BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

LUẬN VĂN THẠC SĨ KHOA HỌC

NGÀNH: CÔNG NGHỆ THÔNG TIN

TRUY VẤN DỮ LIỆU HƯỚNG NGƯỜI DÙNG

HOÀNG NGUYÊN HÙNG

HÀ NỘI 2006

MỤC LỤC

LỜI GIỚI THIỆU

Công nghệ thông tin này càng trở nên quan trọng trong đời sống chúng ta và là một phần không thể thiếu trong cuộc sống hiện đại. Thông tin điện tử ngày càng trở nên phong phú và trải rộng ra hầu hết các lính vực từ khoa học cho đến thương mại. Do đó dữ liệu trở nên quá đồ sộ và việc khai thác nguồn thông tin này đứng trước tình trạng có nguy cơ khó khăn hơn. Từ đó đặt ra một thách thức cho công nghệ cơ sở dữ liệu, đòi hỏi một hệ cơ sở dữ liệu mạnh mẽ và mô hình công nghệ mềm dẻo cho phù hợp với những yêu cầu của người dùng. Người dùng luôn luôn mong ước có được những thông tin cần thiết, thỏa mãn những ước muốn của họ đưa ra, điều này đòi hỏi chúng ta phải có một mô hình dữ liệu gần gũi với người dùng, cụ thể hơn, yêu cầu có một mô hình dữ liệu ưa thích mềm dẻo. Các truy vấn ưa thích phải thoả mãn sự hợp tác bởi ưa thích nghiên cứu như là ràng buộc không bắt buộc, cố gắng có được sự phù hợp tốt nhất khi thực hiện yêu cầu. Chúng ta đề xuất một ngữ nghĩa thứ tự bộ phận nghiêm ngặt cho ưa thích, nó có sự phù hợp gần gũi với trực quan của con người. Sự đa dạng của tự nhiên và của ưa thích phức tạp được bao trùm trong mô hinh này. Chúng tôi đưa ra một cấu trúc quy nạp cho ưa thích phức hợp bởi ý nghĩa của các cấu trúc ưa thích khác nhau. Mô hình này là chìa khóa cho một hướng nghiên cứu mới gọi là kỹ nghệ ưa thích và đại số ưa thích. Mô hình truy vấn phù hợp nhất đã cho, chúng ta sẽ thấy các truy vấn phức tạp có thể được biến đổi về các truy vấn đơn giản hơn. Chúng tôi tin rằng mô hình này là thích hợp với công nghệ cơ sở dữ liệu mở rộng theo hướng hỗ trợ hiệu quả hơn cho cá nhân hóa thông tin

Các công cụ tìm kiếm hiện tại có thể hầu như không phù hợp với sở thích phức tạp. Vấn đề lớn nhất của bộ máy tìm kiếm thực hiện với SQL chuẩn là SQL không có khả năng hiểu được khái niệm của sự ưa thích. SQL ưa thích mở rộng SQL chuẩn bởi mô hình ưa thích dựa trên ràng buộc không bắt buộc, lúc đó các truy vấn ưa thích sẽ xử sự như là các ràng buộc lựa chọn mềm.

Lợi ích của công nghệ SQL ưa thích bao gồm trả lời truy vấn và đưa ra lời khuyên thông minh cho khách hàng, đi đầu là thoả mãn yêu cầu từ người dùng mua

bán trực tuyết ở mức cao hơn và thời gian phát triển ngắn của các bộ máy tìm kiếm hướng người dùng cho người cung cấp các dịch vụ điện tử.

Từ những nhận định trên, tôi muốn trình bày một cách rõ ràng về vấn đề truy vấn ưa thích. Để có thể thực hiện được điều này, tôi đã nghiên cứu các tài liệu liên quan và tổng kết lại những hiểu biết của tôi về truy vấn hướng người dùng và tập trung vào truy vấn ưa thích. Toàn bộ luận văn này được trình bày như sau:

Chương I: Trình bày tổng quan về hệ cơ sở dữ liệu hướng người dùng, bao gồm giới thiệu về cơ bản của sự ưa thích, biểu diễn mô hình ưa thích như là chìa khóa của kỹ nghệ ưa thích, phát triển đại số ưa thích và trình bày một số thuật toán xử lý cho truy vấn ưa thích.

Chương II. Nghiên cứu về tối ưu hóa truy vấn ưa thích trong cơ sở dữ liệu quan hệ, bao gồm giới thiệu đại số quan hệ ưa thích và thiết kế kiến trúc cho tối ưu truy vấn ưa thích, chương này cũng trình bày về tối ưu đại số cho truy vấn ưa thích và các ứng dụng thực tế..

Chương III. Trình bày về SQL ưa thích: Bao gồm vấn đề thiết kế ngôn ngữ SQL ưa thích và môi trường thực thi của SQL ưa thích.

Kết quả, Đây là một hướng đi mới cho công nghệ cơ sở dữ liệu hướng người dùng, nghiên cứu sẽ là một phần trợ giúp đắc lực cho các nhà phát triển ứng dụng, hỗ trợ họ cho các vấn đề ra quyết định, cấu hình và áp dụng các ứng dụng cơ sở dữ liệu vào thực tiễn được tốt hơn, làm cho các ứng dụng ngày càng thân thiện hơn với người dùng.

Chương I. TỔNG QUAN VỀ HỆ THỐNG CƠ SỞ DỮ LIỆU HƯỚNG NGƯỜI DÙNG.

1.1. Giới thiệu.

Sự ưa thích diễn ra mọi nơi trong cuộc sống hàng ngày của chúng ta. Và gần đây, chúng được chú ý nhiều đến trong kỹ nghệ phát triển phần mềm, điển hình là được ứng dụng nhiều trong các ứng dụng dịch vụ điện tử hướng người dùng. Do đó nó trở thành một sự thách thức cho công nghệ cơ sở dữ liệu nhằm tương xứng với nhiều diện mạo phức tạp của sự ưa thích. Cá nhân hóa có nhiều khía cạnh khác nhau: Có một thế giới thực, nơi người sử dụng mong muốn có thể thỏa mãn hoặc không với tất cả. Trong trường hợp này người sử dụng lựa chọn bị hạn chế tới một tập giới hạn trước của các lựa chọn phức tạp, ví dụ: các cấu hình phần mềm tùy thuộc vào tiểu sử người dùng. Cơ sở dữ liệu truy vấn trong ngữ cảnh này là được cá nhân hóa bởi sự ràng buộc chặt chẽ, thực hiện chính xác những đối tượng mơ ước nếu chúng là có và trong trường hợp khác sẽ từ chối những yêu cầu của người dùng. Nhưng trong thế giới thực, nơi mà sự ưa thích cá nhân có sự khác nhau. Như là sự ưa thích được hiểu là sự ước muốn: ước được tự do, nhưng không phải tất cả chúng có thể được thỏa mãn. Trong trường hợp này sẽ không có sự thoản mãn đầy đủ sự mong muốn của con người, nhưng thường xuyên chuẩn bị chấp nhận sự thay đổi tồi tệ hơn hoặc vượt qua được sự thỏa hiệp. Do đó sự ưa thích trong thế giới thực yêu cầu một sự thay đổi mô hình từ yêu cầu phải chính xác và phù hợp nhất, ví dụ: sự ưa thích được xem như là sự ràng buộc không bắt buộc. Xa hơn nữa, sự ưa thích trong thế giới thực không thề bị xem như là sự không đáng mong đợi. Thay vì đó có nhiều tình huống giải quyết cho các sự mong đợi khác nhau là sẽ phức tạp, ví dụ: trong e-shopping, nơi mà khách hàng và người bán hàng có những sự sở hữu của riêng họ, có thể là sự ưa thích sẽ bị xung đột. Vai trò tỏa khắp của cá nhân hóa được xem xét đến trong ngôn ngữ truy vấn cơ sở dữ liệu của cả hai thế giới. Nhưng ngược lại để có sự phù hợp được nghiên cứu trong cơ sở dữ liệu và ngữ cảnh Web là một vấn đề lớn, vấn đề đi đầu trong nghiên cứu công nghệ (ví dụ: SQL, E/R-

modeling, XML), mô hình trong xu thế ưa thích lựa chọn trong thế giới thực là ẩn chứa bên trong.

Chúng ta khảo sát một trạng thái không thoả mãn của sự mưu mẹo bởi nhìn vào các bộ máy tìm kiếm cơ sở dựa trên SQL của e-shop, chúng ta sẽ thấy không thể có sự tương thích với những mong ước của người dùng như trong thế giới thực: Tất cả thường không có sự trả lời chính đáng trả lại từ các tìm kiếm cho phù hợp với mong muốn tốt nhất của người dùng. Phổ biến, sẽ có sự bắt gặp các câu trả lời trước khi nghe những câu giống như "không có khách sạn, xe, chuyến bay, ..v..v. có thể tìm thấy câu trả lời phù hợp hơn; xin vui lòng thử lại với các sự lựa chọn khác". Trong trường hợp nhận được các kết quả trả lời rỗng sẽ gây nên sự thất vọng cho người dùng, và sẽ làm thiệt hại nhiều cho người bán hàng. Lệnh cho người dùng rời bỏ một số điều kiện trong yêu cầu không mong đợi thường gây nên sự thất vọng: Một lượng quá tải với quá nhiều thông tin vào. Sẽ có một sự đến gần với nhiều sự thiếu hụt, đáng chú ý trong ngữ cảnh của hệ thống cơ sở dữ liệu đang hoạt động. Có một công nghệ của truy vấn linh động đã được nghiên cứu nhằm giải quyết vấn đề trả về kết quả rỗng, Đã trải qua nhiều thập kỷ sử dụng sự ưa thích nhằm giải quyết vấn đề lớn trong khoa học kinh tế và xã hội, điển hình là tính ra quyết định trong thao tác tìm kiếm, học máy và khai phá tri thức là các vấn đề tương lai nơi mà sự ưa thích sẽ được lựa chọn để giải quyết. Mỗi một sự tiếp cận và sự nghiên cứu đã từng khám phá ra một số thách thức đặt ra bởi sự ưa thích.

