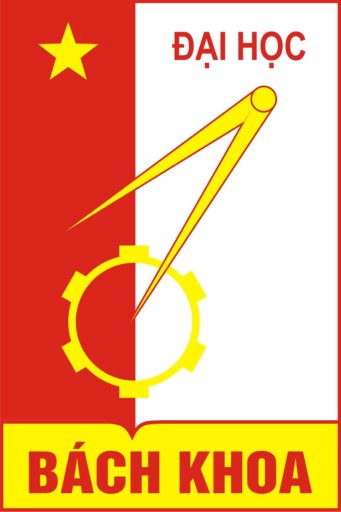
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**BÁO CÁO BÀI TẬP LỚN**

THIẾT KẾT VÀ QUẢN TRỊ CƠ SỞ DỮ LIỆU

Đề tài 16: Sử dụng gợi ý trong Oracle

Giáo viên hướng dẫn: TS. Trần Việt Trung

Sinh viên thực hiện: Lê Anh Tuấn – 20122675

Hà Nội, 04/05/2015

**Lời nói đầu**

Ngày nay công nghệ thông tin đã trở thành một phần không thể thếu trong bất cứ lĩnh vực nào của đời sống. Với việc ứng dụng các công nghệ trong quản lý, công việc của con người trở nên hiệu quả và năng suất hơn.

Với mục đích học tập tốt môn Thiết kế và quản trị cơ cở dữ liệu và bước đầu tiên tiếp cận với nhiều nền tảng trong công nghệ thông tin, em đã cố gắng tìm hiểu về đề tài sử dụng gợi ý trong Oracle.

Trong quá trình thực hiện bài tập lớn nên chắc chắn sẽ không tránh khỏi những sai sót. Em rất mong được thầy giúp đỡ và góp ý để bài tập lớn của em ngày càng hoàn thiện hơn.

Để hoàn thành được bài tập lớn này, em xin được gửi lời cảm ơn chân thành đến Tiến sĩ **Trần Việt Trung**, giảng viên bộ môn Hệ Thống Thông Tin, Viện Công Nghệ Thông Tin và Truyền Thông, trường Đại học Bách Khoa Hà Nội đã hết lòng giúp đỡ, hướng dẫn và góp ý để em có thể hoàn thành được bài tập lớn.

**Hà Nội, ngày 04 tháng 05 năm 2015.**

**Lê Anh Tuấn**

Mục lục

[**I.** **Những kiến thức cơ bản được trình bày trong báo cáo.** 5](#_Toc420396101)

[**1.** **B-Tree** 5](#_Toc420396102)

[**2.** **Các biến thể của B-Tree(B+-Tree, B\*-Tree)** 5](#_Toc420396103)

[**3.** **Hash** 6](#_Toc420396104)

[**4.** **Bitmap Index.** 7](#_Toc420396105)

[**II.** **Tổng quan về sử dụng gợi ý trong Oracle.** 8](#_Toc420396106)

[**1.** **Nội dung tìm hiểu.** 8](#_Toc420396107)

[**2.** **Tổng quan về sử dụng gợi ý.** 8](#_Toc420396108)

[**3.** **Các loại gợi ý.** 8](#_Toc420396109)

[**4.** **Chỉ định gợi ý.** 9](#_Toc420396110)

[**5.** **Các luật của gợi ý.** 9](#_Toc420396111)

[**6.** **Ví dụ về sử dụng gợi ý.** 10](#_Toc420396112)

[**III.** **Optimization Goals and Approaches** 11](#_Toc420396113)

[**1.** **FIRST\_ROWS Hint** 11](#_Toc420396114)

[**2.** **ALL\_ROWS Hint** 11](#_Toc420396115)

[**IV.** **Hints for Access Paths** 11](#_Toc420396116)

[**1.** **FULL Hint** 11](#_Toc420396117)

[**2.** **CLUSTER Hint** 12](#_Toc420396118)

[**3.** **HASH Hint** 12](#_Toc420396119)

[**4.** **INDEX Hint** 12](#_Toc420396120)

[**5.** **NO\_INDEX Hint** 13](#_Toc420396121)

[**6.** **INDEX\_ASC Hint** 13](#_Toc420396122)

[**7.** **INDEX\_DESC Hint** 13](#_Toc420396123)

[**8.** **INDEX\_COMBINE Hint** 14](#_Toc420396124)

[**9.** **INDEX\_JOIN Hint** 14](#_Toc420396125)

[**10.** **INDEX\_FFS Hint** 14](#_Toc420396126)

[**11.** **NO\_INDEX\_FFS Hint** 14](#_Toc420396127)

[**12.** **INDEX\_SS Hint** 15](#_Toc420396128)

[**13.** **NO\_INDEX\_SS Hint** 15](#_Toc420396129)

[**V.** **Hints for Query Transformation** 15](#_Toc420396130)

[**1.** **NO\_QUERY\_TRANSFORMATION Hint** 15](#_Toc420396131)

[**2.** **USE\_CONCAT Hint** 15](#_Toc420396132)

[**3.** **NO\_EXPAND Hint** 15](#_Toc420396133)

[**4.** **NO\_REWRITE Hint** 16](#_Toc420396134)

[**5.** **MERGE Hint** 16](#_Toc420396135)

[**6.** **NO\_MERGE Hints** 17](#_Toc420396136)

[**7.** **STAR\_TRANSFORMATION Hint.** 17](#_Toc420396137)

[**8.** **NO\_STAR\_TRANSFORMATION Hint** 17](#_Toc420396138)

[**9.** **FACT Hint** 18](#_Toc420396139)

[**VI.** **Hints for Join Orders** 18](#_Toc420396140)

[**1.** **LEADING Hint** 18](#_Toc420396141)

[**2.** **ORDERED Hint** 18](#_Toc420396142)

[**VII.** **Hints for Join Operations** 18](#_Toc420396143)

[**1.** **USE\_NL Hint** 18](#_Toc420396144)

[**2.** **NO\_USE\_NL Hint** 19](#_Toc420396145)

[**3.** **USE\_MERGE Hint** 19](#_Toc420396146)

[**4.** **NO\_USE\_MERGE Hint** 19](#_Toc420396147)

[**5.** **USE\_HASH Hint** 20](#_Toc420396148)

[**VIII.** **Tài liệu tham khảo** 20](#_Toc420396149)

