



BÀI TẬP LỚN

HỌC PHẦN: THIẾT KẾ VÀ QUẢN TRỊ CƠ SỞ DỮ LIỆU

Đề tài:

Chapter 3 – Introduction to the Optimizer

Giảng viên hướng dẫn: TS. Trần Việt Trung

Nhóm sinh viên thực hiện: 02

Trương Lộc Bình CNTT-TT 2.2 – K56 2011 1177

Đặng Xuân Trường CNTT-TT 2.3 – K56 2011 2356

Lê Công Thái CNTT-TT 2.1 – K56 2011 2673

Nguyễn Huy Hùng CNTT-TT 2.4 – K56 2011 1520

Mã lớp: 79265 – Mã học phần: IT4859

TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI
VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÀI TẬP LỚN
HỌC PHẦN: THIẾT KẾ VÀ QUẢN TRỊ CƠ SỞ DỮ LIỆU

Đề tài:
Chapter 3 – Introduction to the Optimizer

Sinh viên thực hiện:	Trương Lộc Bình	2011 1177
	Đặng Xuân Trường	2011 2356
	Lê Công Thái	2011 2673
	Nguyễn Huy Hùng	2011 1520

Giảng viên hướng dẫn: TS. Trần Việt Trung

LỜI NÓI ĐẦU

Oracle database hay còn gọi là Oracle RDBMS (hoặc đơn giản: Oracle) là một hệ quản trị cơ sở dữ liệu, được phát triển và phân phối bởi tập đoàn Oracle. Phần mềm này vẫn đang được phát triển tiếp lên những phiên bản cao hơn và đang là một trong những phần mềm quản trị cơ sở dữ liệu hàng đầu thế giới. Oracle database hỗ trợ nhiều ngôn ngữ và nền tảng hệ điều hành khác nhau. Việc phổ biến trên nhiều nền tảng như vậy giúp các doanh nghiệp có thể dễ dàng triển khai Oracle trên các nền tảng phần cứng sẵn có, hoặc nền tảng quen thuộc với doanh nghiệp mà không bị bắt buộc phải thay đổi theo phần mềm. Điều này cũng giúp mở rộng thị trường của Oracle.

Tối ưu hóa trong các hệ quản trị cơ sở dữ liệu, trong đó có Oracle database, là một chủ đề lớn và bao hàm nhiều vấn đề, nhiều yếu tố, bao gồm cả cấu hình phần cứng và phần mềm, cài đặt hệ điều hành, bộ đệm... Trong đó, vấn đề tối ưu hóa các câu truy vấn là rất quan trọng ảnh hưởng đến hiệu quả của cơ sở dữ liệu. Trên cơ sở học tập học phần “Thiết kế và quản trị cơ sở dữ liệu”, chúng em đã lựa chọn đề tài bài tập lớn “Introduction to the Optimizer”. Hệ quản trị cơ sở dữ liệu được xem xét ở đây là Oracle database. Qua đó, chúng em đã có hiểu biết tốt hơn về những vấn đề gặp phải khi làm việc với cơ sở dữ liệu.

Với khả năng còn nhiều hạn chế, bài tập lớn của nhóm em sẽ khó tránh khỏi những thiếu sót, những chỗ chưa thỏa đáng. Chúng em rất mong nhận được sự góp ý của thầy giáo và các bạn để có thể khắc phục những nhược điểm đó và đạt được những mục tiêu đề ra của môn học. Chúng em xin chân thành cảm ơn.

Hà Nội, tháng 5 năm 2015

Nhóm sinh viên thực hiện đề tài

BẢNG PHÂN CÔNG NHIỆM VỤ

Họ và tên	Công việc thực hiện
Trương Lộc Bình	<ul style="list-style-type: none">✓ Nghiên cứu tài liệu: phần 3✓ Thuyết trình bài tập lớn
Đặng Xuân Trường	<ul style="list-style-type: none">✓ Nghiên cứu tài liệu: phần 1, 2
Lê Công Thái	<ul style="list-style-type: none">✓ Nghiên cứu tài liệu: phần 1, 2
Nguyễn Huy Hùng	<ul style="list-style-type: none">✓ Nghiên cứu tài liệu: phần 3✓ Tổng hợp, viết báo cáo

MỤC LỤC

LỜI NÓI ĐẦU	1
BẢNG PHÂN CÔNG NHIỆM VỤ	2
MỤC LỤC	3
PHẦN 1 – SƠ LƯỢC VỀ KIẾN TRÚC CƠ SỞ DỮ LIỆU ORACLE	4
1.1 CÁC THÀNH PHẦN CHÍNH	4
1.2 CẤU TRÚC BỘ NHỚ	5
1.3 CẤU TRÚC LƯU TRỮ VẬT LÝ VÀ LOGIC	6
PHẦN 2 – CÂU LỆNH SQL	8
2.1 PHÂN LOẠI	8
2.2 BIỂU DIỄN VÀ CÀI ĐẶT TRONG CƠ SỞ DỮ LIỆU ORACLE	9
2.2.1 Biểu diễn câu lệnh SQL	9
2.2.2 Cài đặt	10
2.3 QUÁ TRÌNH THỰC THI	11
2.3.1 Tổng quan	11
2.3.2 Các bước thực thi	12
PHẦN 3 – TỐI ƯU HÓA TRONG CƠ SỞ DỮ LIỆU ORACLE	14
3.1 SỰ CẦN THIẾT CỦA BỘ TỐI ƯU HÓA	14
3.2 CÁC PHA CỦA QUÁ TRÌNH TỐI ƯU	15
3.2.1 Tổng quan	15
3.2.2 Bộ chuyển đổi (Transformer)	16
3.2.3 Tối ưu hóa dựa trên chi phí	20
3.3 ĐIỀU KHIỂN HÀNH VI CỦA BỘ TỐI ƯU HÓA	22
KẾT LUẬN VÀ LỜI CẢM ƠN	26
TÀI LIỆU THAM KHẢO	26

PHẦN 1

SƠ LƯỢC VỀ KIẾN TRÚC CƠ SỞ DỮ LIỆU ORACLE

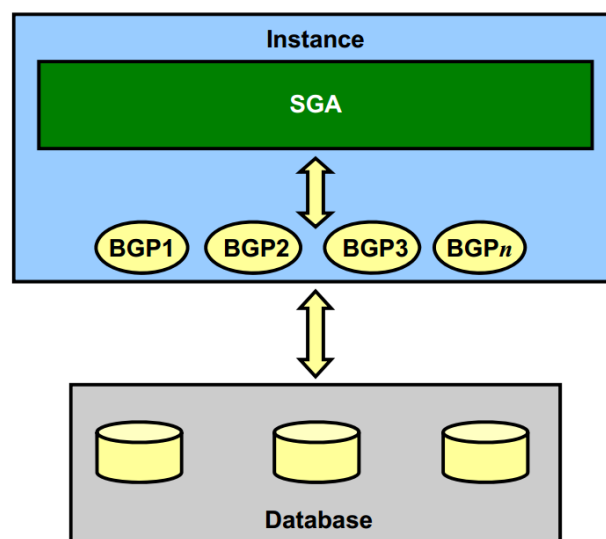
1.1 CÁC THÀNH PHẦN CHÍNH

Máy chủ cơ sở dữ liệu Oracle bao gồm một cơ sở dữ liệu Oracle và một hoặc nhiều thực thể cơ sở dữ liệu Oracle. Mỗi thực thể bao gồm nhiều cấu trúc nhớ và tiến trình nền. Khi một thực thể được khởi động, một vùng nhớ chia sẻ, gọi là System Global Area (SGA) được cấp phát và các tiến trình nền bắt đầu chạy.

Vùng SGA chứa dữ liệu và thực hiện kiểm soát thông tin cho thực thể cơ sở dữ liệu Oracle.

Các tiến trình nền hợp nhất các chức năng được thực thi bởi các chương trình máy chủ cơ sở dữ liệu Oracle chạy cho mỗi tiến trình người dùng khác nhau. Chúng có thể thực hiện bất đồng bộ hóa vào/ra và giám sát các tiến trình cơ sở dữ liệu Oracle khác để cung cấp quan hệ song song gia tăng, nhằm đem lại hiệu năng và độ tin cậy lớn hơn.

Cơ sở dữ liệu bao gồm các tệp vật lý và cấu trúc logic. Bởi vì các cấu trúc vật lý và logic được phân chia ra, bộ lưu trữ vật lý của dữ liệu có thể được quản lý mà không ảnh hưởng đến việc truy cập cấu trúc lưu trữ vật lý.



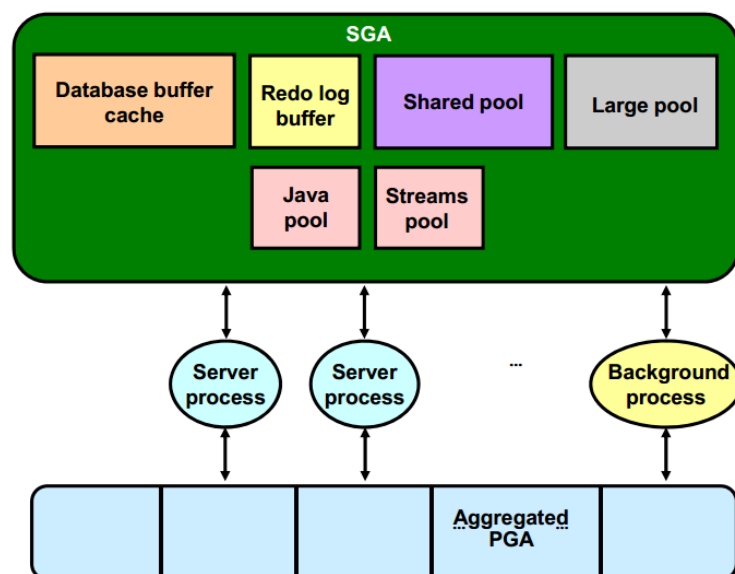
Hình 1.1 – Kiến trúc tổng thể

Cụm ứng dụng thực Oracle (Oracle Real Application Clusters – Oracle RAC) bao gồm hai hoặc nhiều thực thể cơ sở dữ liệu Oracle chạy trên các máy tính đa cụm giao tiếp với những máy khác thông qua kết nối liên tuyến và cùng truy cập một cơ sở dữ liệu.