Tuy nhiên, một giải pháp tổng quát mà làm nền tảng dẫn đường cho một sự ổn thỏa và tích hợp hiệu quả của sự ưa thích với công nghệ cơ sở dữ liệu mà đã không từng được nêu ra. Tôi nghĩ là mô hình sự ưa thích có thể làm được cho hệ thống cơ sở dữ liệu nên đạt được như các mong muốn dưới đây:

(1) Ngữ nghĩa trực quan: Sự ưa thích phải trở thành sự quan tâm nhất trong xử lý mô hình. Điều này đòi hỏi một cách trực quan và giải thích rõ ràng của sự ưa thích. Mô hình sự ưa thích nên bao gồm biểu diễn phi số như là phương pháp phân hạng..

(2) Nền tảng toán học ngắn gọn: Yêu cầu này đưa ra là tất yếu, nhưng nền tảng toán học phải được cân đối với ngữ nghĩa trực quan.

(3) Xây dựng và mở rộng mô hình ưa thích: sự ưa thích đầy đủ nên được xây dựng quy nạp từ các vấn đề đơn giản sử dụng thông tin mở rộng của cấu trúc ưa thích.

(4) Các xung đột của các ưa thích phải không là nguyên nhân làm cho hệ thống bị lỗi: kết cấu động của ưa thích phức tạp phải được hỗ trợ ngay cả trong sự có mặt của sự xung đột. Mô hình ưa thích thực hiện nên có thể tồn tại cùng với sự xung đột, không ngăn chặn chúng hoặc gây ra lỗi nếu chúng xẩy ra.

(5) Xây dựng ngôn ngữ truy vấn ưa thích: Sự phù hợp trong thế giới thực làm cầu nối giữa những mong muốn và sự tin cậy. Sự thể hiện này là cần thiết cho một mô hình truy vấn mới khác phù hợp với mô hình của ngôn ngữ truy vấn cơ sở dữ liệu đã có trước đây.

1.2. Biểu diễn sự ưa thích trong hệ thống cơ sở dữ liệu

Sự ưa thích trong thế giới thực được thể hiện trong nhiều dạng khác nhau như là mọi người có thông tin về một đối tượng nào đó. Chúng ta làm một cuộc kiểm tra về những biểu lộ tự nhiên của con người khi ước muốn về một vấn đề gì đó. Hãy thử khám phá cuộc sống hàng ngày với sự phong phú của sự ưa thích đến từ sự cảm nhận hoặc ảnh hưởng khác. Trong thế giới thực này, nó trả lại một cách nhanh chóng những mong muốn thường xuyên xẩy ra, như là "tôi thích A hơn B". Loại ưa thích này là phổ biến và trực quan cho mọi người. Sự thật là, mỗi đứa trẻ học điều này từ khi chúng còn rất nhỏ. Nghĩ đến sự ưa thích có nghĩa là mong muốn "tốt hơn", điều này cũng có chút liên quan đến toán học: Toán học có thể ánh xạ chúng vào thành một thứ tự bộ phận chặt. Con người là thường xuyên đề cấp đến vấn đề sự ưa thích, thông thường với nó là không diễn tả trong phạm vi con số cụ thể. Nhưng cũng có một phần khác của cuộc sống thế giới thực với sự nguyên thủy có dính líu với tiết kiệm chi phí hoặc công nghệ đưa ra, nơi mà những con số là quan trọng. Một cách dễ hiểu hơn là xếp hạng số có thể được được xem như một phần

của ưa thích. Do đó mô hình ưa thích như là một ràng buộc không trọn vẹn có được hơn là lời hứa, điều này đã từng được chứng tỏ trong nhiều ngành khoa học khác nhau, đặc biệt là trong khoa học máy tính và các môn học .

Sự ưa thích là một trình bày rõ ràng cụ thể dựa trên một tập các thuộc tính định danh với một miền quan hệ của giá trị, theo cách nói ẩn dụ là "thuộc về ước muốn". Khi kết hợp sự ưa thích P_1 và P_2, chúng ta nói rằng P_1 và P_2 có thể chồng chéo lên những thuộc tính của chúng, cho phép nhiều sự ưa thích cùng tồn tại dựa trên cùng những thuộc tính như nhau. Sự phổ biến này là nên được quan tâm đến khi thiết kế hệ thống, ngay cả khi xẩy ra xung đột của sự ưa thích phải được cho phép trong thử nghiệm và không phải được xem như là lỗi.

Cho một tập không rỗng $A = (\{A_1, A_2, ..., A_k\})$ của các tên thuộc tính A_i có quan hệ với các miền của giá trị $dom(A_i)$. Xem xét theo thứ tự của các thành phần trong tích Đề các như là không quan trọng, chúng ta có:

$Dom(A) = dom(\{A_1, A_2, ... , A_k\}) := dom(A_1) \times dom(A_2) \times ... \times dom(A_k)$

Chú ý là định nghĩa này bao gồm điều kiện sau đây:

Nếu $B = \{A_1, A_2\}$ và $C = \{A_2, A_3\}$,

thì $dom(B \cup C) = dom(\{A_1, A_2\} \cup \{A_2, A_3\}) = dom(A_1) \times dom(A_2) \times dom(A_3)$.

Định nghĩa 1. Sự ưa thích P = (A, <P)

Cho một tập A của các tên thuộc tính, một sự ưa thích P là một thứ tự bộ phận chặt **P = (A, <P)** với $<P \subseteq dom(A) \times dom(A)$.

Do đó <P là phong phú và linh động. Điều quan trọng là giải thích sự mong đợi này: "x <P y" là được giải nghĩa như là "tôi thích y hơn x".

Xa hơn nữa: $range(<P) := \{x \in dom(A) \mid \exists y \in dom(A): (x, y) \in <P \text{ hoặc } (y, x) \in <P\}$.

Khi đó sự ưa thích mang lại một diện mạo quan trọng của thế giới thực và biểu diễn một cách trực quan tốt hơn.

Định nghĩa 2: Đồ thị better-than, những nét đặc trưng.

Trong những miền hữu hạn cho một ưa thích P có thể được vẽ như là một đồ thị không chu trình có hướng G, được gọi là đồ thị "better-than" của P . Dùng G thay cho P chúng ta định nghĩa một số khái niệm sau đây giữa giá trị x, y trong G:

- x <P y, nếu y có tổ tiên của x trong G.
- Các giá trị trong G mà không là tổ tiên là phần tử lớn nhất của P (max(P)), trở thành cấp độ 1.
- X là cùng cấp j, nếu đường đi dài nhất từ x tới giá trị lớn nhất có j-1 đỉnh.
- Nếu không có đường đi có hướng giữa x và y trong G, thì x và y là không phân hạng được.

Định nghĩa 3 Các trường hợp đặc biệt của sự ưa thích.

a) P = (A, <P) là một sự ưa thích móc xích, nếu cho tất cả x, y $\in$ dom(A), x $\neq$ y: x <P y $\vee$ y <P x

b) $S^{\leftrightarrow}$ = (S, □) được gọi là ưa thích không móc xích, cho bất kỳ tập giá trị S nào.

c) Ưa thích đối ngẫu P^{δ} = (A, $<P^{\delta}$) nghịch đảo thứ tự trong P: x $<P^{\partial}$ y nếu và chỉ nếu y <P x

d) Cho P = (A, <P), mọi S □ dom(A) bao gồm một tập con ưa thích $P^{\square}$ = (S, $<P^{\square}$) , nếu cho bất kỳ x, y $\in$ S: x $<P^{\square}$ y nếu và chỉ nếu x <P y

Do đó tất cả các giá trị x của một ưa thích móc xích P (cũng được gọi là thứ tự tổng thể) là được xếp hạng cho tất cả các giá trị y khác. Bất kỳ tập S, bao gồm dom(A), có thể được bao gồm trong một không móc xích. Tập con đặc biệt ưa thích, được gọi là ưa thích cơ sở dữ liệu, sẽ trở thành quan trọng sau này.

1.3. Kỹ nghệ ưa thích

Những ước muốn là phong phú và xẩy ra hàng ngày trong cuộc sống của chúng ta. Do đó có một yêu cầu cao về sự hiểu biết và nền tảng cơ sở cho việc hỗ trợ sự góp nhặt các ưa thích đơn vào trong một tập hoàn chỉnh. Chúng ta biểu diễn một biến đổi quy nạp theo hướng gần đúng với cấu trúc ưa thích phức tạp. Mô hình này

sẽ trở thành chìa khóa then chốt của ngữ nghĩa kỹ nghệ ưa thích và cho đại số ưa thích.

1.3.1 Cấu trúc quy nạp của ưa thích.

Kết quả là với mục đích cung cấp một cách trực quan và phương hướng thuận tiện cho cấu trúc quy nạp của ưa thích P = (A, <P). Kết thúc sẽ đưa ra P phù hợp với miền giá trị A và ràng buộc <P. Chúng ta phân vân giữa ưa thích cơ bản (điều kiện ưa thích nguyên tử) và ưa thích lai ghép. Khi đó mỗi điều kiện ưa thích biểu diễn một ràng buộc, chúng ta gọi nó là ưa thích P.

Định nghĩa 4: Thuật ngữ ưa thích.

Cho một giới hạn ưa thích P1 và P2, P là một giới hạn ưa thích nếu P là thỏa mãn một trong các điều kiện sau:

(1) Bất kỳ ưa thích cơ sở: P := basepref_i.

(2) Bất kỳ ưa thích tập con: P := $P1^{\square}$

(3) Bất kỳ ưa thích đối ngẫu: P := $P1^{\delta}$.

(4) Bất kỳ ưa thích phức tạp P nhận được từ sử dụng một trong các cấu trúc ưa thích sau:

Cấu trúc ưa thích tích lũy:

- Tích lũy Pareto: P := P1 $\square$ P2
- Tích lũy ưu tiên: P := P1 & P2
- Tích lũy số: P := rank_F(P1, P2)

Cấu trúc ưa thích kết tập:

- Kết tập giao nhau: P := P1 ♦ P2
- Kết tập liên kết rời rạc: P := P1 + P2
- Kết tập tổng tuyến tính: P := P1 $\oplus$ P2

Cả hai tập ưa thích cơ sở và tập cấu trúc ưa thích phức tạp có thể được mở rộng bất kỳ lúc nào cho miền ứng dụng khi mà có yêu cầu.