1. **Những kiến thức cơ bản được trình bày trong báo cáo.**
2. **B-Tree**

Trong [khoa học máy tính](http://vi.wikipedia.org/wiki/Khoa_h%E1%BB%8Dc_m%C3%A1y_t%C3%ADnh), **B-tree** là một [cấu trúc dữ liệu](http://vi.wikipedia.org/wiki/C%E1%BA%A5u_tr%C3%BAc_d%E1%BB%AF_li%E1%BB%87u) dạng cây cho phép tìm kiếm, truy cập tuần tự, chèn, xóa trong thời gian lôgarit. B-tree là một tổng quát hóa của [cây nhị phân tìm kiếm](http://vi.wikipedia.org/wiki/C%C3%A2y_t%C3%ACm_ki%E1%BA%BFm_nh%E1%BB%8B_ph%C3%A2n), trong đó một nút có thể có nhiều hơn hai con. Không như cây nhị phân tìm kiếm, B-tree tự động cân bằng. B-tree được tối ưu hóa cho các hệ thống đọc và ghi dữ liệu lớn. B-tree thường được dùng trong các [cơ sở dữ liệu](http://vi.wikipedia.org/wiki/C%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u) và [hệ thống tập tin](http://vi.wikipedia.org/w/index.php?title=H%E1%BB%87_th%E1%BB%91ng_t%E1%BA%ADp_tin&action=edit&redlink=1).

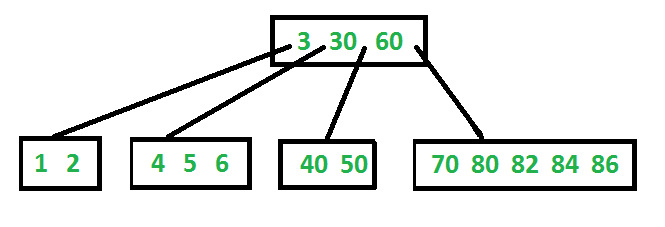
Trong B- tree, các nút trong (nút không là lá) có thể có số lượng nút con khác nhau, và được giới hạn trong một khoảng nhất định. Khi dữ liệu được chèn vào hoặc xóa đi từ một nút, số nút con của nó thay đổi. Khi đó, để duy trì số nút con trong khoảng đã định, các nút trong có thể được hợp hai làm một hoặc chia đôi. Vì số lượng nút con có thể nằm trong một khoảng lớn, B- tree không cần tái cân bằng thường xuyên như cây nhị phân tìm kiếm, nhưng lại sử dụng bộ nhớ lãng phí hơn do các nút không chứa tối đa dữ liệu.

B- tree duy trì tính cân bằng bằng cách đảm bảo các nút lá đều có cùng độ sâu. Chiều cao của cây tăng dần dần khi các nút mới được chèn vào cây, nhưng con số này thay đổi rất chậm. Khi chiều cao của cây tăng lên, độ sâu của tất cả các nút lá tăng lên cùng một lúc.

B- tree có lợi thế hơn các cấu trúc dữ liệu tìm kiếm khác khi thời gian truy cập lớn hơn nhiều lần thời gian đọc dữ liệu liên tiếp nhau. Điều này thường xảy ra khi các nút được lưu trên [bộ nhớ ngoài](http://vi.wikipedia.org/wiki/B%E1%BB%99_nh%E1%BB%9B#B.E1.BB.99_nh.E1.BB.9B_ngo.C3.A0i). Bằng cách tăng số lượng nút con của mỗi nút, chiều cao của cây giảm xuống và số lần truy cập cũng giảm. Thêm vào đó, số thao tác tái cân bằng cây cũng giảm đi. Thông thường, tham số *t* (quyết định số khóa của mỗi nút) được chọn tùy theo lượng thông tin trong mỗi khóa và kích thước mỗi [khối đĩa](http://vi.wikipedia.org/w/index.php?title=Kh%E1%BB%91i_%C4%91%C4%A9a&action=edit&redlink=1).

Số lần truy cập bộ nhớ trong mỗi thao tác trên B-Tree tỉ lệ với chiều cao của cây. Một B-Tree với *n* nút có chiều cao không quá .

Ví dụ về B-Tree:

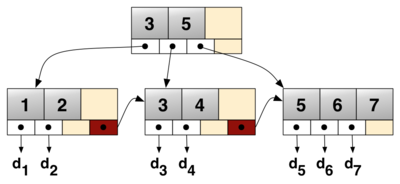


1. **Các biến thể của B-Tree(B+-Tree, B\*-Tree)**

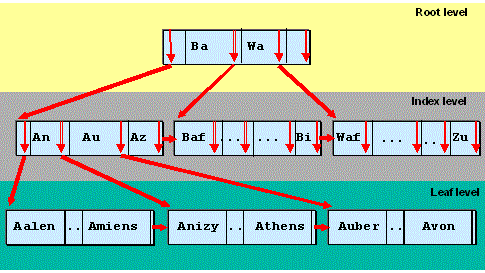
Thuật ngữ **B-** tree có thể được dùng để chỉ một thiết kế cụ thể, cũng có thể được dùng để chỉ một lớp các thiết kế. Theo nghĩa hẹp, B-tree lưu một số khóa ở các nút trong và không lưu bản sao của các khóa đó ở nút lá. Theo nghĩa rộng, nó còn bao gồm những biến thể khác như B+- tree hay B\*- tree.

* Trong B+- tree, các nút trong lưu bản sao của khóa. Tất cả các khóa, cùng với dữ liệu đi kèm được lưu ở các nút lá. Ngoài ra các nút lá còn có con trỏ đến các nút lá kế bên để tăng tốc truy cập tuần tự.
* B\*- tree thực hiện nhiều thao tác tái cân bằng hơn để lưu trữ dự liệu dày đặc hơn. Mỗi nút trong khác gốc phải đầy tới hai phần ba thay vì chỉ một nửa như trong B- tree.