1.2 CẤU TRÚC BỘ NHỚ

Cơ sở dữ liệu Oracle cấp phát các cấu trúc nhớ cho nhiều mục đích khác nhau. Ví dụ, vùng nhớ lưu trữ mã nguồn của chương trình đang chạy, dữ liệu chia sẻ giữa những người dùng, và một vùng dữ liệu riêng cho mỗi người dùng đã kết nối. Hai cấu trúc nhớ cơ bản liên quan đến một thực thể là:

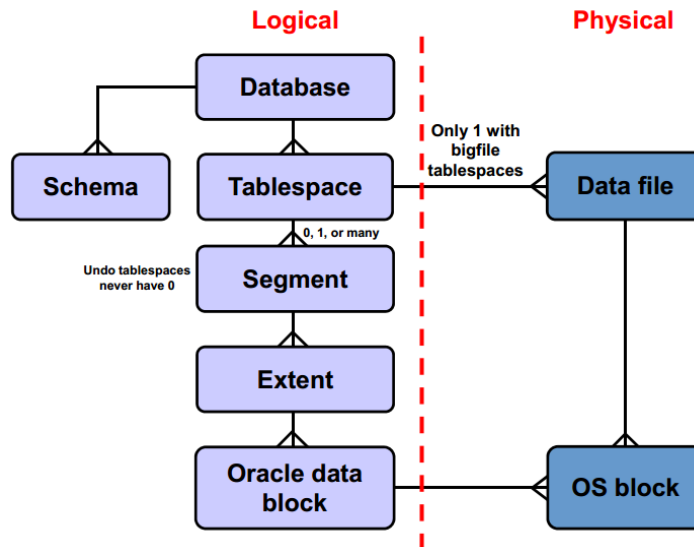
- Vùng hệ thống toàn cục (System Global Area – SGA): vùng SGA được chia sẻ bởi tất cả các tiến trình phía server và các tiến trình nền. Vùng này bao gồm các cấu trúc dữ liệu sau:
 - Database buffer cache: khối nhớ đệm nhanh những dữ liệu tìm thấy từ các tệp cơ sở dữ liệu.
 - Redo log buffer: bộ nhớ đệm nhanh khôi phục thông tin trước khi ghi vào các tệp vật lý.
 - Shared pool: các bộ nhớ đệm nhanh được xây dựng khác nhau, có thể chia sẻ các phiên làm việc.
 - Large pool: vùng tùy chọn, được sử dụng cho những thao tác chặn chắn, chẳng hạn như các thao tác sao lưu và phục hồi, các tiến trình vào ra phía server.
 - Java pool: sử dụng cho mã nguồn Java và dữ liệu chứa trong máy ảo Java (Java Virtual Machine – JVM)
 - Streams pool: sử dụng bởi Oracle Streams để lưu trữ thông tin những tiến trình được thu nạp và áp dụng.
- Vùng chương trình toàn cục (Program Global Area – PGA): đây là vùng nhớ bao gồm dữ liệu và những thông tin được kiểm soát liên quan đến server hoặc tiến trình nền. PGA được cấp phát từ vùng PGA tích hợp.



Hình 1.2 – Tổng quan cấu trúc bộ nhớ

1.3 CẤU TRÚC LƯU TRỮ VẬT LÝ VÀ LOGIC

Cơ sở dữ liệu Oracle có cấu trúc vật lý và logic. Có thể mô tả bằng sơ đồ dưới đây:



Hình 1.3 – Cấu trúc lưu trữ trong cơ sở dữ liệu

- Tablespaces (không gian bảng): một cơ sở dữ liệu được phân chia thành những đơn vị lưu trữ logic, gọi là các tablespace, có chức năng gom nhóm các cấu trúc logic có quan hệ với nhau. Ví dụ, các tablespace thường gộp tất cả các đối tượng của một ứng dụng để đơn giản hóa một vài thao tác quản trị. Có thể có những tablespace khác nhau cho những ứng dụng khác nhau.
- Database, Tablespaces và Data Files (cơ sở dữ liệu, không gian bảng và tệp dữ liệu): mối quan hệ giữa database, tablespace và data file được minh họa trong hình 1.3. Mỗi database được phân chia một cách logic thành một hoặc nhiều tablespace. Một hoặc nhiều data file được khởi tạo công khai cho mỗi tablespace để lưu trữ về mặt vật lý các dữ liệu của tất cả các cấu trúc logic trong một tablespace. Nếu đây là một tablespace tạm thời thay vì một tablespace có chứa dữ liệu, tablespace ấy phải có một file tạm thời.
- Schemas (lược đồ): một schema là một bộ sưu tập các đối tượng cơ sở dữ liệu được sở hữu bởi người dùng. Các đối tượng schema là các cấu trúc logic tham chiếu trực tiếp đến dữ liệu trong cơ sở dữ liệu. Các đối tượng schema bao gồm các cấu trúc, ví dụ như các bảng, khung nhìn, chuỗi, các thủ tục được lưu trữ, từ đồng nghĩa, cấu trúc chỉ mục, các nhóm và các liên kết cơ sở dữ liệu. Như vậy, hiểu theo cách thông thường, các đối tượng schema bao gồm mọi thứ mà ứng dụng của người dùng tạo ra trong cơ sở dữ liệu.
- Data Blocks (khối dữ liệu): đi sâu vào chi tiết, dữ liệu trong cơ sở dữ liệu Oracle được lưu trữ thành từng khối. Mỗi khối dữ liệu tương ứng với một

số lượng cụ thể các byte của không gian vật lý trên đĩa. Kích thước của khối dữ liệu thì phụ thuộc vào mỗi không gian bảng (tablespace) khi nó được khởi tạo. Một cơ sở dữ liệu sử dụng và cấp phát không gian dữ liệu miễn phí trong khối dữ liệu.

- Extents (khoảng rộng): cấp độ tiếp theo của không gian dữ liệu vật lý là khoảng rộng. Một khoảng rộng là một số lượng cụ thể những khối dữ liệu kế tiếp nhau được sử dụng để lưu trữ những loại thông tin cụ thể.
- Segments (phân đoạn): cấp độ ở trên khoảng rộng (extent) trong bộ lưu trữ cơ sở dữ liệu logic được gọi là các phân đoạn. Đây là tập hợp các khoảng rộng được cấp phát cho những cấu trúc logic nhất định. Các loại phân đoạn khác nhau là:
 - Data segments (phân đoạn dữ liệu): mỗi bảng dữ liệu không phân cụm, không đánh chỉ mục có một phân đoạn dữ liệu, trừ trường hợp các bảng ngoài và các bảng toàn cục tạm thời không có phân đoạn, các bảng đã được phân vùng (partitioned tables) có một hoặc nhiều phân đoạn. Tất cả dữ liệu trong các bảng được lưu trữ trong những khoảng rộng (extent) của các phân đoạn dữ liệu của nó. Dữ liệu của các bảng phân cụm được lưu trữ trong phân đoạn dữ liệu của cụm.
 - Index segments (phân đoạn cấu trúc chỉ mục): mỗi chỉ mục có một phân đoạn cấu trúc chỉ mục để lưu trữ tất cả dữ liệu của nó. Với những chỉ mục đã được phân vùng (partitioned index), mỗi phân vùng có một phân đoạn chỉ mục.
 - Undo segments (phân đoạn hoàn tác): một không gian bảng hoàn tác (undo tablespace) được khởi tạo cho mỗi thực thể cơ sở dữ liệu. Không gian bảng này bao gồm số lượng lớn các phân đoạn hoàn tác để lưu trữ tạm thời những thông tin hoàn tác. Thông tin này được sử dụng để tạo ra thông tin cơ sở dữ liệu đọc nhất quán, và trong suốt quá trình khôi phục cơ sở dữ liệu sẽ thu hồi lại những giao tác không liên kết cho người dùng.
 - Temporary segments (phân đoạn tạm thời): phân đoạn này được tạo bởi cơ sở dữ liệu Oracle khi một câu lệnh SQL cần một vùng làm việc tạm thời để hoàn thành việc thực thi. Khi kết thúc thực thi câu lệnh, khoảng rộng (extent) của phân đoạn tạm thời được trả lại cho thực thể để sử dụng trong tương lai. Cần chỉ rõ không gian bảng tạm thời mặc định được sử dụng cho mọi người dùng hay được sử dụng trong cơ sở dữ liệu.

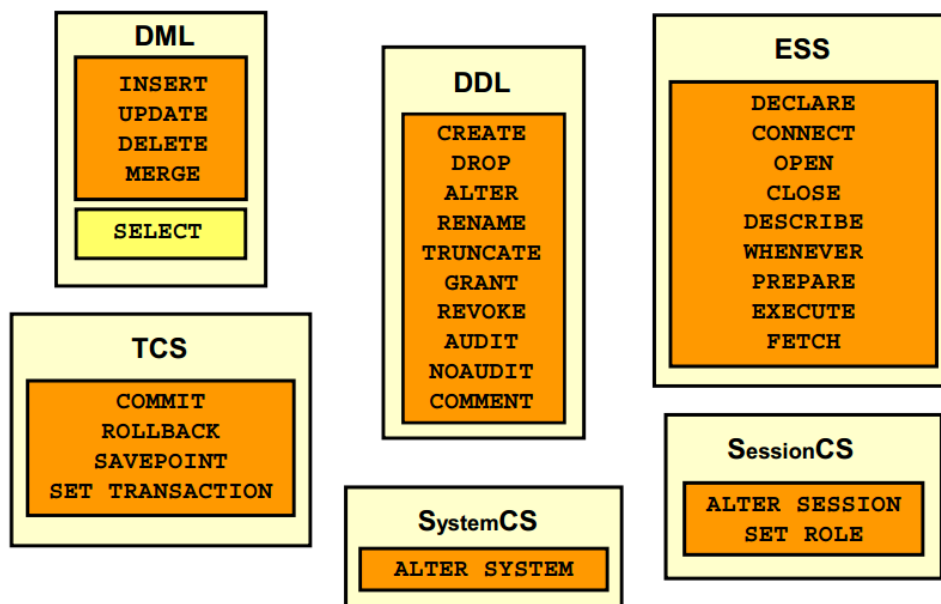
Cơ sở dữ liệu Oracle thực hiện cấp phát động không gian. Khi tồn tại những khoảng rộng của một phân đoạn nào đó đã lấp đầy, khoảng rộng bổ sung sẽ được thêm vào. Bởi vì những khoảng rộng của phân đoạn được cấp phát dựa vào nhu cầu nên chúng có thể có hoặc không liên kế nhau trên đĩa.