1.3.2 Các cấu trúc ưa thích cơ sở.

Một điều quan trọng trong kỹ nghệ ưa thích là chúng ta có thể cung cấp cấu trúc ưa thích cơ sở, mà được gọi là các ưa thích mẫu. Kinh nghiệm thực tế cho thấy các điều khoản sau là được đánh giá cao cho cấu trúc bộ máy tìm kiếm hướng người dùng.

Chính thức, một cấu trúc ưa thích cơ sở có một hoặc nhiều đối số, tính chất đầu tiên của thuộc tính tên A và các ràng buộc khác <P, tham chiếu tới A. Chúng tôi sẽ cung cấp cả hai dạng và một định nghĩa quy nạp cùng với một ví dụ cụ thể used_car.

1.3.2.1 Ưa thích cơ sở phi số.

a) Ưa thích POS: POS(A, POS-set)

P là một ưa thích POS, nếu: x <P y khi và chỉ khi x□ POS-set ∧ y ∈ POS-set

Giá trị mong muốn nên ở trong tập giới hạn của ưa thích.POS-set □ dom(A). Nếu điều này không thể làm được, tốt hơn không lấy bất cứ giá trị nào từ dom(A) là chấp nhận.

Ứng dụng Used_car viết như sau:

POS(transmission, {automatic})

b) Ưa thích NEG: NEG(A, NEG-set)

P là một ưa thích NEG, nếu:

x <P y khi chỉ khi y □ NEG-set ∧ x ∈ NEG-set

Giá trị mong đợi nên là một từ tập giới hạn của giá trị có. Các trường hợp khác nó không nên là bất cứ giá trị nào từ tập hữu hạn của các giá trị không mong muốn. Nếu điều này không khả thi, tốt hơn không nên lấy giá trị nào cả.

Ứng dụng cho Used_car như sau:

POS/NEG(color, {yellow};{gray})

d) Ưa thích POS/POS : POS/POS(A, POS1-set; POS2-set)

P được gọi là ưa thích POS/POS, nếu:

x <P y khi và chỉ khi (x ∈ POS2-set ∧ y ∈ POS1-set) ∨

(x □ POS1-set ∧ x □ POS2-set ∧ y ∈ POS2-set) ∨

(x □ POS1-set ∧ x □ POS2-set ∧ y ∈ POS1-set)

Giá trị mong đợi nên bao gồm một tập hữu hạn POS1-set. Các trường hợp khác nên là từ các tập hữu hạn rời rạc của tập POS2. Nếu không phải giá trị mong đợi, tốt hơn nên không chọn lựa giá trị nào cả.

Áp dụng cho trường hợp Used_car như sau:

POS/POS(category,{cabrio};{roadster})

e) Ưa thích EXPLICIT: EXP(A, E-graph)

Cho đồ thị E-graph = {(val_1, val_2), … } biểu diễn một đồ thị "better-than" không vòng hữu hạn, V là một tập các đỉnh xuất hiện trong đồ thị E. Một ràng buộc E = (V, <E) là bao gồm các điều kiên sau

- (val_i, val_j) ∈ E-graph đưa đến val_i <E val_j
- val_i <E val_j ∧ val_j <E val_k đưa đến val_i <E val_k

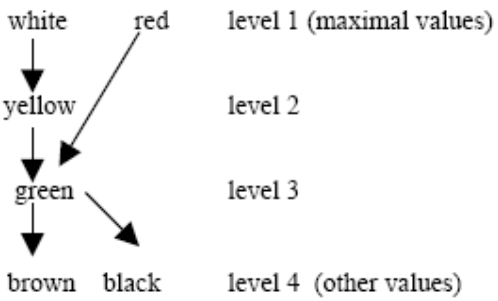
P là một ưa thích EXPLICIT, nếu:

x <P y khi và chỉ khi x <E y ∨ (x □ range(<E) ∧ y ∈ range(<E))

Áp dụng cho trường hợp Used_car như sau:

EXP(color, {(green, yellow), (green, red), (yellow, white)})

Cho dom(Color) = {white, red, yellow, green, brown, black}, đồ thị 'better-than' như sau:



1.3.2.2 Ưa thích cơ sở kiểu số.

Bây giờ chúng ta nghiên cứu P = (A, <P), với dom(A) là một loại dữ liệu ngày tháng, ví dụ: số nhị phân hoặc ngày tháng, hỗ trợ một phép so sánh'<' và một toán

tử trừ '-'. Thay vì làm giảm mức của chức năng trên, chúng tôi tiếp tục sử dụng phép toán '<' và '-'.

a) Ưa thích AROUND: AROUND(A, z)

Cho z ∈ dom(A), và cho tất cả v ∈ dom(A) chúng ta có:

distance(v, z) := abs(v − z)

P được gọi là ưa thích AROUND, nếu:

x <P y khi chỉ khi distance(x, z) > distance(y, z)

Giá trị mong đợi nên là z. Nếu đây là giá trị thích hợp, những giá trị với với khoảng cách ngắn nhất từ x là có thể chấp nhận.

Áp dụng cho trường hợp Used_car như sau: AROUND(price, 40000)

Chú ý là nếu distance(x, z) = distance(y, z) và x ≠ y, sau đó x và y là không mong đợi.

b) Ưa thích BETWEEN: BETWEEN(A, [low, up])

Cho [low, up] ∈ dom(A) × dom(A), chúng ta xác định cho tất cả v ∈ dom(A):

distance(v, [low, up]) :=

nếu v ∈ [low, up] kết quả 0 hoặc

nếu v < low kết quả low − v hoặc v − up

P được gọi là BETWEEN ưa thích, nếu: x <P y khi và chỉ khi

distance(x, [low, up]) > distance(y, [low, up])

Giá trị mong đợi nên là giữa đường biên của một khoảng thời gian. Nếu giá trị này là khả thi, các giá trị có khoảng cách ngắn nhất từ đường biên bên trong với giá trị chấp nhận được.

Áp dụng cho trường hợp Used_car:

BETWEEN(mileage, [20000, 30000])

c) Ưa thích LOWEST, HIGHEST: LOWEST(A), HIGHEST(A)

P được gọi là ưa thích LOWEST , nếu: x <P y nếu và chỉ nếu x > y

P được gọi là ưa thích HIGHEST, nếu: x <P y nếu và chỉ nếu x < y

Giá trị mong đợi nên là thấp (cao) là có thể được.

Áp dụng cho ứng dụng Used_car: HIGHEST(power)

Chú ý: Ưa thích LOWEST và HIGHEST là các chuỗi.

d) Ưa thích SCORE: SCORE(A, f)

Cho một hàm f: dom(A) → ℝ. Xem '<' là tương tự như 'less-than' trong ℝ.

P được gọi là ưa thích SCORE nếu cho x, y ∈ dom(A):

x <P y khi và chỉ khi f(x) < f(y)

Nhìn chung biểu diễn không trực quan.

1.3.3 Cấu trúc ưa thích phức tạp.

Chúng ta tiến tới nghiên cứu một mô hình ưa thích phức tạp hơn.

1.3.3.1 Cấu trúc ưa thích tích lũy.

Cấu trúc ưa thích tích lũy ('□', '&', '$rank_F$') kết hợp các ưa thích đến từ một hoặc nhiều phần khác nhau. Nguyên lý Pareto tối ưu đã từng được nghiên cứu cho nhiều bài toán ra quyết định trong khoa học xã hội và kinh tế. Ở đây chúng ta định nghĩa nó cho n=2 ưa thích (tổng quát hóa cho n > 2 là hiển nhiên).

Định nghĩa 5: Ưa thích Pareto: P1□P2

P1 và P2 là được xem như là ưa thích quan trọng ngang nhau. Cho x = (x_1, x_2) là tốt hơn y = (y_1, y_2), Điều này là không vi phạm x là xấu hơn y trong bất kỳ x_i

Cho P1 = (A1, <P1) và P2 = (A2, <P2), cho x, y ∈ dom(A1) × dom(A2) chúng ta định nghĩa:

x <P1□P2 y nếu và chỉ nếu (x_1 <P1 y_1 ∧ (x_2 <P2 y_2 ∨ $x_2 = y_2$)) ∨

(x_2 <P2 y_2 ∧ (x_1 <P1 y_1 ∨ $x_1 = y_1$))

P = (A1 ∪ A2, <P1□P2) được gọi là ưa thích Pareto. Giá trị lớn nhất của P là tập Pareto tối ưu.

Ví dụ 1 Ưa thích Pareto (disjoint attrib. names)

Cho dom(A1) = dom(A2) = dom(A3) = integer và

P1 := AROUND(A1, 0),

P2 := LOWEST(A2), P3 := HIGHEST(A3)

P4 = ({A1, A2, A3}, <P4) := (P1 □ P2) □ P3

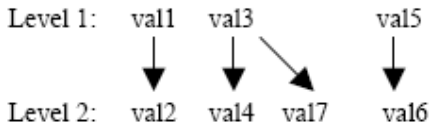
Hãy xem xét tập con ưa thích của P4 cho tập:

R(A1, A2, A3) = {val1: (−5, 3, 4), val2: (−5, 4, 4),

val3: (5, 1, 8), val4: (5, 6, 6), val5: (−6, 0, 6),

val6: (−6, 0, 4), val7: (6, 2, 7)}

Đồ thị "better than" của P4 cho tập con R có thể đạt được bởi thực hiện nhiều lần kiểm tra "better-than".



Do đó tập Pareto tối ưu là {val1, val3, val5}. Chú ý cho mỗi P1, P2 và P3 có ít nhất một giá trị lớn nhất xuất hiện trong tập Pareto tối ưu: 5 và -5 cho P1, 0 cho P2 và 8 cho P3.

Định nghĩa 6: Ưa thích ưu tiên: P1&P2

P1 được xem như là quan trọng hơn P2; P2 là được chú ý duy nhất khi P1 không được chú ý:

Cho P1 = (A1, <P1) và P2 = (A2, <P2), cho x, y $\in$ dom(A1) × dom(A2) chúng ta định nghĩa:

x <P1&P2 y nếu và chỉ nếu x_1 <P1 $y_1 \vee (x_1 = y_1 \wedge x_2$ <P2 $y_2)$

P = (A1 $\cup$ A2, <P1&P2) là một ưa thích ưu tiên

Định nghĩa 7: Ưa thích kiểu số: $rank_F$(P1, P2)

Cho P1 = SCORE(A1, f1), P2 = SCORE(A2, f2) và một hàm kết hợp F: $\mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$, cho x, y $\in$ dom(A1) × dom(A2) chúng ta định nghĩa: x <$rank_F$(P1, P2) y khi và chỉ khi nếu :

$F(f1(x_1), f2(x_2)) < F(f1(y_1), f2(y_2))$

P = (A1 $\cup$ A2, <$rank_F$(P1, P2)) là một ưa thích số.