Ví dụ về B+-Tree:



Ví dụ về B\*-Tree:



1. **Hash**

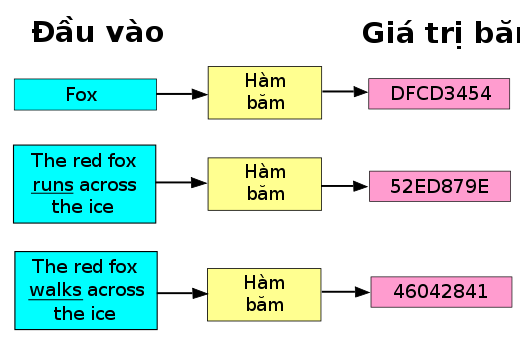
Hàm băm (*hash function*) là giải thuật nhằm sinh ra các giá trị băm tương ứng với mỗi khối dữ liệu. Giá trị băm đóng vai gần như một khóa để phân biệt các khối dữ liệu, tuy nhiên người ta chấp hiện tượng trùng khóa hay còn gọi là [đụng độ](http://vi.wikipedia.org/w/index.php?title=%C4%90%E1%BB%A5ng_%C4%91%E1%BB%99_b%C4%83m&action=edit&redlink=1) và cố gắng cải thiện giải thuật để giảm thiểu sự [đụng độ](http://vi.wikipedia.org/w/index.php?title=%C4%90%E1%BB%A5ng_%C4%91%E1%BB%99_b%C4%83m&action=edit&redlink=1) đó. Hàm băm thường được dùng trong [bảng băm](http://vi.wikipedia.org/wiki/B%E1%BA%A3ng_b%C4%83m) nhằm giảm chi phí tính toán khi tìm một khối dữ liệu trong một tập hợp (nhờ việc so sánh các giá trị băm nhanh hơn việc so sánh những *khối dữ liệu* có kích thước lớn).

Một hàm băm tốt phải thỏa mãn các điều kiện sau:

* Tính toán nhanh
* Các khoá được phân bố đều trong bảng.
* Ít xảy ra [đụng độ](http://vi.wikipedia.org/w/index.php?title=%C4%90%E1%BB%A5ng_%C4%91%E1%BB%99_b%C4%83m&action=edit&redlink=1).
* Xử lý được các loại khóa có kiểu dữ liệu khác nhau.

Một số hàm băm nổi tiếng: MD4, MD5(128bit), SHA-1(160 bit), SHA-256(256 bit), SHA-512(512-bit).

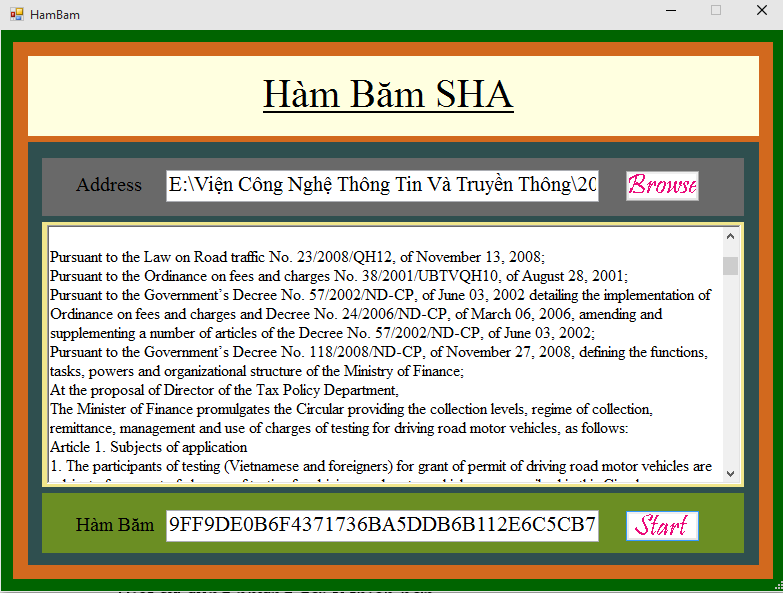
Ví dụ về hàm băm:



Kết quả của hàm băm SHA-1(do em tự cài đặt).

|  |  |
| --- | --- |
| Input | Output |
| Lê Anh Tuấn | FE863D1D40FBF78B38231AAFA4D4F82AAD879D19 |
| Cryptographic hash function | 442C19F8568C9E3748A5DA7CC591738A6831C815 |
| Cryptographic hash functions | 90D6AA5CB8666B0717DB58C1E9DB75027359FDC0 |

Hình ảnh chương trình do em cài đặt trên môi trường .NET.



1. **Bitmap Index.**

Lập chỉ mục đầy đủ của một bảng lớn với chỉ số B-tree truyền thống có thể gây tốn kém về không gian đĩa vì kích thước các chỉ số có thể lớn hơn gấp nhiều lần so với các dữ liệu trong bảng. Chỉ số bitmap thường chỉ có kích thước bằng một phần nhỏ trong kích thước của dữ liệu được lập chỉ mục trong bảng.

Mỗi bit trong bitmap có thể tương ứng với một rowid. Một chức năng lập bản đồ chuyển đổi các vị trí bit để trở thành một rowid thực tế, để các chỉ số bitmap cung cấp các chức năng tương tự như một chỉ số thông thường. Nếu số lượng các giá trị khác biệt là nhỏ, chỉ số bitmap nén tốt hơn và tiết kiệm không gian lợi ích so với một chỉ số B-cây trở nên tốt hơn.

Chỉ số bitmap được thiết kế chủ yếu cho các ứng dụng kho dữ liệu mà người dùng truy vấn dữ liệu hơn là cập nhật nó. Chứ bitmap không thích hợp cho các ứng dụng OLTP với số lượng lớn các giao dịch đồng thời sửa đổi dữ liệu.

Ví dụ về bitmap:



1. **Tổng quan về sử dụng gợi ý trong Oracle.**
2. **Nội dung tìm hiểu.**

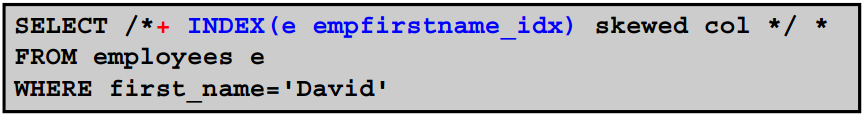
* Biết sử dụng những gợi ý thích hợp.
* Chỉ rõ gợi ý cho:
* Optimizer mode
* Query transformation
* Access path
* Join orders
* Join methods
* Views

1. **Tổng quan về sử dụng gợi ý.**

Sử dụng gợi ý làm ảnh hưởng đến các quyết định của tối ưu hóa. Cung cấp cho chúng ta một cơ chế chỉ đạo để lựa chọn một kế hoạch thực hiện câu truy vấn dựa trên các tiêu chí cụ thể.

Các dấu cộng (+) trong câu truy vấn nhằm giúp hệ thống hiểu đó là một danh sách các gợi ý. Các dấu cộng phải được thực hiện ngay sau các câu lệnh đầu tiên.