PHẦN 2

CÂU LỆNH SQL

2.1 PHÂN LOẠI

Công ty Oracle cố gắng tuân thủ theo chuẩn công nghiệp và tham gia vào ủy ban chuẩn hóa SQL (theo chuẩn của ANSI và ISO). Các câu lệnh SQL có thể phân loại thành 6 nhóm chính như hình dưới đây:



Hình 2.1 – Phân loại câu lệnh SQL

- Ngôn ngữ thao tác dữ liệu (Data manipulation language – DML) bao gồm các câu lệnh thực hiện việc thao tác và truy vấn dữ liệu từ các lược đồ có sẵn.
- Ngôn ngữ định nghĩa dữ liệu (Data definition language – DDL) gồm các câu lệnh thực hiện định nghĩa, tùy chỉnh và xóa đối với cấu trúc của các lược đồ dữ liệu.
- Các câu lệnh điều khiển giao tác (Transaction control statements – TCS) quản lý những thay đổi thực hiện bởi các câu lệnh thao tác dữ liệu DML, đồng thời gom nhóm chúng thành các giao tác.
- Các câu lệnh điều khiển hệ thống (System Control statements – SystemCS) thay đổi thuộc tính của các thực thể cơ sở dữ liệu Oracle.
- Các câu lệnh điều khiển phiên (Session Control statements – SessionCS) quản lý những thuộc tính trong phiên làm việc của người dùng cụ thể.

- Câu lệnh SQL nhúng (Embedded SQL statements – ESS) tiến hành việc hợp nhất các loại câu lệnh DDL, DML và TCS thành một chương trình ngôn ngữ có tính chất thủ tục, chẳng hạn như PL/SQL và những bộ tiền biên dịch Oracle. Việc hợp nhất được thực hiện bằng cách sử dụng những câu lệnh liệt kê trong danh sách ứng với ESS (xem hình 2.1).

Rõ ràng, câu lệnh SELECT được sử dụng phổ biến nhất khi làm việc với cơ sở dữ liệu bằng ngôn ngữ SQL. Tuy nhiên, cần nhấn mạnh một điều rằng mọi câu lệnh SQL đều được xem xét đến khi thực hiện việc tối ưu hóa.

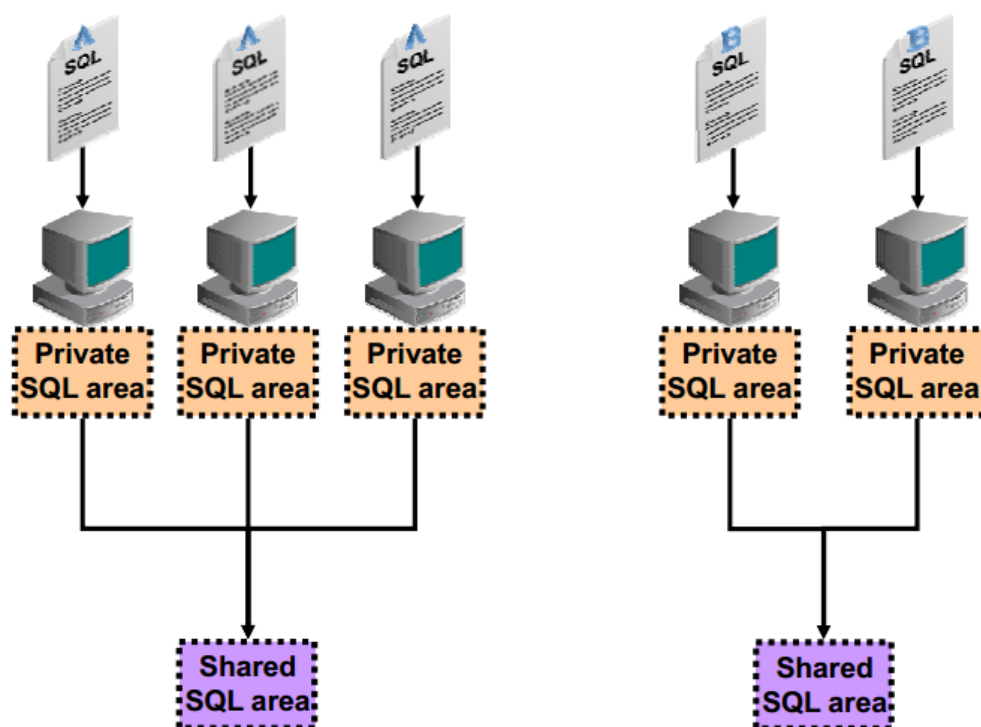
2.2 BIỂU DIỄN VÀ CÀI ĐẶT TRONG CƠ SỞ DỮ LIỆU ORACLE

2.2.1 Biểu diễn câu lệnh SQL

Cơ sở dữ liệu Oracle biểu diễn mỗi câu lệnh SQL được thực thi thông qua vùng SQL chia sẻ (shared SQL area) và một vùng SQL riêng (private SQL area). Cơ sở dữ liệu Oracle cho phép hai người dùng thực hiện đồng thời một câu lệnh SQL và tái sử dụng vùng SQL chia sẻ cho những người dùng khác. Tuy nhiên, mỗi người dùng phải có một bản sao rời vùng SQL riêng của câu lệnh ấy.

Vùng SQL chia sẻ bao gồm tất cả thông tin tối ưu hóa cần thiết cho việc thực thi câu lệnh, trong khi đó, vùng SQL riêng bao gồm những thông tin thực thi liên quan đến việc thực hiện câu lệnh cụ thể.

Cơ sở dữ liệu Oracle tiết kiệm bộ nhớ bằng cách sử dụng một vùng SQL chia sẻ cho câu lệnh SQL thực thi nhiều lần. Điều này thường xảy ra khi nhiều người dùng khởi chạy cùng một ứng dụng.



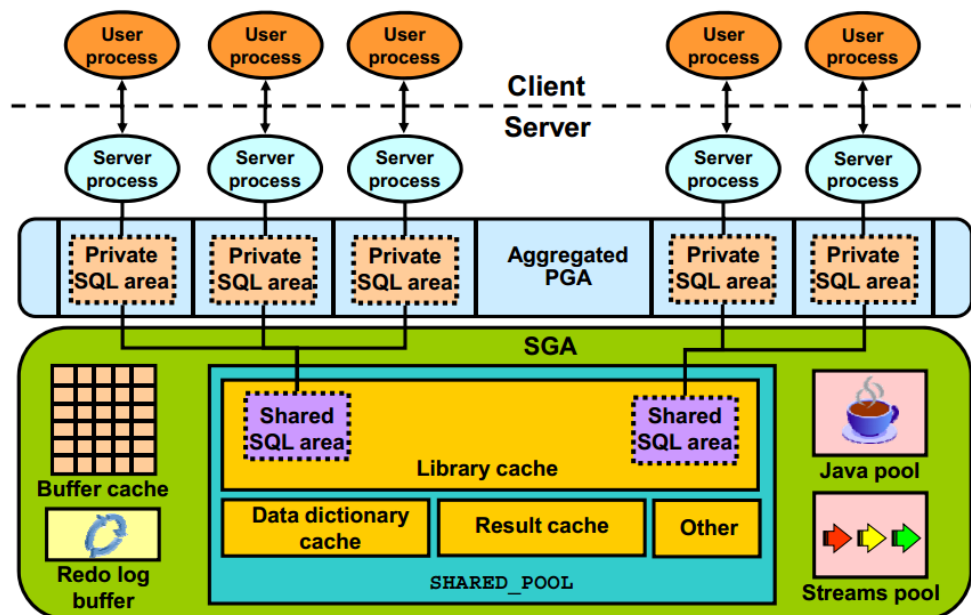
Hình 2.2 – Biểu diễn câu lệnh SQL trong cơ sở dữ liệu Oracle

Để đánh giá các câu lệnh SQL tương tự hoặc đồng nhất với nhau, cơ sở dữ liệu Oracle thực hiện xem xét các câu lệnh đưa ra bởi những người dùng hoặc ứng dụng, cũng như câu lệnh SQL đệ quy được định nghĩa bởi ngôn ngữ DDL.

2.2.2 Cài đặt

Cơ sở dữ liệu Oracle khởi tạo và sử dụng các cấu trúc nhớ cho nhiều mục đích khác nhau. Ví dụ, bộ nhớ lưu trữ các mã nguồn chương trình được khởi chạy, dữ liệu được chia sẻ với những người dùng khác nhau và vùng dữ liệu riêng cho mỗi người dùng kết nối.

