong vùng nhớ này.

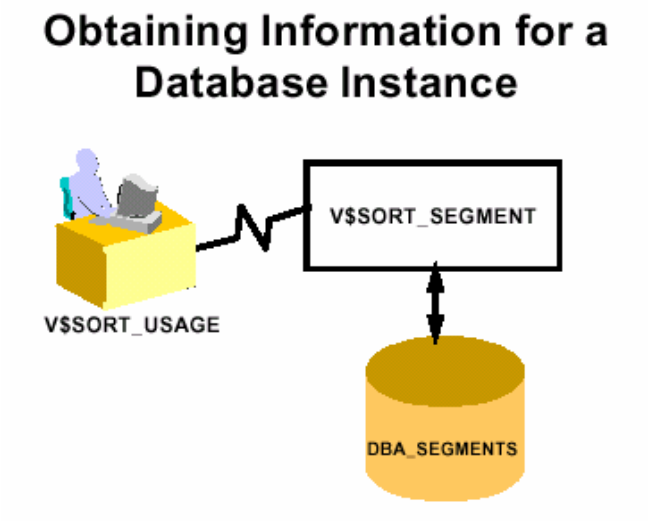
12.2.CẤP PHÁT KHÔNG GIAN CHO TEMPORARY SEGMENT

Temporary tablespaces được sử dụng để tăng hiệu quả sắp xếp dữ liệu. Kích thước của các extents trong temporary segment được xác định bởi `DEFAULT STORAGE` clause của tablespace tương ứng.

Do lượng dữ liệu ghi lên temporary segment bằng phần nguyên lần giá trị `SORT_AREA_SIZE`. Do vậy, ta nên đặt `INITIAL = NEXT = (n*SORT_AREA_SIZE)+ DB_BLOCK_SIZE`

Giá trị `PCTINCREASE=0`, để đảm bảo các extents có cùng kích thước.

12.3.THÔNG TIN VỀ CÁC TEMPORARY SEGMENT



Hình vẽ 60. Thu nhận thông tin về database instance

Ta có thể lấy được các thông tin về temporary segment trong một số bảng từ điển dữ liệu:

`DBA_SEGMENTS`: chứa thông tin về tất cả các loại segments trong database.

`V$SORT_SEGMENT`: cho biết trạng thái của các sort extent pool (vùng không gian sắp xếp). Với từ điển dữ liệu này, ta có thể biết được những thông tin sau:

Tên cột	Diễn giải
TABLESPACE_NAME	Tên temporary tablespace
EXTENT_SIZE	Kích thước của extent
TOTAL_EXTENTS	Tổng số các extents
TOTAL_BLOCKS	Tổng số các blocks
USED_EXTENTS	Số lượng extents đã sử dụng
USED_BLOCKS	Số lượng blocks đã sử dụng
FREE_EXTENTS	Số lượng extents còn trống
FREE_BLOCKS	Số lượng blocks còn trống
MAX_SORT_SIZE	Kích thước tối đa của vùng dữ liệu sắp xếp
MAX_SORT_BLOCKS	Số lượng blocks tối đa dùng để sắp xếp dữ liệu

Ví dụ:

```
SVRMGR> SELECT tablespace_name, extent_size,
     2> total_extents, max_sort_blocks
     3> FROM v$sort_segment;

TABLESPACE_NAME EXTENT_SIZ       TOTAL_EXTE  MAX_SORT_B
--------------- ----------       ----------  ----------
TEMP                   128        1           128
1 row selected.
```

MAX_SORT_SIZE và MAX_SORT_BLOCKS là số lượng các extents và các blocks sử dụng bởi phép sắp xếp lớn nhất. Thông tin này là hữu ích trong việc điều chỉnh kích thước của temporary tablespace

V$SORT_USAGE: cho biết thông tin về các sắp xếp hiện có của instance, ta kết hợp với V$SESSION để biết thêm các thông tin:

Ví dụ:

```
SVRMGR> SELECT s.username, u."USER", u.tablespace,
     2> u.contents, u.extents, u.blocks
     3> FROM v$session s,v$sort_usage u
     4> WHERE s.saddr=u.session_addr;

USERNAME        USER  TABLESPACE  CONTENTS   EXTENTS  BLOCKS
--------        ----- ----------  ---------  -------  ------
SYSTEM          SYS   TEMP        TEMPORARY  1        128
1 row selected.
```