Ví dụ:



1. **Các loại gợi ý.**

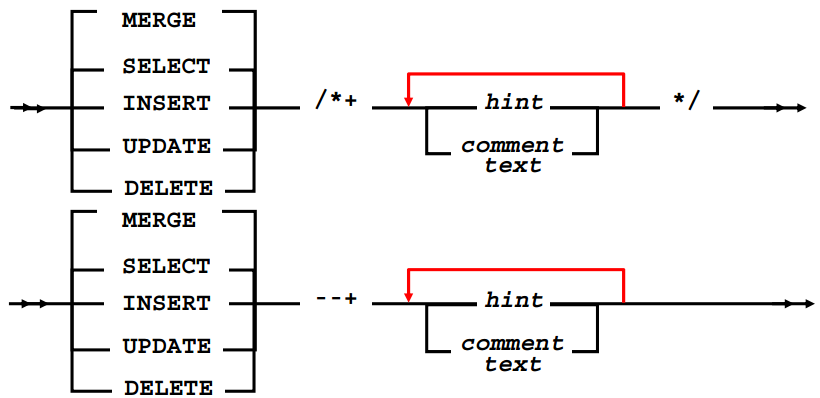
Có 4 loại gợi ý cơ bản như sau:

* Single-table: Gợi ý đơn bảng được chỉ định trên một bảng hoặc view. INDEX và USE\_NL là ví dụ về những gợi ý đơn bảng.
* Multitable: Gợi ý multitable giống như gợi ý đơn bảng, ngoại trừ những gợi ý có thể chỉ định một hoặc nhiều bảng hoặc khung nhìn. LEADING là một ví dụ về một gợi ý multitable.
* Query block: Chặn truy vấn gợi ý hoạt động trên các khối truy vấn đơn. STAR\_TRANSFORMATION và UNNEST là những ví dụ của các khối truy vấn gợi ý.
* Statement: Gợi ý được áp dụng cho toàn bộ câu lệnh SQL. ALL\_ROWS là một ví dụ về một gợi ý này.

Note: USE\_NL (table1 table2) không được coi là một gợi ý multitable bởi vì nó thực sự là một cách viết tắt cho USE\_NL (table1) và USE\_NL (table2).

1. **Chỉ định gợi ý.**

Gợi ý được áp dụng cho việc tối ưu hóa chỉ có một khối lệnh

Tối ưu hóa gợi ý chỉ xuất hiện trong khối câu lệnh(hoặc là một truy cấn cụ thể). Một khối câu lệnh có thể là:

* Một khối lệnh đơn giản MERGE, SELECT, INSERT, UPDATE, hoặc DELETE tuyên bố
* Một khối lệnh cha mẹ hoặc một truy vấn phụ của một câu lệnh phức tạp
* Một phần của một truy vấn phức hợp sử dụng các toán tử (UNION, MINUS, INTERSECT)

Ví dụ, một truy vấn hợp chất gồm hai truy vấn thành phần có hai khối được kết hợp của toán tử UNION, mỗi khối tương ứng với mỗi truy vấn thành phần. Vì lý do này, gợi ý trong các truy vấn thành phần đầu tiên chỉ áp dụng để tối ưu hóa của nó, không phải để tối ưu hóa các truy vấn thành phần thứ hai.

Bạn có thể sử dụng một trong hai phong cách của chú thích. Các dấu phân cách gợi ý (+) phải đến ngay lập tức sau khi các dấu phân cách nhận xét. Nếu bạn tách chúng bằng một dấu cách, bộ tối ưu không nhận ra rằng những chú thích có gợi ý.

1. **Các luật của gợi ý.**

* Nơi gợi ý ngay sau từ khóa SQL đầu tiên của một khối lệnh.
* Mỗi khối câu lệnh có thể chỉ có một gợi ý chú thích, nhưng nó có thể chứa nhiều gợi ý.
* Gợi ý chỉ áp dụng cho khối lệnh mà chúng xuất hiện.
* Nếu một tuyên bố sử dụng bí danh, gợi ý phải tham khảo các bí danh chứ không phải là tên bảng.
* Các tối ưu bỏ qua những gợi ý không chính xác mà không cần báo lỗi.

Bạn phải đặt các bình luận gợi ý ngay lập tức sau khi các từ khóa đầu tiên (MERGE, SELECT, INSERT, DELETE, hoặc UPDATE) của một khối câu lệnh SQL.

Một khối câu lệnh có thể chỉ có một nhận xét có chứa các gợi ý, nhưng nó có thể chứa nhiều gợi ý bên trong chú thích bằng dấu cách.

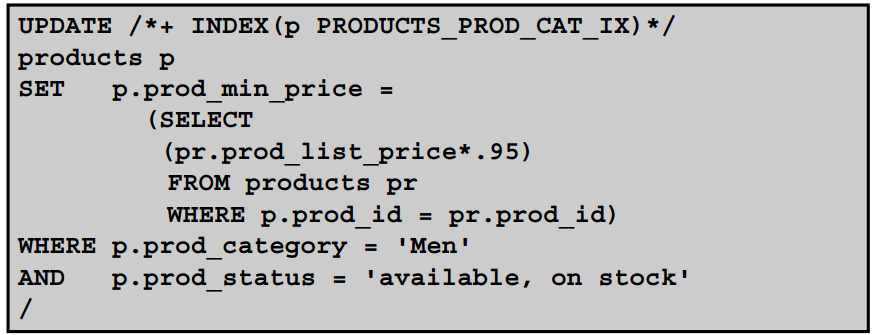
Nếu một câu lệnh SQL sử dụng bí danh, gợi ý phải tham khảo các bí danh chứ không phải là tên bảng.

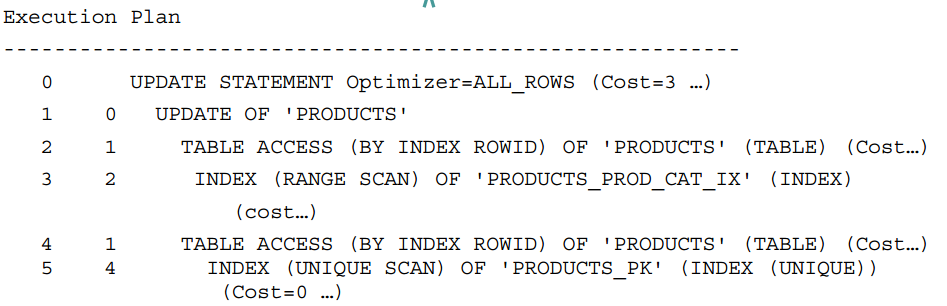
Các bộ tối ưu của Oracle bỏ qua những gợi ý cụ thể không chính xác. Tuy nhiên, cần chú ý các tình huống sau đây:

* Không bao giờ nhận được một thông báo lỗi.
* Bộ tối ưu Oracle cũng bỏ qua sự kết hợp của những gợi ý trái ngược nhau.