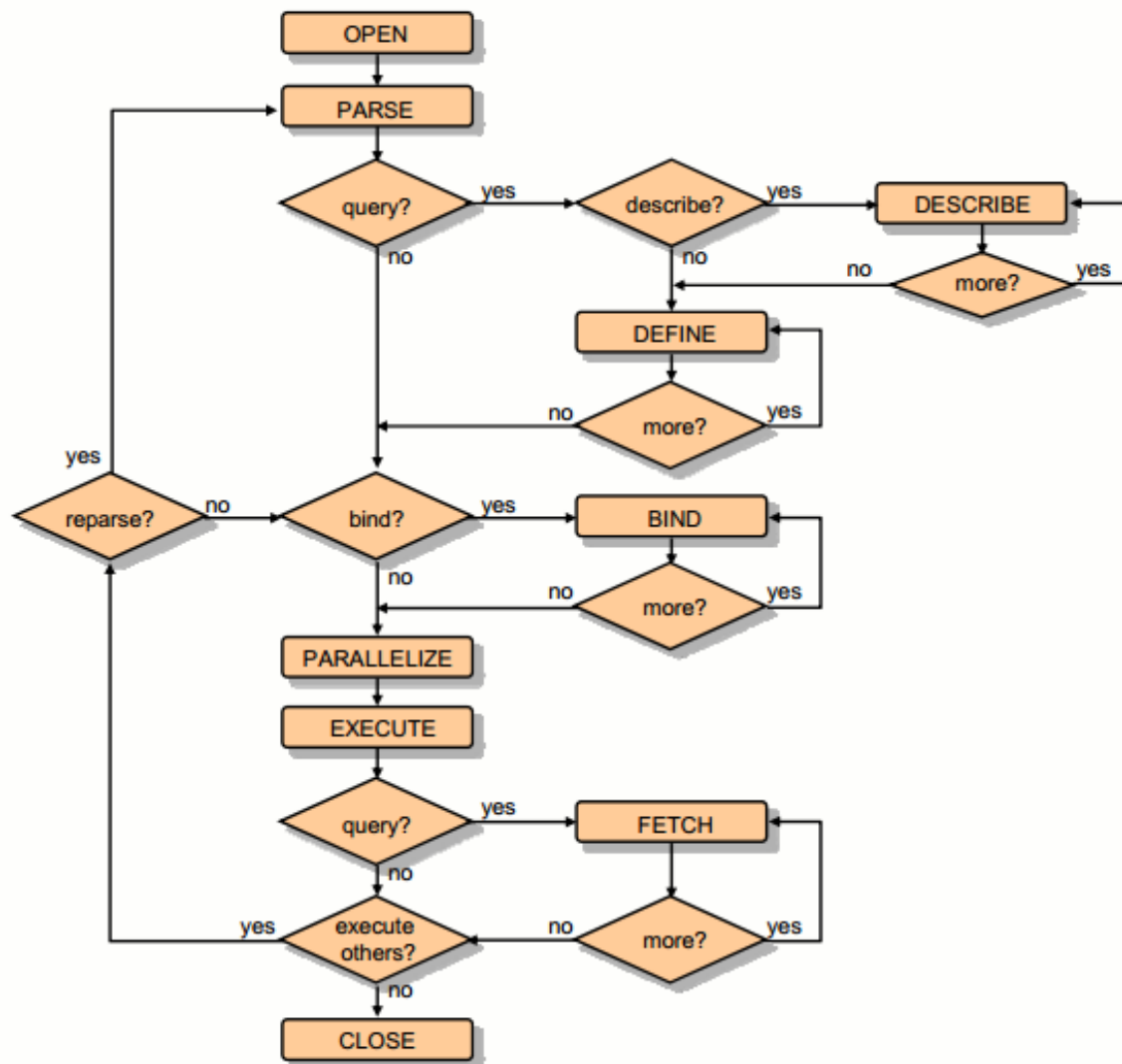
Cơ sở dữ liệu cũng thực hiện việc cấp phát bộ nhớ từ bộ trữ chia sẻ (shared pool) khi một câu lệnh SQL mới được phân tích cú pháp, và lưu trữ trong vùng SQL chia sẻ. Kích thước vùng nhớ này phụ thuộc vào độ phức tạp của câu lệnh. Nếu toàn bộ bộ trữ chia sẻ đã được cấp phát, cơ sở dữ liệu có thể từ chối việc cấp phát bằng cách sử dụng giải thuật LRU (least recently used) chỉnh sửa tới khi có đủ không gian trống cho vùng SQL chia sẻ của câu lệnh mới đó. Nếu cơ sở dữ liệu từ chối việc cấp phát vùng SQL chia sẻ, câu lệnh SQL có liên quan phải được phân tích cú pháp và chỉ định lại tới vùng SQL chia sẻ khác tại lần thực thi tiếp theo của nó.



Hình 2.3 – Cài đặt câu lệnh SQL trong cơ sở dữ liệu Oracle

2.3 QUÁ TRÌNH THỰC THI

2.3.1 Tổng quan



Sơ đồ khối ở trên trình bày quá trình thực thi câu truy vấn. Có thể chia ra làm 9 bước như sau:

- i. Khởi tạo con trỏ
- ii. Phân tích cú pháp
- iii. Mô tả kết quả truy vấn
- iv. Định nghĩa đầu ra (output)
- v. Ràng buộc biến số
- vi. Song song hóa (parallelize) câu lệnh
- vii. Thực thi
- viii. Truy xuất truy vấn theo các hàng
- ix. Đóng con trỏ

Cần lưu ý không phải việc thực thi mọi câu lệnh SQL đều phải thực hiện đầy đủ các bước nêu trên. Chẳng hạn, các câu lệnh DDL không song song chỉ cần hai bước để giải quyết: khởi tạo con trỏ và phân tích cú pháp.

2.3.2 Các bước thực thi

Sau đây chúng ta sẽ tìm hiểu cụ thể hơn về các bước nêu trên.

i. Khởi tạo con trỏ

Một con trỏ được xem như sự kết hợp giữa vùng dữ liệu con trỏ trong chương trình phía client và cấu trúc dữ liệu của máy chủ Oracle. Phần lớn các công cụ trong Oracle ẩn đi nhiều thao tác con trỏ đối với người dùng, nhưng chương trình giao diện lời gọi của Oracle (Oracle Call Interface – OCI) cần sự mềm dẻo để có thể những phần riêng rẽ của việc thực thi truy vấn. Do vậy, những bộ xử lý tiền biên dịch (precompilers) cho phép khai báo con trỏ hiện.

ii. Phân tích cú pháp

Trong bước này, câu lệnh SQL sẽ được chuyển từ tiến trình người dùng đến thực thể cơ sở dữ liệu Oracle và biểu diễn cú pháp của nó sẽ được nạp vào vùng SQL chia sẻ.

Việc dịch và xác thực bao gồm cả quá trình kiểm tra sự tồn tại của câu lệnh trong thư viện bộ nhớ đệm nhanh. Với những câu lệnh phân tán thì sẽ kiểm tra sự có mặt của những liên kết cơ sở dữ liệu.

Thông thường, pha phân tích cú pháp được biểu diễn như trạng thái mà kế hoạch thực thi được sinh ra. Bước này có thể bị hoãn lại do phần mềm phía client giảm lưu lượng mạng. Điều đó có nghĩa rằng việc phân tích cú pháp được gộp chung với bước thực thi, do vậy sẽ có ít vòng tuần hoàn hơn cho server.

iii. Mô tả kết quả truy vấn

Bước này chỉ cần thiết nếu đặc điểm của kết quả truy vấn chưa được biết trước, ví dụ như khi câu truy vấn được đưa vào bởi người dùng. Khi đó, bước mô tả sẽ xác định đặc điểm của kết quả truy vấn (bao gồm kiểu dữ liệu, độ dài, tên). Quá trình này nhằm dùng ứng dụng cơ sở dữ liệu biết được những mục cần thiết phải có trong câu lệnh select. Chẳng hạn, xét truy vấn:

```
SQL> select * from employees;
```

khi đó thông tin trong các cột của bảng employees sẽ được yêu cầu truy cập.

iv. Định nghĩa đầu ra (output)

Trong bước này, cần xác định vị trí, kích cỡ và kiểu dữ liệu của những biến được định nghĩa để nhận những giá trị đã nạp. Những biến này gọi là biến định nghĩa (define variable). Cơ sở dữ liệu Oracle sẽ thực hiện việc chuyển đổi kiểu dữ liệu nếu cần thiết.

Hai bước 3 và 4 thông thường sẽ được ẩn đi đối với người dùng, thông qua những công cụ như SQL*Plus. Tuy nhiên, với DBMS_SQL hoặc OCI, cần phải thông báo với phía client dữ liệu đầu ra là gì và được cài đặt ở chỗ nào.

v. Ràng buộc biến số

Tại bước này, cơ sở dữ liệu Oracle sẽ biết được ý nghĩa của câu lệnh SQL, nhưng vẫn không có đủ thông tin để thực thi câu lệnh ấy. Nó cần những giá trị cho tất cả các biến xuất hiện trong câu lệnh. Quá trình thu lượm những giá trị cần thiết của biến gọi là ràng buộc biến số.

vi. Song song hóa (parallelize) câu lệnh

Cơ sở dữ liệu Oracle có thể song song hóa việc thực thi những câu lệnh SQL như SELECT, INSERT, UPDATE, MERGE, DELETE và những thao tác sử dụng ngôn ngữ định nghĩa dữ liệu như việc khởi tạo chỉ mục, khởi tạo bảng với truy vấn con, các thao tác đối với những phân vùng (partitions). Việc song song hóa sẽ tạo ra tình trạng đa tiến trình phía server khi xử lý câu lệnh, do vậy quá trình sẽ được tiến hành nhanh hơn.

Song song hóa bao gồm việc phân chia các nhiệm vụ con nhằm xử lý câu lệnh cho các tiến trình phụ thuộc.

Việc phân tích cú pháp sẽ được đồng nhất nếu câu lệnh nếu câu lệnh có thể song song hóa hoặc không và đã xây dựng được kế hoạch thực thi song song phù hợp. Trong quá trình thực thi, kế hoạch đó sẽ được cài đặt nếu nguồn tài nguyên đầy đủ và khả dụng.

vii. Thực thi

Thực thi câu lệnh SQL để tạo ra kết quả mong muốn.

viii. Truy xuất truy vấn theo các hàng

Kết quả truy vấn sẽ được trả về dưới dạng bảng

ix. Đóng con trỏ

Tại bước thứ 7, cơ sở dữ liệu Oracle đã có được những thông tin và tài nguyên cần thiết, do đó câu lệnh sẽ được thực thi. Nếu nó là một câu truy vấn không có mệnh đề FOR UPDATE, không cần phải khóa bản ghi nào bởi vì không có dữ liệu nào bị thay đổi. Nếu đó là một câu lệnh UPDATE hoặc DELETE, tất cả các bản ghi mà câu lệnh tác động đến sẽ được khóa lại cho đến khi gặp những lệnh COMMIT, ROLLBACK hoặc SAVEPOINT cho giao tác. Điều này đảm bảo sự toàn vẹn của dữ liệu.

Với một vài câu lệnh, chúng ta có thể xác định số lượng các bước thực thi cụ thể sẽ được tiến hành. Đây được gọi là quá trình xử lý mảng (array processing). Cho n quá trình thực thi, khi đó giả sử vị trí định nghĩa và ràng buộc là phần bắt đầu của một mảng có n phần tử.