Chú ý là $rank_p$ không là một cấu trúc ưa thích trực giao giống như □ hoặc &. Nó có thể duy nhất được sử dụng cho ưa thích SCORE. Nhưng qua versa, ưa tích số có thể được dùng như là dữ liệu đầu vào cho tất cả các cấu trúc ưa thích khác.

1.3.3.2 Cấu trúc ưa thích kết tập.

Cấu trúc ưa thích kết hợp (♦, +, ⊕)

Tiếp tục một sự khác biệt, mục đích công nghệ. Điểm giao '♦' và không giao '+' lắp ghép thành một ưa thích P từ các phần $P_1, P_2, ..., P_n$, tất cả hoạt động trên cùng một tập các thuộc tính. Qua versa, chúng ta sẽ nhìn thấy sau trên ưa thích phức tạp có thể được giải mã vào trong '♦' và '+'.

Chúng ta gọi P1 = (A1, <P1) và P2 = (A2, <P2) ưa thích không giao nhau, nếu range(<P1) ∩ range(<P2) = □.

Định nghĩa 8: Ưa thích giao và ưa thích hợp rời

Giả sử P1 =(A,<P1) và P2 = (A,P2)

a) P = (A, < P1♦P2) là một ưa thích giao nhau, nếu:

x <P1♦P2 y nếu và chỉ nếu x <P1 y ∧ x <P2 y

b) Cho ưa thích rời nhau P1 và P2, P = (A, <P1+P2) được gọi là ưa thích hợp nhất không kết hợp, nếu:

x <P1+P2 y nếu và chỉ nếu x <P1 y ∨ x <P2 y

Định nghĩa 9: Ưa thích tổng tuyết tính

Cho P1 = (A1, <P1), P2 = (A2, <P2) cho các thuộc tính đơn A1 ≠ A2 và dom(A1) ∩ dom(A2) = □. Sau đó P1 và P2 là các ưa thích tách rời. Cho một thuộc tính tên mới A mà dom(A) := dom(A1) ∪ dom(A2).

Sau đó P = (A, <P1 ⊕ P2) là một ưa thích tổng tuyến tính, nếu:

x <P1 ⊕ P2 y nếu và chỉ nếu x <P1 y ∨ x <P2 y ∨ (x ∈ dom(A2) ∧ y ∈ dom(A1))

Tổng tuyến tính ‘$\oplus$’ có thể được xem như là một thiết kế phù hợp và phương thức chứng minh cho cấu trúc ưa thích cơ bản. Với khái niệm thích hợp của ‘other-values’ chúng ta có trạng thái sau:

Một POS ưa thích là một tổng tuyến tính của nhiều chuỗi trên tập POS-set với đa chuỗi trên các giá trị khác:

$$\text{POS} = \text{POS-set}^{\leftrightarrow} \oplus \text{other-values}^{\leftrightarrow}$$

Tương tự chúng ta theo dõi:

$$\text{POS/NEG} = (\text{POS-set}^{\leftrightarrow} \oplus \text{other-values}^{\leftrightarrow}) \oplus \text{NEG-set}^{\leftrightarrow}$$

$$\text{POS/POS} = (\text{POS1-set}^{\leftrightarrow} \oplus \text{POS2-set}^{\leftrightarrow}) \oplus \text{other-values}^{\leftrightarrow}$$

$$\text{EXPLICIT} = \text{E} \oplus \text{other-values}^{\leftrightarrow}$$

Tại điểm này, chúng ta có thể tính tổng tất cả các kết quả được phát biểu như sau, tham chiếu trở lại định nghĩa 4.

Mệnh đề 1

Mỗi số hạng ưa thích định nghĩa một ưa thích thứ tự bộ phận nghiêm ngặt.

Định lý này cho chúng ta thấy được sự kết hợp mềm dẻo của các số hạng ưa thích tùy thuộc vào yêu cầu đưa ra trong các ứng dụng cụ thể.

1.3.4 Phân cấp ưa thích.

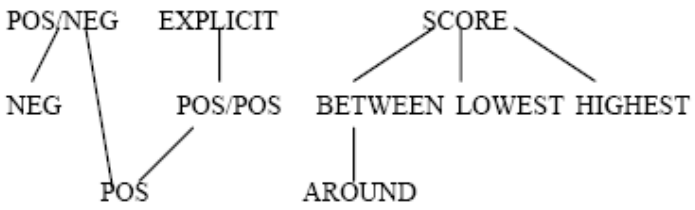
Cấu trúc ưa thích C1 và C2 có thể được được sắp xếp trong sơ đồ phân cấp. Chúng ta gọi C1 một cấu trúc phụ ưa thích của C2 (C1 < C2), nếu định nghĩa của C1 có thể thu được từ tập xác định của C2 bởi một số ràng buộc đặc biệt.

Phân cấp không rỗng. Cấu trúc ưa thích cơ bản:

- POS/POS < EXPLICIT, nếu E-graph = $(\text{POS1-set})^{\leftrightarrow} \oplus (\text{POS2-set})^{\leftrightarrow}$
- POS < POS/POS, nếu POS2-set = $\square$
- POS < POS/NEG, nếu NEG-set = $\square$
- NEG < POS/NEG, nếu POS-set = $\square$

Phân cấp của cấu trúc ưa thích cơ sở số: (‘N’ nghĩa là ‘numeric’)

- BETWEEN < SCORE, nếu A là ‘N’ và f(x) = − distance(x, [low, up])
- AROUND < BETWEEN, nếu low = up
- HIGHEST < SCORE, if A là ‘N’ và f(x) = x
- LOWEST < SCORE, nếu A là ‘N’ và f(x) = −x



Phân cấp của cấu trúc ưa thích phức tạp:

- ‘♦’ < ‘⊗’
- Không phải mọi cấu trúc ưa thích có thể được trình bày như một cấu trúc con của ‘$rank_F$’. [1]

Khi đó chúng ta có các ràng buộc cụ thể, các phân cấp cấu trúc phụ là được phân loại. Bên cạnh đó, có một thuận lợi cho kỹ nghệ phần mềm hướng đối tượng là cố gắng tiết kiệm chi phí: Ngữ nghĩa của sự ràng buộc phải được thẩm định duy nhất cho cấu trúc ưa thích mức độ cao nhất. Xa hơn nữa chúng ta giả sử nguồn gốc của cấu trúc các tình huống phụ, ví dụ: thay vì cấu trúc được yêu cầu cũng là một cấu trúc phụ có thể được sử dụng. Ví dụ, rankF(P1, P2) yêu cầu là P1 và P2 là ưa thích SCORE. Thay vào đó, chúng ta cũng sử dụng ưa thích P1 và P2 được xây dựng bởi AROUND và HIGHEST, theo thứ tự định sẵn.

1.4 Đại số ưa thích.

Ràng buộc chặt là được trình bày rõ ràng bởi các công thức logic thứ tự ưu tiên, chúng có thể được thực hiện bằng đại số logic. Với ưa thích khác, sẽ được biểu diễn bằng điều kiện ưa thích, là được dùng để diễn tả ràng buộc đơn giản. Do đó mong đợi phát triển một đại số ưa thích mà có luật biến đổi nằm trong điều kiện ưa thích.

Các nghiên cứu tiếp theo sẽ nêu các vấn đề ngữ nghĩa của ràng buộc ưa thích. Trước tiên chúng ta sẽ cần quan tâm đến tính tương đương của các số hạng ưa thích.

Định nghĩa 10: Sự tương đương của các thuật ngữ ưa thích.

P1 = (A, <P1) và P2 = (A, <P2) là tương đương **(P1 ≡ P2)**, nếu cho tất cả x và y $\in$ dom(A): x <P1 y nếu và chỉ nếu x <P2 y

Nếu P1 ≡ P2, sau đó các điều kiện ưa thích P1 và P2 có thể là cú pháp khác nhau, nhưng các ưa thích được biểu diễn bởi P1 và P2, .. là giống nhau.

1.4.1 Tập các luật đại số.

Mệnh đề 2 Các luật kết hợp và giao hoán.

a) P1 □ P2 ≡ P2 □ P1

(P1 □ P2) □ P3 ≡ P1 □ (P2 □ P3)

b) (P1 & P2) & P3 ≡ P1 & (P2 & P3)

c) P1♦ P2 ≡ P2 ♦ P1

(P1♦ P2)♦ P3 ≡ P1♦ (P2 ♦ P3)

d) P1 + P2 ≡ P2 + P1

(P1 + P2) + P3 ≡ P1 + (P2 + P3)

e) (P1 $\oplus$ P2) $\oplus$ P3 ≡ P1 $\oplus$ (P2 $\oplus$ P3)

Mệnh đề 3 Các luật thay thế cho các số hạng ưa thích

a) $(S^{\leftrightarrow})^{\partial} \equiv S^{\leftrightarrow}$ cho bất kỳ tập S , $(P^{\partial})^{\partial} \equiv P$

b) $(P1 \oplus P2)^{\partial} \equiv P2^{\partial} \oplus P1^{\partial}$

c) $HIGHEST \equiv LOWEST^{\partial}$

d) $POS^{\partial} \equiv NEG$,

$NEG^{\partial} \equiv POS$ nếu POS-set = NEG-set

e) P ♦ P ≡ P

f) $P \blacklozenge P^{\delta} \equiv P \blacklozenge A^{\leftrightarrow} \equiv A^{\leftrightarrow}$ nếu P = (A, <P)

g) Nếu P1 và P2 là các chuỗi, thì

P1 & P2 và P2 & P1 là các chuỗi.

h) $P \& P \equiv P \& P^{\partial} \equiv P$

i) $P \& A^{\leftrightarrow} \equiv P$ nếu $P = (A, <P)$

j) $A^{\leftrightarrow} \& P \equiv A^{\leftrightarrow}$ nếu $P = (A, <P)$

k) $P \square P \equiv P$, $A^{\leftrightarrow} \square P \equiv A^{\leftrightarrow} \& P$

l) $P \square A^{\leftrightarrow} \equiv P \square P^{\partial} \equiv A^{\leftrightarrow}$ nếu $P = (A, <P)$

Các luật này là phù hợp với ngữ nghĩa trực quan mong đợi. Ví dụ., hãy xem $P \square P^{\partial} \equiv A^{\leftrightarrow}$: Khi P và P^{∂} là quan trọng như nhau , trong trường hợp xung đột cho các giá trị của x và y không có giá trị nào vượt trội, thay vì x và y còn lại không được xếp hạng. Khi đó P và P^{∂} là bị xung đột, miền đầy đủ trở thành không được xếp hạng, chuỗi ngược lại $A^{\leftrightarrow}$.