1. **Ví dụ về sử dụng gợi ý.**

Cho một khối lệnh như sau:

****Kế hoạch thực hiện như sau:

Những gợi ý trong ví dụ chỉ hoạt động nếu một chỉ số gọi là PRODUCTS\_PROD\_CAT\_IX tồn tại trên bảng PRODUCTION trong cột PROD\_CATEGORY.

1. **Optimization Goals and Approaches**
2. **FIRST\_ROWS Hint**

Description of first_rows_hint.gif follows

Bộ tối ưu hóa câu lệnh SQL lựa chọn phương án trả về kết quả với n hàng đầu tiên, một cách hiệu quả nhất.

Sử dung phương pháp tối ưu hóa để tối ưu các khối lệnh và trả về kết quả với thời gian đáp ứng tốt nhất: Ví dụ

SELECT /\*+ FIRST\_ROWS(10) \*/ employee\_id, last\_name, salary, job\_id

FROM employees

WHERE department\_id = 20;

Trong ví dụ này, mỗi văn phòng có chứa nhiều nhân viên. Nhưng người dùng muốn lấy ra 10 nhân viên đầu tiên của văn phòng có ID bằng 20 sẽ được hiển thị càng nhanh càng tốt.

1. **ALL\_ROWS Hint**

Description of all_rows_hint.gif follows  
Các ALL\_ROWS gợi ý chỉ thị các ưu để tối ưu hóa một khối lệnh với một mục tiêu của thông lượng tốt nhất, đó là tổng số tiêu thụ tài nguyên tối thiểu. Ví dụ, bộ tối ưu sử dụng phương pháp tối ưu hóa truy vấn để tối ưu hóa câu lệnh này cho kết quả tốt nhất:

SELECT /\*+ ALL\_ROWS \*/ employee\_id, last\_name, salary, job\_id

FROM employees

WHERE employee\_id = 107;

Nếu bạn chỉ định một trong hai ALL\_ROWS hoặc FIRST\_ROWS gợi ý trong một câu lệnh SQL, và nếu các dữ liệu từ điển không có số liệu thống kê về các bảng truy cập bởi các khối lệnh, sau đó tối ưu hóa sử dụng được giá trị thống kê mặc định, chẳng hạn như lưu trữ được phân bổ cho các bảng như vậy, để ước tính số liệu thống kê và sau đó chọn một kế hoạch thực hiện. Những ước tính này có thể không được chính xác như những tập hợp của các gói DBMS\_STATS, vì vậy bạn nên sử dụng các gói DBMS\_STATS để thu thập số liệu thống kê.

Nếu bạn chỉ định gợi ý cho đường dẫn truy cập hoặc tham gia các hoạt động cùng với một trong hai ALL\_ROWS hoặc FIRST\_ROWS gợi ý, sau đó tối ưu hóa cho quyền ưu tiên cho các đường dẫn truy cập và tham gia các hoạt động được quy định bởi các gợi ý.

1. **Hints for Access Paths**
2. **FULL Hint**

Description of full_hint.gif follows

Các gợi ý FULL yêu cầu bộ tối ưu thực hiện một phương án quét toàn bộ bảng quy định. Ví dụ như:

SELECT /\*+ FULL(e) \*/ employee\_id, last\_name

FROM hr.employees e

WHERE last\_name LIKE :b1;

Cơ sở dữ liệu Oracle thực hiện một phương án quét toàn bộ trên các bảng nhân viên để thực hiện câu lệnh này.

Các bảng nhân viên có bí danh là e trong mệnh đề FROM, vì vậy những gợi ý phải tham khảo bảng bằng bí danh của nó chứ không theo tên của bảng ban đầu. Không nên dùng tên lược đồ trong những gợi ý ngay cả khi họ được sử dụng trong mệnh đề FROM.

1. **CLUSTER Hint**

Description of cluster_hint.gif follows

Các gợi ý CLUSTER chỉ thị bộ tối ưu để sử dụng một phương án quét theo cụm để truy cập vào các bảng được quy định sẵn. Lời gợi ý này chỉ áp dụng cho các bảng trong một cụm được cài đặt chỉ mục.

1. **HASH Hint**

Description of hash_hint.gif follows

Các gợi ý HASH chỉ thị bộ tối ưu để sử dụng một phương án quét theo hàm băm để truy cập vào các bảng quy định có sẵn trước đó. Lời gợi ý này chỉ nên sử dụng cho các bảng trong một hàm băm.

1. **INDEX Hint**

Description of index_hint.gif follows

Các gợi ý INDEX chỉ thị bộ tối ưu để sử dụng phương án quét theo chỉ số theo các bảng đã được quy định sẵn. Bạn có thể sử dụng các gợi ý cho INDEX dựa trên chức năng, B-Tree, Bitmap.

Các hành vi của các gợi ý phụ thuộc vào các đặc điểm kỹ thuật indexspec:

Nếu những gợi ý INDEX có sẵn một chỉ số duy nhất, sau đó cơ sở dữ liệu thực hiện việc quét trên các chỉ số này. Việc tối ưu hóa sẽ không xem xét các phương án quét toàn bộ bảng hoặc một phương án quét trên một chỉ số khác của bảng.

Đối với một gợi ý có sự kết hợp của nhiểu chỉ số, Oracle khuyến cáo nên sử dụng INDEX\_COMBINE hơn là sử dụng INDEX,bởi vì INDEX\_COMBINE là một gợi ý linh hoạt hơn. Nếu những gợi ý INDEX xác định danh sách các chỉ mục, sau đó tối ưu hóa các chi phí của một phương án quét trên mỗi chỉ số trong danh sách chỉ số ban đầu, sau đó thực hiện các chỉ số với phương án quét là thấp nhất. Các cơ sở dữ liệu cũng có thể lựa chọn để quét nhiều chỉ mục từ danh sách này và kết hợp các kết quả, tìm ra một con đường tiếp cận có chi phí thấp nhất. Các cơ sở dữ liệu không xem xét phương án quét toàn bộ bảng hoặc một phương án quét trên một chỉ số mà không được liệt kê trong các gợi ý.