Trong bước truy xuất, các bản ghi được lựa chọn và sẽ được sắp thứ tự nếu truy vấn yêu cầu, và mỗi bản ghi được truy xuất thành công sẽ tiếp tục phát hiện các bản ghi phù hợp khác cho đến khi bản ghi cuối cùng được phát hiện. Đến đây, thực hiện việc đóng con trỏ, kết thúc quá trình xử lý câu lệnh SQL.

PHẦN 3

TỐI ƯU HÓA TRONG CƠ SỞ DỮ LIỆU ORACLE

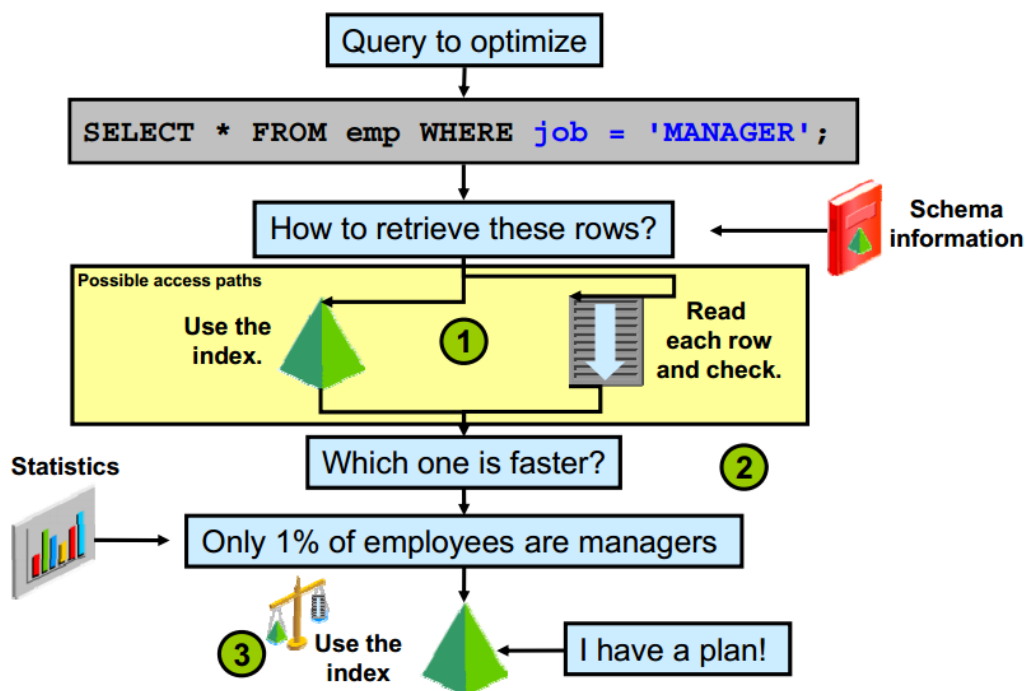
3.1 SỰ CẦN THIẾT CỦA BỘ TỐI ƯU HÓA

Bộ tối ưu hóa luôn trả về kết quả đúng nhanh nhất có thể. Nó cố gắng xác định kế hoạch thực thi hiệu quả nhất đối với mỗi câu truy vấn bằng cách xem xét các đường dẫn truy cập khả dụng và ... thông tin dựa trên thống kê cho các đối tượng dạng lược đồ (bao gồm bảng hoặc cấu trúc chỉ mục) được truy cập bởi câu lệnh SQL.

Bộ tối ưu hóa câu truy vấn thực hiện chức năng của mình theo trình tự:

- Trước hết, nó sản sinh một tập các kế hoạch thực thi có thể thực hiện được cho câu lệnh SQL, dựa trên các đường dẫn truy cập khả dụng.
- Sau đó, nó ước lượng chi phí cho mỗi kế hoạch thực thi dựa trên những thống kê trong từ điển dành cho dữ liệu phân tán và lưu trữ đặc tính của bảng, cấu trúc chỉ mục được truy cập bởi câu lệnh.
- Cuối cùng, bộ tối ưu hóa so sánh chi phí của các kế hoạch và chọn lựa kế hoạch với chi phí thấp nhất.

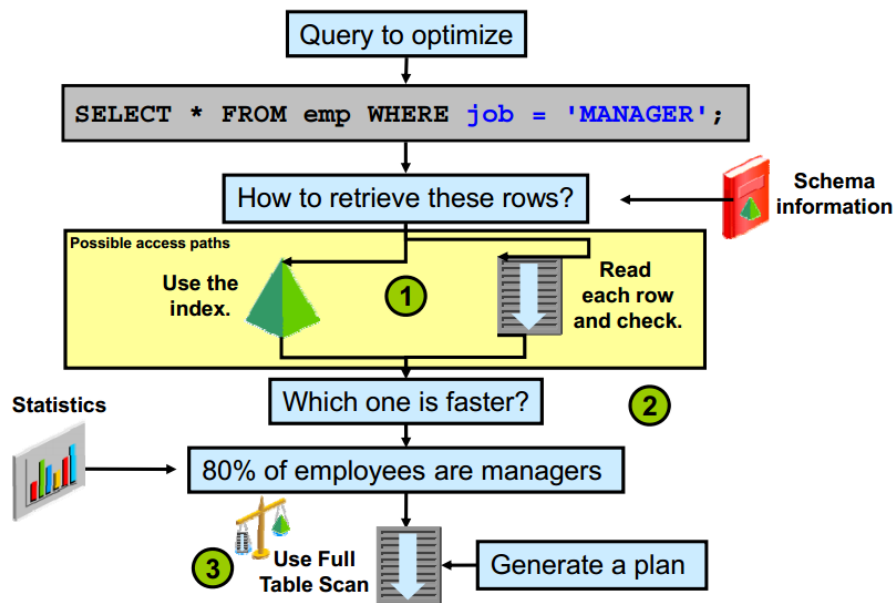
Hình dưới đây minh họa trực quan quá trình ba bước đã nêu:



Hình 3.1 – Quá trình tối ưu hóa câu truy vấn

Bởi vì việc tìm ra kế hoạch thực thi tốt nhất có thể là phức tạp đối với mỗi truy vấn cụ thể, mục tiêu của bộ tối ưu hóa là tìm ra một kế hoạch “đủ tốt” và thường được xem là kế hoạch có chi phí tốt nhất.

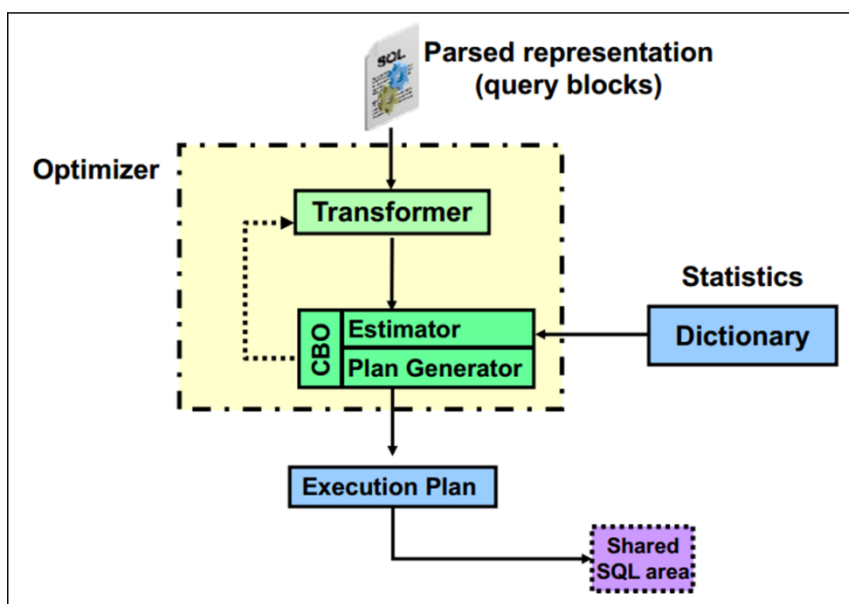
Ví dụ trong hình dưới đây cho thấy rằng nếu có sự thay đổi về mặt thống kê trong bộ dữ liệu thì bộ tối ưu sẽ thích nghi bằng cách thay đổi kế hoạch thực thi. Trong trường hợp này, giả sử 80% nhân viên là người quản lý. Khi đó, giải pháp duyệt qua toàn bộ bảng dữ liệu được xem như tốt hơn là sử dụng cấu trúc đánh chỉ mục index.



Hình 3.2 – Ví dụ minh họa

3.2 CÁC PHA CỦA QUÁ TRÌNH TỐI ƯU

3.2.1 Tổng quan



Hình 3.3 – Tổng quan quá trình tối ưu

Lược đồ trên thể hiện tóm lược các pha của quá trình tối ưu. Bộ tối ưu hóa sẽ chịu trách nhiệm khởi tạo kế hoạch thực thi cho câu lệnh SQL.

Khi những truy vấn SQL được gửi tới hệ thống, chúng sẽ được chuyển qua bộ phân tích cú pháp nhằm phân tích cú pháp và ngữ nghĩa của câu lệnh. Kết quả của pha này gọi là biểu diễn cú pháp (parsed representation) của câu lệnh, được tạo nên bởi tập hợp các khối truy vấn (query blocks). Các khối truy vấn, tự nó đã bao hàm các câu lệnh thao tác dữ liệu, trái ngược với các quan hệ. Một khối truy vấn có thể là một câu lệnh thao tác dữ liệu mức cao hoặc một truy vấn con. Biểu diễn cú pháp sau đó sẽ được gửi tới bộ tối ưu hóa, nơi thực hiện 3 chức năng chính: chuyển đổi (transformation), ước lượng (estimation) và sinh (generation) kế hoạch thực thi.

Trước khi thực hiện những tính toán liên quan đến chi phí, hệ thống có thể chuyển đổi câu lệnh sang dạng tương đương và tính chi phí của câu lệnh tương đương này. Tùy theo phiên bản của Oracle Database, những sự chuyển đổi này có thể hoặc là không được hoặc luôn luôn được thực hiện.