1.4.2 Phân tích ưa thích ưu tiên và ưa thích Pareto

Giải nghĩa thực tế của sự tích lũy ưu tiên.

Mệnh đề 4 Triển khai cho P1&P2

(a) $P1\&P2 \equiv P1$ nếu $P1 = (A, <P1)$ và $P2 = (A, <P2)$

(b) $P1\&P2 \equiv P1 + (A1^{\leftrightarrow}\&P2)$ nếu $A1 \cap A2 = \square$

Mệnh đề 5 .Định lý "Non-discrimination"

$P1 \square P2 \equiv (P1 \& P2) \blacklozenge (P2 \& P1)$

P1 và P2 là được xem như quan trong trọng như nhau bởi '□', khi đó cả hai đã cho quan trọng chủ yếu bởi '&'. Bất cứ xung đột nảy sinh nào được giải quyết trong cách biến đổi bởi sự giao nhau '♦'. Kết quả là chúng ta có trạng thái:

$P1 \square P2 \equiv P1 \blacklozenge P2$ nếu $P1 = (A, <P1)$ và $P2 = (A, <P2)$

Do đó '♦' là một cấu trúc phụ ưa thích của '□'.

1.5 Tổng kết chương

Chúng ta đã nêu ra một mô hình ưa thích mà hoàn toàn thích hợp cho các hệ thống cơ sở dữ liệu. Nhiều yêu cầu của thế giới thực là gặp phải trong mô hình ưa thích như là một thứ tự bộ phận chặt: Nó hợp nhất của phân hạng phi số và kiểu số. Nó có một ngữ nghĩa trực quan mà được hiểu bởi mọi người và nó có thể được ánh xạ trực tiếp vào cơ sở toán. Mô hình ưa thích này mô tả nhiều nét nổi bật của các cấu trúc ưa thích mà thường xuyên được sử dụng. Bao gồm các cấu trúc ưa thích phức tạp và các luật đại số ưa thích, chúng sẽ là nền tảng cho việc xử lý và tối truy vấn ở chương 2.

Chương II. XỬ LÝ VÀ TỐI ƯU TRUY VẤN ƯA THÍCH QUAN HỆ

2.1. Giới thiệu

Sự ưa thích là một phần không thể thiếu trong cuộc sống hàng ngày và trong thương mại. Do đó, sự ưa thích phải là một yếu tố then chốt trong thiết kế những ứng dụng hướng người dùng và hệ thống thông tin dựa trên nền Internet. Sự ưa thích của con người là thường diễn tả một điều ước muốn gì đó. Ước muốn là không bị ràng buộc, nhưng không có sự giới hạn mà chúng có thể thỏa mãn mọi lúc. Trong trường hợp không thoả mãn ước muốn của con người, khi đó sẽ phải chấp nhận một kết quả tương đương, chấp nhận được. Do đó, sự ưa thích trong thế giới thực yêu cầu có một sự biến hóa phù hợp với thực tế, có nghĩa là tìm khả năng phù hợp nhất giữa ước muốn và thực tế xẩy ra. Hay nói cách khác, sự ưa thích có tính ràng buộc mềm dẻo. Như ta đã biết, các ngôn ngữ truy vấn thông thường như SQL không đưa ra cách diễn tả ưu thích. Đây là một thiêu sót lớn đối với hệ cơ sở dữ liệu hỗ trợ trong nhiều ứng dụng quan trọng, điển hình là trong bộ máy tìm kiếm cho e-commerce hoặc m-commerce. Đây là những vấn đề then chốt mà truy vấn ưa thích sẽ hỗ trợ cho SQL hoặc XML, nó sẽ làm cho bộ máy tìm kiếm ngày càng trở nên tốt hơn, trả lại kết qủa cho người sử dụng thân thiện hơn. Ưa thích là đề tài có vai trò lớn trong nhiều viện nghiên cứu trong nhiều thập kỷ, ngay cả trong những ngành khoa học kinh tế và xã hội.

Trong chương này, chúng ta tập trung vào vấn đề then chốt của truy vấn ưa thích. Điển hình, chúng ta nghiên cứu những vấn đề thách thức của truy vấn ưa thích tối ưu trong cơ sở dữ liệu quan hệ.

2.2 Đánh giá cho các truy vấn ưa thích.

Trong cơ sở dữ liệu SQL giường như có sự so sánh đơn giản. Các truy vấn đối với quan hệ R là được phát biểu như là ràng buộc cứng, dẫn đến cách ứng xử hoặc là tất cả hoặc là không: Nếu giá trị mong đợi là trong R, bạn có thể có chính xác những cái gì bạn muốn, trong các trường hợp khác bạn sẽ không nhận được giá trị

nào. Sự thiếu hụt sau cùng là cho kết quả rỗng. Mô hình truy vấn phù hợp nhất có thể trở thành gây khó chịu thực sự trong nhiều ứng dụng thương mại điện tử. Các trường hợp khác, nếu có lo ngại về kết quả rỗng, câu truy vấn là được xây dựng bởi cấu truy vấn phụ. Sau đó sẽ thường có nhiều kết quả tương đương. Điều này là gây ảnh hưởng xấu.

Trong thế giới thực, những điều ước muốn diễn tả một ưa thích, hoặc đơn giản là tất cả hoặc không kiểu nào hoặc sẽ chọn được giá trị mong đợi. Thay vì đó, ngữ nghĩa câu trả lời là phải có, dù ưa thích có được thỏa mãn hay không và tùy thuộc vào trạng thái hiện tại của thế giới thực. Do đó chúng ta phải có sự phù hợp giữa điều mong ước và thực tế. Để thực hiện điều này, chúng ta lựa chọn mô hình được gọi là BMO(Best Matches Only).

2.2.1 Truy vấn ưu thích và mô hình truy vấn BMO.

Ưa thích là được định nghĩa trong phạm vi của các giá trị từ dom(A), một biểu diễn phạm vi của các điều ước. Trong các ứng dụng cơ sở dữ liệu, chúng ta giả sử là thế giới thực là được ánh xạ vào thành các phần tử mà được gọi là các tập cơ sở dữ liệu. Một tập cơ sở dữ liệu R có thể được xem là khung hoặc một quan hệ cơ sở trong SQL hoặc trường hợp DTD trong XML. Trong hệ cơ sở dữ liệu chấp nhận được hoặc trạng thái của thế giới thực. Do đó chúng là tập con của miền giá trị, vì thế chúng là những ưa thích tập hợp con.

Xem xét đến một tập cơ sở dữ liệu $R(B_1, B_2, \ldots, B_m)$. Cho $A = \{A_1, A_2, ..., A_k\}$, với mỗi A_j xác định một thuộc tính B_i từ R, với $R[A] := R[A_1, A_2, ..., A_k]$ xác định một phép chiếu π của R trên những thuộc tính k này.

Định nghĩa 11 Ưa thích cơ sở dữ liệu P^R

Giả sử $P = (A, <P)$, với $A = \{A_1, A_2, \ldots, A_k\}$.

a) Mỗi $R[A] \subseteq dom(A)$ xác định một tập con ưa thích, được gọi là một ưa thích cơ sở dữ liệu và xác định bởi:

$$P^R = (R[A], <P)$$

b) Phần tử $t \in R$ là lý tưởng trong cơ sở dữ liệu tập R, nếu:

$$t[A] \in \max(P) \wedge t[A] \in R$$

So sánh max(P), những đối tượng mong muốn của P, với tập $\max(P^R)$, những đối tượng tốt nhất trong thế giới thực, có thể không bao trùm nhau. Nhưng nếu xẩy ra, chúng tôi có một sự phù hợp lý tưởng giữa những mong ước và hiện thực. Nếu t là một phần tử phù hợp cho P trong R, sau đó $t[A] \in \max(P^R)$. Nhưng sự đảo ngược không được duy trì trong tổng quát. Các truy vấn ưa thích thực hiện một sự hòa hợp giữa trạng thái ưa thích (mong ước) và cơ sở dữ liệu ưa thích(sự tin cậy).

Định nghĩa 12 Ngữ nghĩa của truy vấn ưa thích σ[P](R), mô hình truy vấn BMO.

Giả sử P=(A, <P) và cơ sở dữ liệu ưa thích P_R. Chúng ta định nghĩa cho một truy vấn ưa thích σ[P](R) được nêu ra như sau:

$$\sigma[P](R) = \{t \in R \mid t[A] \in \max(P^R)\}$$

Một truy vấn ưa thích σ[P](R) đánh giá P dựa vào tập cơ sở dữ liệu R bởi lấy lại tất cả các giá trị lớn nhất từ P^R. Chú ý là không tất cả chúng là cần sự phù hợp hoàn toàn của P. Do đó nguồn gốc của quan hệ truy vấn là ẩn trong định nghĩa bên trên. Xa hơn nữa, bất cứ giá trị không lớn nhất nào của P^R là được ngăn chặn từ kết quả truy vấn, vì thế có thể được xem xét như là vứt bỏ trên fly. Trong trường hợp này tất cả các giá trị phù hợp nhất và chỉ những giá trị đó là được lấy lại bởi truy vấn ưa thích. Do đó chúng ta đưa ra thuật ngữ mô hình truy vấn BMO ("Best Matches Only")

Ví dụ 8 Mô hình truy vấn BMO

Chúng ta xem lại ví dụ 1 với ưa thích P EXPLICIT và áp dụng truy vấn σ[P](R) cho R(Color) = {yellow, red, green, black}. BMO kết quả là : σ[P](R) = {yellow, red}. Chú ý là red là giá trị phù hợp nhất. Nhận xét tiếp theo là không phức tạp, nhưng là rất quan trọng cho phát biểu.