Nếu những gợi ý INDEX mặc định không có chỉ số, sau đó tối ưu hóa các chi phí của một phương án quét trên mỗi chỉ số có sẵn trong bảng và tìm ra các chỉ số với chi phí quét là thấp nhất. Các cơ sở dữ liệu cũng có thể chọn để quét nhiều chỉ mục và hợp nhất các kết quả đó lại với nhau, và tìm ra một phương án với chi phí là thấp nhất. Việc tối ưu hóa không xem xét việc quét trên toàn bộ bảng.

For example:

SELECT /\*+ INDEX (employees emp\_department\_ix)\*/ employee\_id, department\_id

FROM employees

WHERE department\_id > 50;

1. **NO\_INDEX Hint**

Description of no_index_hint.gif followsCác gợi ý NO\_INDEX chỉ thị bộ tối ưu không sử dụng một hoặc nhiều chỉ số cho các bảng quy định. Ví dụ như:

SELECT /\*+ NO\_INDEX(employees emp\_empid) \*/ employee\_id

FROM employees

WHERE employee\_id > 200;

Mỗi tham số phục vụ cùng một mục đích như trong "INDEX" với các sửa đổi sau đây:

Nếu gợi ý này chỉ định một chỉ số có sẵn duy nhất, sau đó tối ưu hóa không xem xét phương án quét trên các chỉ số này. Những chỉ số khác không được quy định thì vẫn được xem xét.

Các gới ý NO\_INDEX áp dụng cho các chức năng như B-tree, bitmap, Cluster hoặc Domain Index. Nêu một gợi ý NO\_INDEX và một số gợi (INDEX, INDEX\_ASC, INDEX\_DESC, INDEX\_COMBINE, hoặc INDEX\_FFS) cả hai đều xác định các chỉ số tương tự.

1. **INDEX\_ASC Hint**

Description of index_asc_hint.gif followsCác gợi ý INDEX\_ASC chỉ thị các tối ưu để sử dung một phương án quét cho các bảng quy định. Nếu khối lệnh sử dụng một phạm vi quét theo các chỉ mục, sau đó cơ sở dữ liệu Oracle quét theo chỉ mục trong thứ tự tăng dần của các giá trị được lập chỉ mục. Mỗi tham số phục vụ có cùng một mục đích như trong “INDEX”.

Hành vi mặc định cho một phạm vi quét để quét chỉ mục theo thứ tự tăng dần của các giá trị được lập chỉ mục, hoặc thứ tự giảm dần cho một chỉ số giảm dần. Lời gợi ý này không làm thay đổi thứ tự mặc định của chỉ số, và do đó không thể xác định được gì hơn những gợi ý INDEX. Tuy nhiên, bạn có thể sử dụng gợi ý INDEX\_ASC để xác định phạm vi quét tăng dần.

1. **INDEX\_DESC Hint**

Description of index_desc_hint.gif followsCác gợi ý INDEX\_DESC chỉ thị các tối ưu sử dụng một chỉ số quét giảm dần của các bảng theo quy định. Nếu khối lệnh sử dụng một phạm vi quét chỉ mục của các chỉ số tăng dần, sau đó Oracle sẽ quét chỉ mục theo thứ tự giảm dần của các giá trị được lập chỉ mục. Đối với một chỉ số giảm dần, gợi ý này có hiệu quả triệt tiêu thứ tự giảm dần, kết quả là một phương án quét của chỉ mục theo thứ tự tăng dần.

SELECT /\*+ INDEX\_DESC(e emp\_name\_ix) \*/ \*

FROM employees e;

1. **INDEX\_COMBINE Hint**

Description of index_combine_hint.gif followsCác gợi ý INDEX\_COMBINE chỉ thị các tối ưu để sử dụng một đường dẫn truy cập bitmap cho bảng. Nếu indexes được bỏ qua từ gợi ý INDEX\_COMBINE, sau đó tối ưu hóa sử dụng bất cứ logic nào để kết hợp của các chỉ số có dự đoán chi phí tốt nhất của bảng. Nếu bạn chỉ định indexes sau đó cố gắng tối ưu hóa để sử dụng một số kết hợp logic của các chỉ số xác định. Mỗi tham số phụ thuộc cũng một mục đích như trong “INDEX”. Ví dụ như:

SELECT /\*+ INDEX\_COMBINE(e emp\_manager\_ix emp\_department\_ix) \*/ \*

FROM employees e

WHERE manager\_id = 108

OR department\_id = 110;

1. **INDEX\_JOIN Hint**

Description of index_join_hint.gif followsCác gợi ý INDEX\_JOIN chỉ thị các tối ưu để sử dụng một chỉ số tham gia như một đường dẫn truy cập.

Mỗi tham số cùng một mục đích như trong “INDEX”. Ví dụ truy vấn sau đây sử dụng một chỉ số tham gia truy cập các cột department\_id và Manager\_ID, cả hai đều được lập chỉ mục trong bảng employees.

SELECT /\*+ INDEX\_JOIN(e emp\_manager\_ix emp\_department\_ix) \*/ department\_id

FROM employees e

WHERE manager\_id < 110

AND department\_id < 50;

1. **INDEX\_FFS Hint**

Description of index_ffs_hint.gif followsGợi ý INDEX\_FFS gợi ý chỉ thị tối ưu để thực hiện nhanh chóng một phương án quét chỉ số thay vì một bảng quét đầy đủ.

Mỗi tham số phục vụ cùng một mục đích như trong "INDEX Hint”. Ví dụ như:

SELECT /\*+ INDEX\_FFS(e emp\_name\_ix) \*/ first\_name

FROM employees e;

1. **NO\_INDEX\_FFS Hint**

Description of no_index_ffs_hint.gif followsGợi ý NO\_INDEX\_FFS chỉ thị các tối ưu để loại trừ nhanh chóng phương án quét trên các chỉ số đầy đủ được xác định trên bảng quy định. Mỗi tham số phục vụ cùng một mục đích như trong “NO\_INDEX”. Ví dụ như:

SELECT /\*+ NO\_INDEX\_FFS(items item\_order\_ix) \*/ order\_id

FROM order\_items items;

1. **INDEX\_SS Hint**

Description of index_ss_hint.gif followsCác gợi ý INDEX\_SS chỉ thị các tối ưu để thực hiện một chỉ số quét bỏ qua cho các bảng quy định. Nếu khối lệnh sử dụng một phạm vi quét chỉ mục, sau đó Oracle quét chỉ mục theo thứ tự tăng dần của các giá trị được lập chỉ mục.