Đầu vào cho bộ chuyển đổi truy vấn là những truy vấn đã phân tích cú pháp, được thể hiện dưới dạng tập các khối truy vấn có liên quan với nhau. Mục tiêu chính của bộ chuyển đổi là xác định nếu việc thay đổi cấu trúc câu lệnh là thuận lợi thì nó sẽ cho phép sinh ra kế hoạch thực thi tốt hơn. Một vài kỹ thuật chuyển đổi câu truy vấn được sử dụng ở đây, ví dụ như: tính chất bắc cầu (transitivity), trộn khung nhìn (view merging), vị từ mở rộng (predicate pushing), truy vấn con không lồng nhau (subquery unnesting), viết lại câu truy vấn (query rewrite), chuyển đổi hình sao (star transformation) và phép OR mở rộng (OR expansion). Sau đây chúng ta sẽ tìm hiểu cụ thể một vài kỹ thuật trong số đó.

3.2.2 Bộ chuyển đổi (Transformer)

- Phép OR mở rộng (OR Expansion)

- Original query: B*-tree Index

```
SELECT *
  FROM emp
 WHERE job = 'CLERK' OR deptno = 10;
```

- Equivalent transformed query:

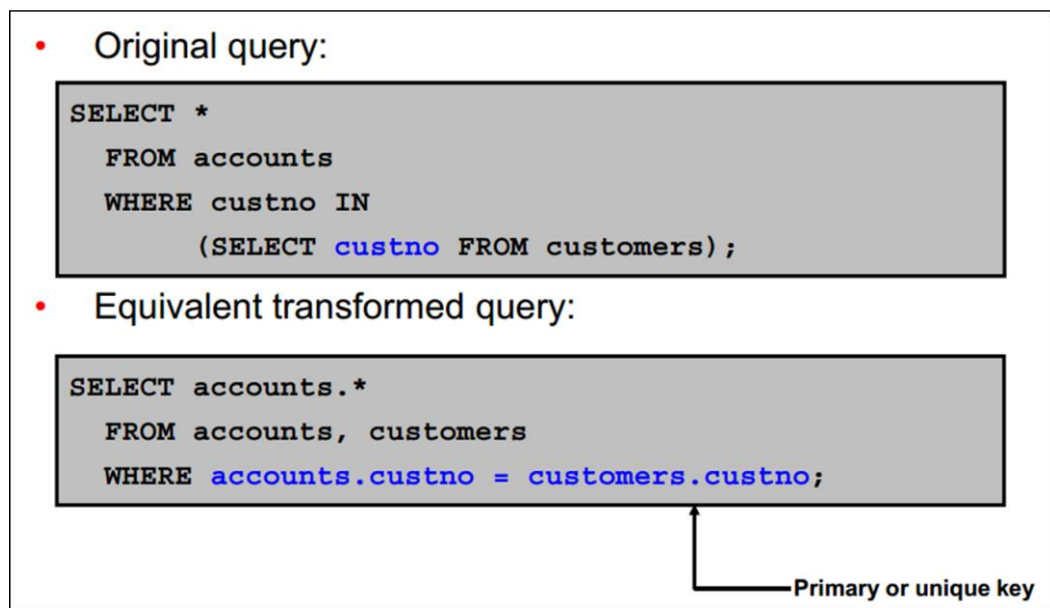
```
SELECT *
  FROM emp
 WHERE job = 'CLERK'
 UNION ALL
 SELECT *
  FROM emp
 WHERE deptno = 10 AND job <> 'CLERK';
```

Nếu một câu truy vấn bao gồm mệnh đề WHERE, với nhiều điều kiện được kết hợp bởi phép OR, bộ tối ưu hóa sẽ chuyển đổi nó thành câu truy vấn hợp tương đương có sử dụng phép UNION ALL, nếu như điều này làm tăng tính hiệu quả của việc thực thi truy vấn.

Chẳng hạn, nếu mỗi điều kiện truy vấn đều đã được đánh chỉ mục, bộ tối ưu hóa có thể thực hiện việc chuyển đổi. Nó sẽ chọn một kế hoạch thực thi cho câu lệnh tương đương bằng cách truy cập bảng và sử dụng các cách đánh chỉ mục khác nhau rồi xuất ra các kết quả. Việc chuyển đổi sẽ được thực hiện nếu chi phí ước lượng được tốt hơn chi phí thực hiện câu lệnh ở dạng ban đầu.

Trong ví dụ trên, giả sử rằng cả hai trường JOB và DEPTNO đều được đánh chỉ mục. Khi đó, bộ tối ưu hóa có thể sẽ chuyển câu truy vấn ban đầu (original) thành dạng tương đương như chúng ta thấy. Một bộ tối ưu hóa dựa trên chi phí (the cost-based optimizer – CBO) sẽ quyết định việc có hay không thực hiện sự chuyển đổi, dựa trên so sánh chi phí khi thực thi câu lệnh ban đầu bằng cách duyệt toàn bộ bảng với việc thực thi câu lệnh đã chuyển đổi.

- Truy vấn con không lồng nhau (Subquery Unnesting)



Để tránh việc có các truy vấn lồng nhau, bộ tối ưu hóa có thể thực hiện chuyển đổi câu truy vấn ban đầu thành dạng tương đương có sử dụng phép nối JOIN, sau đó tối ưu câu lệnh có chứa phép JOIN này.


Bộ tối ưu hóa sẽ thực hiện chuyển đổi nếu như kết quả của phép nối đảm bảo trả về chính xác những bản ghi như trong câu lệnh ban đầu. Điều này cho phép bộ tối ưu tận dụng được những ưu điểm của kỹ thuật kết nối.

Trong ví dụ đưa ra, nếu trường CUSTNO của quan hệ CUSTOMERS là khóa chính hoặc có ràng buộc duy nhất, bộ tối ưu hóa có thể chuyển câu truy vấn phức tạp ban đầu thành câu lệnh có sử dụng phép nối đảm bảo trả về đúng kết quả thực thi.

Nếu bộ tối ưu không thể thực hiện chuyển đổi một câu lệnh phức thành câu lệnh chứa phép nối JOIN, nó sẽ chọn kế hoạch thực thi cho câu lệnh cha (parent statement) và câu lệnh con (subquery) như là các câu lệnh này được chia tách nhau. Bộ tối ưu sẽ thực thi câu lệnh con, sau đó sử dụng những bản ghi trả về để thực thi câu lệnh cha.

Có một chú ý ở đây, đó là những truy vấn phức tạp lồng nhau mà truy vấn con bên trong bao gồm những hàm tích hợp như AVG thì không thể chuyển thành dạng kết nối JOIN.

- Trộn khung nhìn (View merging)

- **Original query:**  Index

```
CREATE VIEW emp_10 AS
  SELECT empno, ename, job, sal, comm, deptno
  FROM emp
  WHERE deptno = 10;
```

```
SELECT empno FROM emp_10 WHERE empno > 7800;
```

- **Equivalent transformed query:**

```
SELECT empno
  FROM emp
  WHERE deptno = 10 AND empno > 7800;
```

Để trộn các truy vấn của khung nhìn thành một khối truy vấn tham chiếu trong câu lệnh đang xử lý, bộ tối ưu hóa sẽ thay thế khung nhìn với tên của các quan hệ gốc của nó trong khối truy vấn và thêm vào mệnh đề WHERE của truy vấn của khung nhìn ấy để truy cập mệnh đề WHERE của khối truy vấn


Thao tác tối ưu hóa này được áp dụng cho những khung nhìn chiếu-chọn-kết nối (select-project-join views), tức là những khung nhìn chỉ bao gồm phép chiếu, chọn, kết nối mà không bao gồm những thao tác tập hợp, tích hợp hàm như DISTINCT, GROUP BY, CONNECT BY, v.v...

Trong ví dụ trên, khung nhìn là tất cả những nhân viên làm việc trong phòng số 10. Câu truy vấn chọn ra những số hiệu nhân viên có giá trị lớn hơn 7800 và đồng thời nhân viên ấy làm việc trong phòng số 10.

Bộ tối ưu hóa có thể chuyển câu truy vấn sang dạng tương đương như ta thấy trong hình, mà nó truy cập đến những quan hệ gốc của khung nhìn.

Nếu các trường DEPTNO hoặc EMPNO được đánh chỉ mục index, mệnh đề kết quả WHERE sẽ làm cho chúng trở thành có giá trị.

○ Vị từ mở rộng (Predicate Pushing)

- Original query:  Index


```
CREATE VIEW two_emp_tables AS
SELECT empno, ename, job, sal, comm, deptno FROM emp1
UNION
SELECT empno, ename, job, sal, comm, deptno FROM emp2;

SELECT ename FROM two_emp_tables WHERE deptno = 20;
```
- Equivalent transformed query:



```
SELECT ename
FROM ( SELECT empno, ename, job, sal, comm, deptno
      FROM emp1 WHERE deptno = 20
      UNION
      SELECT empno, ename, job, sal, comm, deptno
      FROM emp2 WHERE deptno = 20 );
```

Bộ tối ưu hóa có thể chuyển một khối truy vấn truy cập tới một khung nhìn không trộn được (nonmergeable) bằng cách mở rộng vị từ của khối truy vấn bên trong câu truy vấn của khung nhìn ấy.

Trong ví dụ đưa ra, khung nhìn `two_emp_tables` là sự kết hợp của hai quan hệ về nhân viên. Khung nhìn này được định nghĩa thông qua một truy vấn phức sử dụng phép toán tập hợp UNION. Câu truy vấn đưa ra đã truy cập vào khung nhìn. Nó thực hiện việc chọn những ID và tên của nhân viên làm việc trong phòng số 20.

Bởi vì khung nhìn được định nghĩa dưới dạng một câu truy vấn phức, bộ tối ưu hóa không thể trộn truy vấn của khung nhìn thành khối truy vấn truy nhập. Thay vào đó, nó có thể chuyển những câu lệnh truy nhập bằng cách mở rộng vị từ của nó, điều kiện của mệnh đề WHERE là `deptno = 20`, vào câu truy vấn phức của khung nhìn. Câu truy vấn tương đương được thể hiện như trong hình.

○ Tính chất bắc cầu (Transitivity)

- Original query:  Index


```
SELECT *
FROM emp, dept
WHERE emp.deptno = 20 AND emp.deptno = dept.deptno;
```
- Equivalent transformed query:


```
SELECT *
FROM emp, dept
WHERE emp.deptno = 20 AND emp.deptno = dept.deptno
AND dept.deptno = 20;
```


Nếu hai điều kiện trong mệnh đề WHERE đều liên quan đến cùng một trường, đôi lúc bộ tối ưu hóa có thể suy ra điều kiện thứ 3, bằng cách sử dụng nguyên lý bắc cầu. Nó có thể sử dụng điều kiện suy diễn này để tối ưu hóa câu lệnh.