Mệnh đề 6. Nếu P1 ≡ P2 , thì với tất cả R: σ[P1](R) = : σ[P2](R)

Bên cạnh truy vấn ưa thích của dạng : σ[P](R) biến đổi sẽ cần thường xuyên, với khởi đầu từ một tác động lẫn nhau giữa grouping và anti-chains. Cho mục đính này, chúng ta xem xét truy vấn ưa thích σ[$A^{\leftrightarrow}$&P](R), với P = (B,<P):

Chúng ta có x <$A^{\leftrightarrow}$&P y nếu x1 <$A^{\leftrightarrow}$ y1 $\vee$ (x1=y1 $\wedge$ x2 <P y2)

nếu false $\veebar$ (x1 = y1 $\barwedge$ x2 <P y2) nếu x1 = y1 $\barwedge$ x2 <P y2

Khi đó t $\in$ σ[$A^{\leftrightarrow}$&P](R) nếu t[A, B] $\Subset$ max(($A^{\leftrightarrow}$&P)^R)

Nếu $\forall$ v[A, B] $\Subset$ R[A, B]: $\neg$ (t[A, B] <$A^{\leftrightarrow}$&P v[A, B])

Nếu $\forall$ v[A, B] $\Subset$ R[A, B]: $\neg$ (t[A]=v[A] $\wedge$ t[B] = v[B])

Các điều kiện mô tả một nhóm của R bởi cân bằng A giá trị, định giá cho mỗi nhóm G_i của các phần tử truy vấn ưa thích σ[$A^{\leftrightarrow}$&P](Gi). Tử đó ta đưa ra định nghĩa tiếp theo.

Định nghĩa 13 Ngữ nghĩa của σ[P groupby A](R)

Giả sử P = (B, <P) và cơ sở dữ liệu ưa thích P^R. Chúng ta có định nghĩa truy vấn ưa thích với grouping σ[P groupby A](R) như sau: σ[P groupby A](R) := σ[A↔&P](R)

So sánh với các truy vấn lựa chọn bắt buộc, truy vấn lựa chọn ưa thích trệch hướng từ logic phía sau các lựa chọn bắt buộc:

Truy vấn ưa thích luôn luôn là không đơn điệu.

VÍ dụ 9 Tính không đơn điệu của các kết quả truy vấn ưa thích.

Hãy xem P = HIGHEST(Fuel_Economy) □ HIGHEST(Insurance_Rating). Chúng ta lần lượt đánh giá σ[P](Cars) cho Cars(Fuel_Economy, Insurance_Rating, Nickname) như là:

Cars = {(100, 3, frog), (50, 3, cat)}: σ[P](R) = {(100, 3, frog)}

Cars = {(100, 3, frog), (50, 3, cat) (50, 10, shark)}:

σ[P](R) = {(100, 3, frog), (50, 10, shark)}

Cars = {(100, 3, frog), (50, 3, cat) (50, 10, shark), (100, 10, turtle)}:

σ[P](R) = {(100, 10, turtle)}

Không đơn điệu là rõ ràng: mặc dầu đã thêm vào nhiều giá trị cho Cars, các kết quả của truy vấn ưa thích không phô bày sự cư xử như nhau. Thay vì đó có sự thích hợp với kích cỡ của cơ sở dữ liệu Car, kết quả truy vấn của σ[P](R) tương thích với chất lượng của dữ liệu trong Cars.

Định nghĩa 14. Ngữ nghĩa của σ[P](R)

Giả sử P = (A, <P) và một cơ sở dữ liệu ưa thích P^R. Chúng ta xác định một truy vấn ưa thích σ[P](R) xem xét như là: σ[P](R) = {t ∈ R | t[A] ∈ max(P^R)}

σ[P](R) định giá P áp dụng vào một cơ sở dữ liệu tập P bởi lấy lại tất cả những giá trị lớn nhất từ P^R . Chú ý là không phải tất cả chúng là những giá trị phù hợp nhất của P. Do đó nguồn gốc của truy vấn ưa thích là ẩn trong định nghĩa bên trên. Xa hơn nữa, bất cứ không giá trị lớn nhất nào của P^R là nhận được từ kết quả truy vấn, do đó có thể được xem xét như là bị loại bỏ. Với những giá trị phù hợp nhất- và chỉ duy nhất chúng là nhận được từ truy vấn ưa thích. Do đó chúng tôi đưa ra mô hình truy vấn BMO ("Best Matches Only")

Ví dụ 7 Mô hình truy vấn BMO.

Trong ví dụ này chung tôi đưa ra ưa thích P và truy vấn σ[P](R) cho R(color) = {yellow, red, green, black}. BMO cho kết quả là: σ[P4](R) = {yellow, red}. Chú ý là 'red' là giá trị phù hợp nhất.

Đó là kết quả mong muốn, nhưng quan sát kỹ hơn chúng ta thấy:

Nếu P1 ≡ P2, sau đó cho tất cả R: σ[P1](R) = σ[P2](R)

Bên cạnh các truy vấn ưa thích của dạng σ[P](R) một sự thay đổi sẽ cần được thực hiện thường xuyên hơn, với những khởi tạo từ một tác động lẫn nhau giữa nhóm và nhiều chuỗi.

Nghiên cứu σ[$A^{\leftrightarrow}$&P](R), với P = (B, <P).

Khi đó x <$A^{\leftrightarrow}$&P y nếu x1 = y1 ∧ x2 <P y2 , chúng ta có:

t ∈ σ[$A^{\leftrightarrow}$&P](R) nếu □ v[A, B] ∈ R[A, B]: ¬(t[A] = v[A] ∧ t[B] <P v[B])

Trong điều kiện thuộc nhóm mô tả của R bởi cân bằng những giá trị A, định giá cho mỗi nhóm G_i của những phần tử trong truy vấn ưa thích $\sigma[A^{\leftrightarrow}\&P](G_i)$. Kết quả này đưa ta đến định nghĩa tiếp theo.

Định nghĩa 15 σ[P groupby A](R)

Cho P = (B, <P) và một cơ sở dữ liệu ưa thích P^R, một truy vấn ưa thích với nhóm σ[P groupby A](R) là được định nghĩa như sau: $\sigma[P \text{ groupby } A](R) := \sigma[A^{\leftrightarrow}\&P](R)$

So sánh các truy vấn lựa chọn bắt buộc, truy vấn ưa thích duy nhất từ logic chứa sau mệnh đề lựa chọn bắt buộc: Các truy vấn ưa thích xem như non-monotonically.

Ví dụ 8: Không đơn điệu của truy vấn ưa thích

Cho Cars(Fuel_Economy, Insurance_Rating, Nickname) với:

P := HIGHEST(Fuel_Economy) □

HIGHEST(Insurance_Rating),

Chúng ta định giá σ[P](Cars) cho Cars như sau:

Cars = {(100, 3, frog), (50, 3, cat)}: σ[P](Cars) = {(100, 3, frog)}

Cars = {(100, 3, frog), (50, 3, cat) (50, 10, shark)}:

σ[P](Cars) = {(100, 3, frog), (50, 10, shark)}

Cars = {(100, 3, frog), (50, 3, cat) (50, 10, shark), (100, 10, turtle)}:

σ[P](Cars) = {(100, 10, turtle)}

Mặc dầu chúng tôi đã thêm vào nhiều phần tử, kết quả của các truy vấn ưa thích không bị đưa ra giống nhau. Phù hợp với kích cỡ của Cars, , σ[P](Cars) phù hợp với chất lượng của dữ liệu. Giải thích một cách trực quan như sau: 'better than' không phải là một thuộc tính của giá trị đơn, hơn nữa nó có liên quan đến việc so sánh giữa cặp giá trị. Bởi vậy nó là bị ảnh hưởng tới chất lượng của tập hợp các giá trị, và không tuyệt đối là chất lượng của nó. Do đó "chất lượng thay thế chất lượng" là cái tên của trò chơi cho BMO.

2.2.2 Phân tích các truy vấn hợp rời và giao.

Nhiệm vụ then chốt của đánh giá truy vấn ưa thích là tìm ra những thuật toán hiệu quả nhất cho xây dựng ưu thích phức tạp. Cho phạm vi của luận án này chúng

tôi không nêu ra cụ thể vấn đề hiệu quả ở đây, thay vì đó chúng tôi cung cấp một kết qủa phân tích nền tảng mà có thể là dạng cơ bản cho phân tích và đến gần với tối ưu truy vấn ưa thích. Kết quả là phân tích ưa thích Pareto thành '+' và '♦', mà sẽ có thể được thực hiện trong tương lai.

Mệnh đề 7: σ[P1+P2](R) = σ[P1](R) ∩ σ[P2](R)

Chúng ta sẽ phải áp dụng một số định nghĩa, cho P = (A, <P) và một cơ sở dữ liệu ưa thích P^R.

Định nghĩa 16 Nmax(P^R), $_P\uparrow v$, YY(P1, P2)^R

a) Tập các giá trị không lớn nhất Nmax(P^R) là được định nghĩa như là: Nmax(P^R) := R[A] − max(P^R)

b) Cho v ∈ dom(A), 'better than' tập của v trong P là được định nghĩa như: $_P\uparrow v$:= {w ∈ dom(A): v <P w}

c) YY(P1, P2)^R := {t ∈ R : t[A] ∈ Nmax($P1^R$) ∩ Nmax($P2^R$) ∧ $_{P1}\uparrow t[A] \cap _{P2}\uparrow t[A] = \square$}

Mệnh đề 8. khai triển của truy vấn '♦'

σ[P1♦P2](R)= σ[P1](R) ∪ σ[P2](R) ∪ YY(P1, P2)^R

2.2.3 Phân tích tích lũy ưu tiên

Tiếp theo chúng ta nghiên cứu về σ[P1&P2](R). Với P1&P2 ≡ P1 cho những thuộc tính chia sẻ (Định nghĩa 4a) chúng ta giả sử rằng A1 ∩ A2 = □. Đánh giá cho sự chồng chất ưu tiên có thể được thực hiện bằng grouping.