Mỗi tham số phục vụ cùng một mục đích như trong "INDEX Gợi ý". Ví dụ như:

SELECT /\*+ INDEX\_SS(e emp\_name\_ix) \*/ last\_name

FROM employees e

WHERE first\_name = 'Steven';

1. **NO\_INDEX\_SS Hint**

Description of no_index_ss_hint.gif follows

Gợi ý NO\_INDEX\_SS chỉ thị các tối ưu để loại trừ một khả năng quét của các chỉ số xác định trên bảng quy định. Mỗi tham số phục vụ cùng một mục đích như trong “NO\_INDEX”.

1. **Hints for Query Transformation**
2. **NO\_QUERY\_TRANSFORMATION Hint**

Description of no_query_transformatn_hint.gif follows

SELECT /\*+ NO\_QUERY\_TRANSFORMATION \*/ employee\_id, last\_name

FROM (SELECT \* FROM employees e) v

WHERE v.last\_name = 'Smith';

1. **USE\_CONCAT Hint**

Description of use_concat_hint.gif follows

Các gợi ý USE\_CONCET chỉ thị các tối ưu để biến đổi kết hợp hoặc điều kiện trong mệnh đề WHERE của một câu truy vấn vào một câu truy vấn bằng cách sử dụng hợp nhất UNION ALL. Ví dụ như:

SELECT /\*+ USE\_CONCAT \*/ \*

FROM employees e

WHERE manager\_id = 108

OR department\_id = 110;

1. **NO\_EXPAND Hint**

Description of no_expand_hint.gif follows  
Các gợi ý NO\_EXPAND chỉ thị các tối ưu không xem xét hoặc mở rộng cho các câu truy vấ có điều kiện OR hoặc IN trong danh sách mệnh đề WHERE. Thông thường thì các tối ưu xem xét sử dụng hoặc mở rộng và sử dụng phương pháp này nếu nó quyết định rằng chi phí thấp hơn so với việc không sử dụng nó. Ví dụ như:

SELECT /\*+ NO\_EXPAND \*/ \*

FROM employees e, departments d

WHERE e.manager\_id = 108

OR d.department\_id = 110;

1. **NO\_REWRITE Hint**

Description of no_rewrite_hint.gif follows

Các gợi ý NO\_REWRITE chỉ thị các tối ưu để vô hiệu hóa truy vấn viết lại cho các khối truy vấn, thiết lập ghi đè các tham số QUERY\_REWRITE\_ENABLED. Ví dụ như:

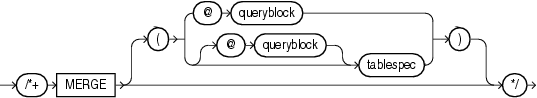
SELECT /\*+ NO\_REWRITE \*/ sum(s.amount\_sold) AS dollars

FROM sales s, times t

WHERE s.time\_id = t.time\_id

GROUP BY t.calendar\_month\_desc

1. **MERGE Hint**

Các gợi ý MERGE cho phép bạn hợp nhất quan điểm trong một truy vấn.

Nếu khối truy vấn của người xem có chứa một mệnh đề GROUP hoặc nhà điều hành riêng biệt trong danh sách SELECT BY, sau đó tối ưu hóa có thể kết hợp xem vào câu lệnh truy nếu phức tạp nhìn nhận sáp nhập được kích hoạt. Sáp nhập phức tạp cũng có thể được sử dụng để hợp nhất một subquery IN vào câu lệnh truy vấn subquery.

Ví dụ như:

SELECT /\*+ MERGE(v) \*/ e1.last\_name, e1.salary, v.avg\_salary

FROM employees e1,

(SELECT department\_id, avg(salary) avg\_salary

FROM employees e2

GROUP BY department\_id) v

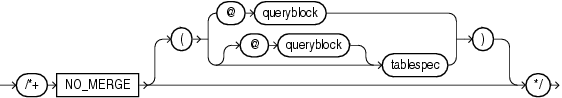
WHERE e1.department\_id = v.department\_id

AND e1.salary > v.avg\_salary

ORDER BY e1.last\_name;

Khi gợi ý MERGE được sử dụng mà không có một đối số, nó nên được đặt trong khối xem truy vấn. Khi MERGE được sử dụng với tên xem như một tham số, nó nên được đặt trong các truy vấn xung quanh.

1. **NO\_MERGE Hints**

Các gợi ý NO\_MERGE chỉ thị các ưu không kết hợp các truy vấn bên ngoài và bất kỳ điểm truy vấn nội tuyến vào một truy vấn đơn.

Lời gợi ý này cho phép bạn có nhiều ảnh hưởng hơn so với cách thức mà các điểm truy cập. Ví dụ, câu lệnh sau sẽ khiến khung nhìn seattle\_dept không được sáp nhập:

SELECT /\*+ NO\_MERGE(seattle\_dept) \*/ e1.last\_name, seattle\_dept.department\_name

FROM employees e1,

(SELECT location\_id, department\_id, department\_name

FROM departments

WHERE location\_id = 1700) seattle\_dept

WHERE e1.department\_id = seattle\_dept.department\_id;

Khi bạn sử dụng các gợi ý NO\_MERGE trong view chặn truy vấn, xác định nó không có một đối số. Khi bạn xác định NO\_MERGE trong truy vấn xung quanh, xác định nó với các tên xem như một đối số.

1. **STAR\_TRANSFORMATION Hint.**

Description of star_transformation_hint.gif follows

Các gợi ý STAR\_ TRANSFORMATION chỉ thị tối ưu để sử dụng phương án tốt nhất, trong đó việc chuyển đổi đã được sử dụng. Nếu không có những gợi ý, các ưu có thể đưa ra quyết định tối ưu hóa truy vấn để sử dụng phương án tốt nhất được tạo ra mà không có sự chuyển đổi, thay vì phương án tốt nhất cho chuyển đổi câu truy vấn. Ví dụ như:

SELECT /\*+ STAR\_TRANSFORMATION \*/ s.time\_id, s.prod\_id, s.channel\_id

FROM sales s, times t, products p, channels c

WHERE s.time\_id = t.time\_id

AND s.prod\_id = p.prod\_id

AND s.channel\_id = c.channel\_id

AND c.channel\_desc = 'Tele Sales';

1. **NO\_STAR\_TRANSFORMATION Hint**

Description of no_star_transformation_hint.gif follows

Các gợi ý NO\_STAR\_TRANSFORMATION chỉ thị bộ tối ưu không thực hiện chuyển đổi truy vấn.