Ví dụ được thể hiện trong hình trên. Mệnh đề WHERE của câu truy vấn ban đầu bao gồm hai điều kiện đều sử dụng trường EMP.DEPTNO. Bằng cách bắc cầu, bộ tối ưu hóa suy ra điều kiện thứ 3: dept.deptno = 20.

Nếu tồn tại cách đánh chỉ mục cho trường DEPT.DEPTNO, điều kiện này làm cho các đường dẫn truy cập sẵn sàng sử dụng cấu trúc chỉ mục đó.

Có một lưu ý là bộ tối ưu chỉ có thể suy ra các điều kiện về mối liên hệ giữa các trường và các biểu thức hằng, ít khi nó suy ra mối liên hệ giữa các trường với nhau.

3.2.3 Tối ưu hóa dựa trên chi phí

Hai bộ phận của mã là bộ ước lượng (estimator) và bộ sinh kế hoạch thực thi (plan generator).

Bộ ước lượng có nhiệm vụ xác định chi phí của những gợi ý tối ưu được thực hiện bởi bộ sinh kế hoạch thực thi. Chi phí ở đây được hiểu là ước lượng tốt nhất của bộ tối ưu về số lượng thao tác vào/ra chuẩn hóa cần thiết để thực hiện việc tối ưu những câu lệnh cụ thể.

Chức năng chính của bộ sinh kế hoạch thực thi là thử nghiệm những kế hoạch thực thi có thể đối với một truy vấn cho trước và đưa ra phương án có chi phí thấp nhất. Có nhiều kế hoạch thực thi khả dĩ khác nhau bởi vì có nhiều cách kết hợp đa dạng của những đường dẫn truy cập khác nhau, phương pháp kết nối và thứ tự kết nối có thể sử dụng để truy cập và xử lý dữ liệu theo những cách khác nhau nhưng cùng đưa ra một kết quả. Số lượng những kế hoạch thực thi khả dĩ cho mỗi khối truy vấn thì tỉ lệ với số mục kết nối (join item) trong mệnh đề FROM. Con số này sẽ tăng theo hàm mũ của số mục kết nối.

Bộ tối ưu hóa sử dụng những mảnh thông tin đa dạng khác nhau để xác định con đường tốt nhất: mệnh đề WHERE, thống kê, tham số khởi tạo, những gợi ý được cung cấp và thông tin lược đồ.

Dưới đây ta sẽ trình bày cụ thể hơn về bộ ước lượng và bộ sinh kế hoạch thực thi.

- Bộ ước lượng (Estimator)
 - Tính chọn lọc (Selectivity)

$\text{Selectivity} = \frac{\text{Number of rows satisfying a condition}}{\text{Total number of rows}}$

Tính chọn lọc là tỉ lệ ước lượng của tập các bản ghi được phát hiện bởi một vị từ cụ thể hoặc tổ hợp các vị từ. Đây là một đại lượng nhận giá trị trong khoảng từ 0.0 đến 1.0. Tính chọn lọc cao đồng nghĩa với tỉ lệ nhỏ các bản ghi và ngược lại. Trên đây là công thức tính độ chọn lọc.

- Lực lượng (Cardinality)

$$\text{Cardinality} = \text{Selectivity} * \text{Total number of rows}$$

Lực lượng của một thao tác cụ thể trong kế hoạch thực thi truy vấn biểu diễn số lượng bản ghi ước lượng được tìm thấy bởi thao tác đó. Trong phần lớn thời gian thực hiện, nguồn cung cấp các bản ghi thường là các quan hệ, khung nhìn hoặc kết quả của phép kết nối hoặc thao tác GROUP BY.

- Chi phí (Cost)

Chi phí của một câu lệnh là ước lượng tốt nhất về số thao tác vào/ra chuẩn hóa cần thiết để thực thi câu lệnh ấy. Thông thường, khái niệm này thường được hiểu như số lượng các khối đơn (single block) được đọc ngẫu nhiên.

$$\text{Cost} = \frac{\text{Single block I/O cost} + \text{Multiblock I/O cost} + \text{CPU cost}}{\text{sreadtim}}$$

Where:

- Single block I/O cost: $\#SRds * sreadtim$
- Multiblock I/O cost: $\#MRds * mreadtim$
- CPU cost: $\#CPUCycles / cpuspeed$

$\#SRds$: Number of single block reads

$sreadtim$: Single block read time

$\#MRds$: Number of multiblock reads

$mreadtim$: Multiblock read time

$\#CPUCycles$: Number of CPU Cycles

$cpuspeed$: Millions instructions per second

- Bộ sinh kế hoạch thực thi (Plan Generator)

Bộ sinh kế hoạch thực thi sẽ khám phá ra những kế hoạch khác nhau cho một khối truy vấn bằng cách thử những đường dẫn truy cập khác nhau, phương pháp kết nối và thứ tự kết nối. Cuối cùng, nó sẽ đưa ra kế hoạch thực thi tốt nhất cho truy vấn đầu vào. Hình ảnh đi kèm dưới đây mô tả một trích dẫn của tập tối ưu sinh ra cho câu lệnh select. Như chúng ta thấy, bộ sinh kế hoạch thực thi đã đưa ra 6 phương án khả dĩ, hoặc 6 phương án khác nhau để kiểm tra: 2 thứ tự kết nối khác nhau, ứng với mỗi cách là 3 phương pháp kết nối. Ví dụ này có giả sử là không có trường nào được đánh chỉ mục.

Để tìm ra những bản ghi phù hợp, có thể bắt đầu bằng việc kết nối các quan hệ DEPARTMENTS và EMPLOYEES. Với thứ tự kết nối cụ thể, có thể một trong 3 cơ chế kết nối khác nhau mà bộ tối ưu hóa biết đến: vòng lặp lồng nhau (nested loop), sắp xếp trộn (sort merge) hoặc kết nối băm (hash join). Với

mỗi lựa chọn này, chúng ta sẽ có được chi phí của kế hoạch thực thi tương ứng. Kế hoạch tốt nhất được in ra ở cuối.

```
select e.last_name, d.department_name
from   employees e, departments d
where  e.department_id = d.department_id;
```

```
Join order[1]: DEPARTMENTS[D]#0 EMPLOYEES[E]#1
NL Join: Cost: 41.13 Resp: 41.13 Degree: 1
SM cost: 8.01
HA cost: 6.51
Best:: JoinMethod: Hash
Cost: 6.51 Degree: 1 Resp: 6.51 Card: 106.00
Join order[2]: EMPLOYEES[E]#1 DEPARTMENTS[D]#0
NL Join: Cost: 121.24 Resp: 121.24 Degree: 1
SM cost: 8.01
HA cost: 6.51
Join order aborted
Final cost for query block SEL$1 (#0)
All Rows Plan:
Best join order: 1
```

Id	Operation	Name	Rows	Bytes	Cost
0	SELECT STATEMENT				7
1	HASH JOIN		106	6042	7
2	TABLE ACCESS FULL	DEPARTMENTS	27	810	3
3	TABLE ACCESS FULL	EMPLOYEES	107	2889	3

3.3 ĐIỀU KHIỂN HÀNH VI CỦA BỘ TỐI ƯU HÓA

Hành vi của bộ tối ưu hóa được điều khiển thông qua các tham số. Sau đây chúng ta sẽ xem xét cụ thể từng loại.

- **CURSOR_SHARING** xác định những loại câu lệnh SQL nào có thể cùng chia sẻ con trỏ, chia thành các trường hợp:
 - **FORCE**: ép các câu lệnh khác nhau một vài chuỗi ký tự, nhưng lại đồng nhất theo khía cạnh khác, cùng chia sẻ một con trỏ, trừ khi các chuỗi ký tự ảnh hưởng đến ý nghĩa của câu lệnh.
 - **SIMILAR**: ép các câu lệnh có thể khác nhau một vài chuỗi ký tự, nhưng lại đồng nhất theo khía cạnh khác, chia sẻ cùng một con trỏ, trừ khi chuỗi ký tự khác nhau ảnh hưởng đến ý nghĩa của câu lệnh hoặc mức độ tối ưu của kế hoạch thực thi. Bắt con trỏ chia sẻ giữa những câu lệnh tương tự (nhưng không đồng nhất) có thể dẫn đến những kết quả không mong muốn trong một vài ứng dụng hệ trợ giúp quyết định (decision support system – DSS) hoặc những ứng dụng sử dụng để lưu trữ những bản phác thảo.