Mệnh đề 9. σ[P1&P2](R) = σ[P1](R) ∩ σ[P2 groupby A1](R), nếu A1 ∩ A2 = □

Chứng minh: Cho P1 = (A1, <P1) và P2 = (A2, <P2). Từ định nghĩa 4 b, định nghĩa 8 và định nghĩa 16 chúng ta có:

$\sigma[P1\&P2](R) = \sigma[P1+(A1^{\leftrightarrow}\&P2)](R) = \sigma[P1](R) \cap \sigma[A1^{\leftrightarrow}\&P2)](R)$
$= \sigma[P1](R) \cap \sigma[P2 \text{ groupby } A1](R)$

Ví dụ 10. Đánh giá cho truy vấn chồng chất ưu tiên.

Chúng ta giả sử P1 = Make$^{\leftrightarrow}$, P2 = AROUND(Price, 40000) và tập cơ sở dữ liệu Cars(Make, Price, Oid): Cars = {(Audi, 40000, 1), (BMW, 35000, 2), (VW, 20000, 3), (BMW, 50000, 4)} Thông tin yêu cầu " Mõi lần thực hiện cho một kết quả với giá khoảng 40000" cụ thể ta có :

$\sigma[P1\&P2](Cars) = \sigma[P1](Cars) \cap \sigma[P2 \text{ groupby Make}](Cars)$
$= Cars \cap \{(Audi, 40000, 1), (BMW, 35000, 2), (VW, 20000, 3)\}$
$= \{(Audi, 40000, 1), (BMW, 35000, 2), (VW, 20000, 3)\}$

Mệnh đề 11 σ[P1&P2](R) = σ[P2](σ[P1](R)) , nếu P1 là một chuỗi.

Chứng minh: Nếu P1 là một chuỗi, tất cả các phần tử trong σ[P1](R) có cùng A1 giá trị. Khi đó định lý 10 trở thành như đã phát biểu.

Do đó một cascade của truy vấn ưa thích là một trường hợp đặc biệt của truy vấn ưu thích ưu tiên, nếu P1 là một chuỗi.

2.2.4 Phân tích truy vấn tích lũy Pareto.

Bây giờ chúng ta phân tích định lý cho việc đánh giá truy vấn ưa thích Pareto.

Mệnh đề 12 [P1⊗P2](R) = (σ[P1](R) ∩ σ[P2 groupby A1](R)) ∪ (σ[P2](R) ∩ σ[P1 groupby A2](R)) ∪ YY(P1&P2, P2&P1)^R

Chứng minh: Từ mệnh đề 5, mệnh đề 8 và mệnh đề 9 chúng ta có:

$\sigma[P1 \otimes P2](R) = \sigma[(P1\&P2) \blacklozenge (P2\&P1)](R)$
$= \sigma[P1\&P2](R) \cup \sigma[P2\&P1](R) \cup YY(P1\&P2, P2\&P1)^R$
$= (\sigma[P1](R) \cap \sigma[P2 \text{ groupby } A1](R)) \cup$

$(\sigma[P2](R) \cap \sigma[P1 \text{ groupby } A2](R)) \cup YY(P1\&P2, P2\&P1)^R$

Mệnh đề này cho một sự hiểu biết sâu sắc bên trong cấu trúc của tập Pareto tối ưu σ[P1□P2](R), nhấn mạnh lại sự khẳng định rõ ràng '□' cho biết P1 và P2 là quan trọng như nhau.

Số hạng đầu tiên chứa tất cả các giá trị lớn nhất của $(P1\&P2)^R$.

Số hạng thứ 2 chứa tất cả các giá trị lớn nhất của $(P2\&P1)^R$

Số hạng thứ 3 chứa tất cả các giá trị hoặc lớn nhất không nằm trong $(P1\&P2)^R$ hoặc không trong $(P2\&P1)^R$.

Chú ý là nếu P1 hoặc P2 là một chuỗi, thì định lý 11 có thể được dùng cho đánh giá tốc độ.

Ví dụ 11. Đánh giá tích lũy Pareto.

Giả sử P1 = LOWEST(A), ưa thích P2 = HIGHEST(A) và R(A) = {3, 6, 9}. Chúng ta tính toán σ[P1□P2](R). Từ định lý 6, định lý 3d, g) chúng ta nhanh chóng biết được:

σ[P1□P2](R) = σ[P1♦P2](R) = σ[P1♦P1∂](R) = σ[$A^{\leftrightarrow}$](R) = R

Nhằm ngăn chặn khi P1 và P2 là chuỗi. Mệnh đề 12 thực hiện như sau:

σ[P1□P2](R) = σ[P2](σ[P1](R)) ∪ σ[P1](σ[P2](R)) ∪ YY(P1&P2, P2&P1)R

= {3} ∪ {9} ∪ YY(P1&P2, P2&P1)R

Chúng ta có: Nmax((P1&P2)R) ∩ Nmax((P2&P1)R) = {6, 9} ∩ {3, 6} = {6}

Khi đó P1&P2↑6 ∩ P2&P1↑6 = {3} ∩ {9} = □,

chúng ta có YY(P1&P2, P2&P1)R = {6}

Do đó chúng ta có kết quả cuối cùng là: σ[P1□P2](R) = {3} ∪ {9} ∪ {6} = R

2.2.5 Hiệu quả phép lọc của các truy vấn Pareto

Các truy vấn ưa thích dùng BMO tránh được kết quả rỗng và kết quả quá nhiều. Trong trường hợp khác, bộ máy tìm kiếm với một mô hình truy vấn phù hợp cố

gắng gạt bỏ những phiền toái bởi đưa ra những miếng vá giống như giới hạn tìm kiếm, chúng thực hiện bán tự động, cố gắng đạt được truy vấn tối ưu nhất.

Chúng tôi muốn nghiên cứu một cách đầy đủ hơn về hiệu quả của phép lọc của truy vấn ưa thích sử dụng mô hình BMO. Cho P = (A, <P) là kết quả của truy vấn σ[P](R) được xác định như sau:

size(P, R) := card(π_A(σ[P](R)) = card(max(P^R))

Định nghĩa 15 Thế mạnh của phép lọc ưa thích

Cho P1 = (A, <P1) và P2 = (A, <P2), P1 là phép lọc ưa thích mạnh hơn P2 (P1→ P2), nếu:

size(P1, R) ≤ size(P2, R).

Mệnh đề 13 Sự ưu việt trong phép lọc của ưa thích phức tạp

a) P1+P2 → P1, P1+P2 → P2

b) P1 → P1♦P2, P2 → P1♦P2

c) P1&P2 → P1

d) P1&P2 → P1□P2, P2&P1 → P1□P2

Hãy xem giải thích hiệu quả của phép lọc của tích lũy Pareto trong phép suy diễn qua phép toán Boolean 'AND/OR' thực hiện trong một bộ máy tìm kiếm sử dụng mô hình truy vấn phù hợp nhất. Chúng ta có trạng thái:

P1□P2 ← P1&P2 → P1, P1□P2 ← P2&P1 → P2

Từ đó cho ta thấy sự hiểu quả của phép lọc sử dụng mô hình BMO.

2.3 Tối ưu truy vấn ưa thích quan hệ.

2.3.1 Đại số quan hệ ưa thích.

Trong phạm vi của chương này, chúng ta quan tâm đến trường hợp quan hệ không đệ quy, mặc dầu mô hình ưa thích của chúng ta là được ứng dụng cho trường hợp tổng quát của cơ sở dữ liệu suy diễn đệ quy.

Đại số quan hệ ưa thích bao gồm 2 tập các phép toán sau:

- Đại số quan hệ chắc chắn:

Phép chọn cứng H(R) cho một điều kiện Boolean H, phép chiếu p(R), phép hợp $R \cup S$, tích đề các RxS.

- Các phép toán ưa thích:

Chọn ưa thích s[P](R)

Chọn ưa thích nhóm s[P groupby B](R)

Công bằng với các định lý diễn tả có ý nghĩa mạnh của đại số quan hệ ưa thích là cũng như đại số quan hệ cổ điển. Do đó các câu truy vấn ưa thích dùng BMO có thể được viết lại vào trong đại số quan hệ (nó được thực hiện bởi SQL ưa thích). Nó cũng làm ý bao gồm vấn đề tối ưu cho đại số quan hệ ưa thích là không khó hơn cho đại số quan hệ cổ điển, ví dụ. Trong ưa thích nói chung có thể được tích hợp vào trong SQL với hiệu năng cao.

2.3.2 Ngữ nghĩa toán tử của truy vấn ưa thích.

Hạn chế về ngữ nghĩa của ngôn ngữ truy vấn đưa ra trong yêu cầu chung một ngữ nghĩa trên phương diện lý thuyết và một quan điểm cụ thể để thực hiện tối ưu hóa truy vấn. Đối với SQL ưa thích nhiệm vụ này có thể được đưa vào nền tảng lớn của Datalog-S, Các mô hình sắp xếp sẽ áp dụng và các tối ưu sắp xếp.