1. **FACT Hint**

Description of fact_hint.gif follows

Các gợi ý FACT được sử dụng trong bối cảnh của sự chuyển đổi. Nó chỉ thị bộ tối ưu rằng các bảng quy định trong tablespec cần được xem xét như là một bảng thực tế.

1. **NO\_FACT Hint**

Description of no_fact_hint.gif follows

Các gợi ý NO\_FACT được sử dụng trong bối cảnh của sự chuyển đổ. Nó hướng dẫn tối ưu hóa các bảng truy vấn không nên xem là một bảng thực tế.

1. **Hints for Join Orders**
2. **LEADING Hint**

Description of leading_hint.gif follows

Các gợi ý LEADING chỉ thị bộ tối tối ưu để sử dụng các bộ quy định của bảng là tiền tố trong kế hoạch thực hiện. Điều này là linh hoạt hơn so với gợi ý ORDERED. Ví dụ như:

SELECT /\*+ LEADING(e j) \*/ \*

FROM employees e, departments d, job\_history j

WHERE e.department\_id = d.department\_id

AND e.hire\_date = j.start\_date;

Các gợi ý LENDING được bỏ qua nếu các bảng quy định không thể được tham gia đầu tiên trong thứ tự quy định vì phụ thuộc vào đồ thị tham gia. Nếu bạn chỉ định nhiều hơn hai mâu thuẫn trong gợi ý LEADING thì chúng đều được bỏ qua. Nếu bạn chỉ định những gợi ý ORDERED nó sẽ ghi đè tất cả các gợi ý LEADING.

1. **ORDERED Hint**

Description of ordered_hint.gif follows

Các gợi ý ORDERED chỉ thị Oracle tham gia bảng trong thứ tự mà chúng xuất hiện trong mệnh đề FROM. Oracle khuyến cáo bạn sử dụng gợi ý LEADING vì linh hoạt hơn so với các gợi ý ORDERED.

Các truy vấn sau đây là một ví dụ về việc sử dụng các gợi ý ORDERED:

SELECT /\*+ ORDERED \*/ o.order\_id, c.customer\_id, l.unit\_price \* l.quantity

FROM customers c, order\_items l, orders o

WHERE c.cust\_last\_name = 'Taylor'

AND o.customer\_id = c.customer\_id

AND o.order\_id = l.order\_id;

1. **Hints for Join Operations**
2. **USE\_NL Hint**

Description of use_nl_hint.gif follows

Sử dụng các gợi ý USER\_NL và USER\_MERGE được khuyến cáo với gợi ý LEADING và ORDERED. Việc tối ưu hóa sử dụng những gợi ý khi bảng tham chiếu buộc phải được các bảng bên trong tham gia. Các gợi ý được bỏ qua nếu các bảng tham chiếu là bảng bên ngoài.

Trong ví dụ sau đây, nơi một vòng lặp lồng nhau được buộc phải thông qua một gợi ý, các ORDERED được truy cập thông qua một bảng quét toàn bộ và các điều kiện lọc l.order\_id = h.order\_id được áp dụng cho mỗi ORDERED. Đối với mỗi hàng đáp ứng các điều kiện lọc order\_items được truy cập thông qua các chỉ số order\_id.

SELECT /\*+ USE\_NL(l h) \*/ h.customer\_id, l.unit\_price \* l.quantity

FROM orders h, order\_items l

WHERE l.order\_id = h.order\_id;

1. **NO\_USE\_NL Hint**

Description of no_use_nl_hint.gif followsSELECT /\*+ NO\_USE\_NL(l h) \*/ \*

FROM orders h, order\_items l

WHERE l.order\_id = h.order\_id

AND l.order\_id > 2400;

Tuy nhiên, trong một số trường hợp bảng có thể được tham gia chỉ bằng cách sử dụng các vòng lặp lồng nhau. Trong trường hợp như vậy, bộ tối ưu bỏ qua gợi ý cho những bảng.

1. **USE\_MERGE Hint**

Description of use_merge_hint.gif follows  
Các gợi ý USE\_MERGE chỉ thị bộ tối ưu để tham gia mỗi bảng quy định với một nguồn dòng bằng một loại hợp nhất tham gia. Ví dụ như:

SELECT /\*+ USE\_MERGE(employees departments) \*/ \*

FROM employees, departments

WHERE employees.department\_id = departments.department\_id;

Sử dụng các USE\_NL và USE\_MERGE gợi ý được khuyến cáo với gợi ý LEADING và ORDERED. Việc tối ưu hóa sử dụng những gợi ý khi bảng tham chiếu buộc phải được các bảng bên trong của một tham gia. Các gợi ý được bỏ qua nếu các bảng tham chiếu là bảng bên ngoài.

1. **NO\_USE\_MERGE Hint**

Description of no_use_merge_hint.gif follows  
Các gợi ý NO\_USE\_MERGE chỉ thị bộ tối ưu để loại trừ các loại hợp nhất tham gia khi tham gia mỗi bảng quy định đến một nguồn bằng bảng quy định như trong bảng. Ví dụ như:

SELECT /\*+ NO\_USE\_MERGE(e d) \*/ \*

FROM employees e, departments d

WHERE e.department\_id = d.department\_id

ORDER BY d.department\_id;

1. **USE\_HASH Hint**

Description of use_hash_hint.gif follows  
Các gợi ý USE\_HASH chỉ thị bộ tối ưu để tham gia mỗi bảng quy định với một nguồn bằng một hàm băm tham gia. Ví dụ như:

SELECT /\*+ USE\_HASH(l h) \*/ \*

FROM orders h, order\_items l

WHERE l.order\_id = h.order\_id

AND l.order\_id > 2400;

1. **Tài liệu tham khảo**
2. Oracle Database 11g: SQL Tuning Workshop
3. Slide Thiết kế và quản trị cơ sở dữ liệu, TS. Vũ Tuyết Trinh, bộ môn hệ thống thông tin, viện công nghệ thông tin, trường Đại Học Bách Khoa Hà Nội.
4. Database Management and Tuning, Johann Gamper, Free University of Bozen-Bolzano Faculty of Computer Science IDSE.