- EXACT: chỉ cho phép những câu lệnh với nguyên văn đồng nhất chia cùng một con trỏ. Đây là thuộc tính mặc định.
- DB_FILE_MULTIBLOCK_READ_COUNT: một trong những tham số có thể sử dụng để tối thiểu hóa vào/ra trong quá trình duyệt bảng dữ liệu hoặc đánh chỉ mục đầy đủ. Nó quy định rõ số lượng tối đa các block được đọc trong một thao tác vào/ra trong suốt quá trình duyệt dữ liệu tuần tự. Số lượng thao tác vào/ra cần thiết để thực hiện duyệt hoặc đánh chỉ mục toàn bộ bảng dữ liệu phụ thuộc vào nhiều yếu tố, chẳng hạn như kích thước các đoạn, tổng số khối được đọc, và có hay không các thao tác thực thi song song.
- PGA_AGGREGATE_TARGET: chỉ ra mục tiêu tích hợp bộ nhớ PGA khả dụng với tất cả tiến trình phía server đính kèm với thực thể. Cài đặt tham số này với giá trị khác 0 sẽ dẫn đến việc cài đặt tự động tham số WORKAREA_SIZE_POLICY thành AUTO. Điều này có nghĩa là vùng SQL đang hoạt động được sử dụng bởi những thao tác SQL cần nhiều bộ nhớ (ví dụ như sắp xếp, gom nhóm, kết nối băm, trộn ảnh bitmap, tạo ảnh bitmap) sẽ được xác định kích thước tự động. Mặc định, giá trị của tham số sẽ khác 0, bởi vì trừ khi người dùng chỉ định, hệ thống sẽ chọn giá trị lớn hơn, hoặc là 20% của SGA hoặc 10MB.
- STAR_TRANSFORMATION_ENABLED: tham số chỉ ra rằng có hay không sự chuyển đổi câu truy vấn dựa trên chi phí được áp dụng vào các câu truy vấn dấu sao.
- RESULT_CACHE_MODE: bộ tối ưu hóa truy vấn quản lý cơ chế bộ nhớ đệm nhanh lưu trữ kết quả dựa trên việc cài đặt tham số này khi tiến hành khởi tạo tập tham số. Có thể dùng nó để xác định bộ tối ưu hóa có tự động gửi kết quả câu truy vấn tới bộ nhớ đệm nhanh hay không. Tham số này nhận một trong hai giá trị:
 - Mặc định là MANUAL, khi đó phải xác định kết quả cụ thể lưu trữ trong bộ nhớ đệm nhanh bằng cách sử dụng gợi ý RESULT_CACHE.
 - Khi tham số nhận giá trị FORCE, tất cả các dữ liệu được lưu trữ trong bộ nhớ đệm nhanh. Trong tình huống này, nếu câu lệnh bao gồm gợi ý [NO_] RESULT_CACHE thì sẽ dẫn đến việc có thể bỏ qua cài đặt tham số.
- RESULT_CACHE_MAX_SIZE: kích thước bộ nhớ được cấp phát dành cho bộ nhớ đệm nhanh lưu trữ kết quả phụ thuộc vào kích thước bộ nhớ của SGA cũng như hệ thống quản lý nhớ. Có thể thay đổi bộ nhớ cấp phát cho bộ nhớ đệm nhanh lưu trữ kết quả bằng cách cài đặt tham số RESULT_CACHE_MAX_SIZE. Bộ nhớ đệm nhanh sẽ bị vô hiệu hóa nếu giá trị tham số bằng 0. Giá trị này được làm tròn tới bội số lớn nhất của 32KB nhưng không lớn hơn giá trị đã được chỉ định trước.

- **RESULT_CACHE_MAX_RESULT**: tham số này được sử dụng để xác định số lượng bộ nhớ đệm nhanh lớn nhất mà mỗi kết quả đơn được phép sử dụng. Giá trị mặc định là 5%, nhưng người dùng có thể điều chỉnh lại giá trị khác nằm giữa 1 và 100%.
- **RESULT_CACHE_REMOTE_EXPIRATION**: tham số dùng để xác định thời gian (tính bằng phút) cho những kết quả phụ thuộc vào đối tượng cơ sở dữ liệu từ xa còn hợp lệ. Giá trị mặc định là 0, muốn chỉ ra rằng những kết quả sử dụng những đối tượng dữ liệu từ xa không được lưu trữ trong bộ nhớ đệm nhanh. Việc cài đặt tham số với những giá trị khác 0 có thể đưa ra những câu trả lời không còn chính xác.
- **OPTIMIZER_INDEX_CACHING**: tham số điều khiển giá trị của chỉ mục thăm dò (index probe) trong mỗi liên quan với vòng lặp lồng nhau (nested loop) hoặc một biến lặp trong danh sách (inlist iterator). Phạm vi giá trị của tham số từ 0 đến 100 chỉ ra rằng tỉ lệ phần trăm các khối chỉ mục trong bộ nhớ đệm, mà đã được cải biên trong giả định của bộ tối ưu hóa. Giá trị bằng 100 cho biết 100% các khối chỉ mục được tìm thấy trong bộ nhớ đệm và bộ tối ưu hóa điều chỉnh chi phí của chỉ mục thăm dò hoặc vòng lặp. Mặc định của tham số là giá trị 0, là kết quả trong cách xử lý mặc định của bộ tối ưu hóa. Cần sử dụng cảnh báo khi dùng đến tham số này bởi vì kế hoạch thực thi có thể thay đổi cho phù hợp với bộ nhớ đệm nhanh chỉ mục.
- **OPTIMIER_INDEX_COST_ADJ**: sử dụng trong việc điều chỉnh hành vi của bộ tối ưu với sự lựa chọn đường dẫn truy cập để có nhiều hoặc ít chỉ mục hơn, có nghĩa là làm cho bộ tối ưu có ít hoặc nhiều xu hướng hơn trong việc lựa chọn đường dẫn truy cập cho chỉ mục khi duyệt toàn bộ bảng dữ liệu. Phạm vi của biến là từ 1 đến 10000. Giá trị mặc định là 100%, bộ tối ưu hóa ước lượng đường dẫn truy cập chỉ mục với mức chi phí thông thường. Những giá trị khác làm cho bộ tối ưu hóa ước lượng giá trị đường dẫn tại mức phần trăm tương ứng theo chi phí thông thường. Ví dụ, cài đặt giá trị tham số là 50 làm cho đường dẫn truy cập chỉ mục có vẻ như có chi phí đắt gấp đôi so với thông thường.
- **OPTIMIZER_FEATURES_ENABLED**: được xem như một tham số bảo vệ (umbrella parameter) cho việc kích hoạt chuỗi đặc trưng tối ưu hóa dựa vào phiên bản phát hành của phần mềm.
- **OPTIMIZER_MODE**: thiết lập hành vi mặc định trong việc lựa chọn cách tiếp cận tối ưu hóa cho thực thể cơ sở dữ liệu hoặc tác vụ người dùng. Những giá trị có thể nhận của tham số:
 - **ALL_ROWS**: bộ tối ưu hóa sử dụng cách tiếp cận dựa trên chi phí cho tất cả câu lệnh SQL trong phiên bất chấp sự có mặt của bộ thống kê, và nó thực hiện tối ưu với mục tiêu đạt được thông lượng tốt nhất (tối

thiếu hóa tài nguyên sử dụng cho việc thực thi câu lệnh). Đây là giá trị mặc định của tham số.

- **FIRST_ROW_n**: bộ tối ưu hóa sử dụng cách tiếp cận dựa trên chi phí, bất kể sự có mặt của bộ thống kê, và tối ưu hóa với mục tiêu thời gian phản hồi tốt nhất khi trả về n bản ghi đầu tiên, với n nhận một trong các giá trị 1, 10, 100 hoặc 1000.
- **FIRST_ROWS**: bộ tối ưu hóa sử dụng sự kết hợp giữa chi phí và heuristics để tìm ra kế hoạch tốt nhất để nhanh chóng trả về một vài bản ghi đầu tiên. Việc sử dụng heuristics thỉnh thoảng dẫn đến bộ tối ưu hóa truy vấn sinh ra một kế hoạch với chi phí lớn hơn đáng kể so với không sử dụng heuristics. Giá trị này khả dụng cho sự tương hợp lùi và tính ổn định của kế hoạch. Có thể thay thế bằng giá trị **FIRST_ROW_n**.
- **OPTIMIZER_CAPTURE_SQL_PLAN_BASELINES**: tham số cho phép hoặc vô hiệu hóa việc xác nhận tự động các câu lệnh SQL lặp lại, cũng như việc tạo ra các đường cơ sở kế hoạch SQL (SQL plan baselines) cho những câu lệnh như vậy.
- **OPTIMIZER_USE_SQL_PLAN_BASELINES**: tham số cho phép hoặc vô hiệu hóa việc sử dụng các đường cơ sở kế hoạch SQL được lưu trữ trong cơ sở quản lý SQL. Nếu được phép, bộ tối ưu hóa sẽ tìm kiếm một đường cơ sở kế hoạch SQL cho câu lệnh SQL đã được biên dịch. Trong trường hợp tìm thấy, bộ tối ưu hóa sẽ tính chi phí của mỗi kế hoạch cơ sở và chọn cái có chi phí thấp nhất.
- **OPTIMIZER_DYNAMIC_SAMPLING**: tham số điều khiển mức độ lấy mẫu tự động của bộ tối ưu. Tham số này được cài đặt như sau:
 - Phiên bản 10.0.0 hoặc mới hơn, giá trị mặc định là 2.
 - Phiên bản 9.2.0, giá trị mặc định là 1.
 - Phiên bản 9.0.1 hoặc cũ hơn, giá trị mặc định là 0.
- **OPTIMIZER_USE_INVISIBLE_INDEXES**: tham số cho phép hoặc vô hiệu hóa việc sử dụng cấu trúc chỉ mục ẩn.
- **OPTIMIZER_USE_PENDING_STATISTICS**: tham số chỉ ra có hay không việc bộ tối ưu hóa sử dụng những thống kê chưa được duyệt trong quá trình biên dịch câu lệnh SQL.

Như vậy, chúng ta đã kết thúc việc trình bày về 17 tham số dùng cho việc điều hành vi của bộ tối ưu hóa. Mỗi tham số có những đặc điểm, chức năng riêng phù hợp với nhiều tình huống đa dạng phát sinh trong quá trình làm việc với cơ sở dữ liệu Oracle.

KẾT LUẬN VÀ LỜI CẢM ƠN

Như vậy, chúng em đã kết thúc báo cáo trình bày về bộ tối ưu hóa trong cơ sở dữ liệu Oracle. Dưới sự hướng dẫn của thầy giáo, TS. Trần Việt Trung, thông qua học phần Thiết kế và quản trị cơ sở dữ liệu, chúng em đã có được những kiến thức cơ bản, nền tảng để tìm hiểu và hoàn thành đề tài bài tập lớn này. Quá trình làm bài tập lớn đã giúp chúng em hiểu rộng và rõ hơn nhiều khía cạnh khác nhau của vấn đề tối ưu hóa trong cơ sở dữ liệu nói chung, Oracle Database nói riêng.

Bài tập lớn cũng là cơ hội để chúng em trau dồi hơn nữa khả năng ngoại ngữ và rèn luyện năng lực lĩnh hội tài liệu chuyên ngành công nghệ thông tin. Đây thực sự là những điều vô cùng bổ ích.

Chúng em xin chân thành cảm ơn thầy.

*
* *

TÀI LIỆU THAM KHẢO

[1] James Spiller, Tulika Srivastava, *Oracle Database 11g: SQL Tuning Workshop_Student Guide*, Oracle and/or its affiliates, 2010.

[2] Nikolaus Augsten, Dennis Shasha, Philippe Bonnet, slide bài giảng *Database Management and Performance Tuning*, 2014.