Khi đó chúng ta tập trung vào tối ưu đại số của truy vấn ưa thích quan hệ, chúng ta có thể khai thác tính tương đương của đại số quan hệ và đại số quan hệ ưa thích nhằm đưa ra định nghĩa về ngữ nghĩa. Chúng ta xem xét đến câu truy vấn Q trong cú pháp SQL ưa thích. Sau đó ngữ nghĩa toán tử của Q là được định nghĩa như sau:

If <mệnh đề nhóm không tồn tại>

then $\pi_{\mathbf{T}}$ (σ[P](σ_{H}(R1 x ... x Rn)))

else $\pi_{\mathbf{T}}$ (σ[P groupby{B1, …, Bm}] (σ_{H}(R1 x ... x Rn)))

Điều này là hợp với quy tắc mở rộng của trường hợp quan hệ tương đương. Nghiên cứu kỹ ví dụ 2 của chúng ta, ngữ nghĩa thực thi của truy vấn ưa thích là như sau:

Ví dụ 2 : Ngữ ngữ toán tử

Micheal là một người giàu có, anh ấy có thể bỏ ra hơn 35000Euro để mua xe hơi. Anh ấy mong muốn nhãn hiệu xe hơi nên là BMW hoặc Porsche, thời gian sử dụng khoảng trong vòng 3 năm và màu sắc không nên là màu đỏ. Tất cả điều kiện là quan trọng ngang bằng nhau đối với anh ấy"

$\pi_{u.price,c.brand,u.age,u.color}$

(σ[POS(c.brand,{'BMW','Porsche'}) ⊗

AROUND(u.age,3) ⊗ NEG(color,{'red'})]

($\sigma_{u.ref=c.ref \wedge u.price<=35000}$

(used_cars u × category c)))

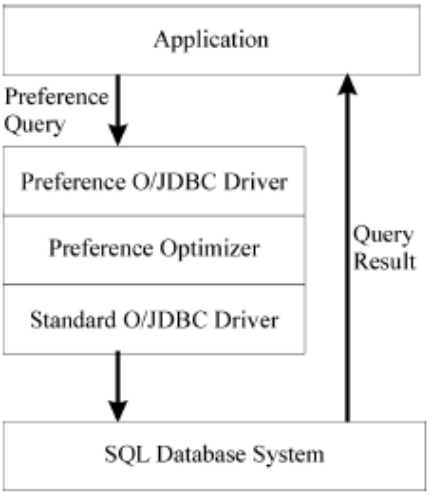
Sự diễn tả đại số quan hệ ưa thích ban đầu sẽ là chủ đề cho phương thức tối ưu đại số được phát triển tiếp theo.

2.3.3 Vấn đề thiết kế kiến trúc.

Mở rộng một SQL tồn tại bởi các câu truy vấn ưa thích yêu cầu một số thiết kế cốt yếu quyết định cho tối ưu truy vấn ưa thích.

- *Kiến trúc tiền xử lý ghép nối lỏng:*

Các câu truy vấn ưa thích được xử lý bởi viết lại chúng thành SQL chuẩn và áp dụng chúng vào cơ sở dữ liệu SQL. Phiên bản hiện tại của SQL ưa thích tiếp theo như là một cặp lỏng lẻo, sử dụng truy vấn ưa thích trong nhiều ứng dụng thương mại.



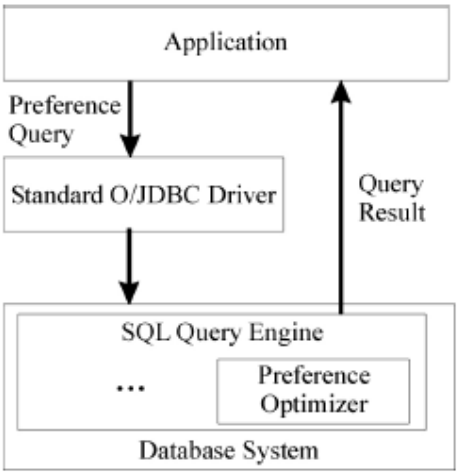
Hình1. Tiền xử lý gần đúng cho các truy vấn ưa thích

- *Kiến trúc ghép nối:*

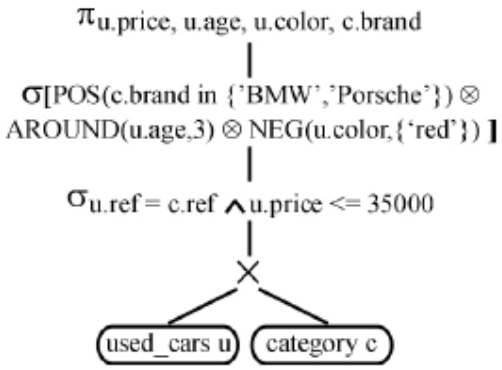
Sự tích hợp tối ưu truy vấn ưa thích và tối ưu SQL chặt hứa hẹn tăng hiệu năng xử lý. Sự trở ngại có thể là những thực hiện này yêu cầu thao tác thực hiện chính xác với cơ sở dữ liệu SQL.

Dưới đây là vấn đề tối ưu cho các truy vấn ưa thích có thể được ánh xạ vào đại số quan hệ ưa thích. Sự thật là chúng ta có 2 kiểu cổ điển của tối ưu truy vấn cơ sở dữ liệu. Cho một truy vấn ưa thích Q.

1. Ngữ nghĩa của Q xác định một cây thao tác khởi tạo T_{start} , những đỉnh của nó là những toán từ từ đại số quan hệ ưa thích.
2. T_{start} là được tối ưu sử dụng một số tập luật chuyển đổi đại số. Những luật này được sử dụng, và thứ tự, phải được điều khiển bởi một số tính toán khôn ngoan mà độ thông minh được giảm bớt với không gian tìm kiếm lớn bằng hàm mũ. Hãy xem T_{end} xác định cây cuối cùng.
3. Đại số quan hệ ưa thích hoạt động trong T_{end} phải được ánh xạ vào thuật toán ước lượng hiệu quả. Nếu có nhiều lựa chọn, tối ưu dựa trên giá trị là được lựa chọn.



Hình 2. Ước lượng truy vấn ưa thích ghép nối chặt



Hình 3. Thao tác tạo cây cho ví dụ 2.

2.4 Các luật đại số quan hệ ưa thích.

Nhằm nâng cáo vị thế của tối ưu đề cập trước đây đến gần với sự hiểu biết sâu sắc về các luật chuyển đổi đại số cơ sở dựa trên đại số quan hệ ưa thích là được yêu cầu. Cho ví dụ, chiến lược khám phá thành công của tối ưu truy vấn quan hệ đại số là "đẩy phép chọn cứng" và "đẩy phép chiếu". chúng ta đưa ra ý kiến này bởi những luật chuyển đổi phát triển cho đại số quan hệ ưa thích mà cho phép chúng ta thực hiện "đẩy ưa thích " trong những cây toán tử.

2.4.1 Các luật chuyển đổi.

Một số lời chú giải của các thuyết sau đây tham chiếu đến trái qua phải chuyển đổi 'push'. Chúng tôi dùng attr(R) nhằm đưa ra tất cả các thuộc tính của quan hệ R.

Chú ý, trong luật này đúng với những giới hạn không gian duy nhất đúng với các luật quan trọng đưa ra. Cho cái nhìn đầy đủ của các luật đại số và tùy thuộc vào chứng minh đưa ra của KieBling và Hafenrichter. Cũng đánh số của các thuyết sau đã từng được nêu ra trong luận văn này.

Định lý L1: *Đẩy ưa thích lên toàn phép chiếu*

Cho P = (A, <P) và A, X $\subseteq$ attr(R).

a) $\sigma[P](\pi_x(R)) = \pi_x(\sigma[P](R))$ nếu A $\subseteq$X

b) $\pi_x(\sigma[P](\pi_x \cup A(R))) = \pi_x(\sigma[P](R))$ cho các trường hợp khác

Chứng minh: a) Cho P = (A,<P), A □ X □ attr(R),

thì:$\pi_x(\sigma[P](R))= \pi_x(\{w \in R| \neg\Box v \in R : w[A] <P v[A]\})$ // Bổ đề 1

$= \pi_x(\{w \in R| \neg\Box v \in R : w <X v\})$

$= \{w \in \pi_x(R) | \neg\Box v \in \pi X(R) : w <X v\}$ // Bổ đề 1

$= \sigma[P](\pi_x(R))$

Định lý L2: *Đẩy ưa thích lên toàn tích Đề các*

Cho P = (A, <P) và A $\subseteq$ attr(R).

$\sigma[P](R \times S) = \sigma[P](R) \times S$

Chứng minh: R × S có thể được viết lại theo điều kiện của hàm mở rộng:

R × S = R ext× S, ×(r, s)

= true$\sigma[P](R \times S)= \sigma[P](R \text{ ext}\times S)$ // Bổ đề 2

$= \sigma[P](R) \text{ ext}\times S = \sigma[P](R) \times S$

Định lý L3: *Đẩy ưa thích lên toàn phép hợp*

Cho P = (A, <P) và A $\subseteq$ attr(R) = attr(S).

$\sigma[P](R \cup S) = \sigma[P](\sigma[P](R) \cup \sigma[P](S))$

Chứng minh: Let T := $\sigma[P](R) \cup \sigma[P](S)$.

$\sigma[P](T)$

$= \{w \in T | \neg\Box v \in T : w[A] <P v[A]\}$ // T

$= \{w \in T | \neg((\Box v \in \sigma[P](R) : w[A] <P v[A]) \vee$

$(\Box v \in \sigma[P](S) : w[A] <P v[A]))\}$

$= \{w \in T \mid \neg((\square v \in R : w[A] <_P v[A] \wedge v \in \sigma[P](R)) \vee$

$(\square v \in S : w[A] <_P v[A] \wedge v \in \sigma[P](S)))\}$

$= \{w \in T \mid \neg((\square v \in R : w[A] <_P v[A]) \vee$

$(\square v \in S : w[A] <_P v[A]))\}$

$= \{w \in R \cup S \mid \neg(\square v \in R \cup S: w[A] <_P v[A]) \wedge w \in T\}$

$= \{w \in R \cup S \mid \neg(\square v \in R \cup S : w[A] <_P v[A])\}$

$= \sigma[P](R \cup S)$

Định lý L4 : *Tách ưu tiên trong grouping*

Cho $P_1 = (A_1, <_{P_1})$, $P_1 = (A_2, <_{P_2})$ và $A_1, A_2 \subseteq$ attr(R).

$\sigma[P_1 \,\&\, P_2](R) = \sigma[P_2 \text{ groupby } A_1](\sigma[P_1](R))$

Chứng minh:σ[P2 groupby A1](σ[P1](R))

$= \{w \in \sigma[P1](R) \mid \neg\square v \in \sigma[P1](R) :$

$w[A1] = v[A1] \wedge w[A2] <_{P2} v[A2] \}$

$= \{w \in R \mid \neg\square v \in R: w[A1]$

$= v[A1] \wedge w[A2] <_{P2} v[A2] \wedge v \in \sigma[P1](R) \wedge w \in \sigma[P1](R)\}$

// $w \in \sigma[P1](R)$, $w[A1] = v[A1]$ dẫn đến $v \in \sigma[P1](R)$

$= \{w \in R \mid \neg\square v \in R: w[A1] = v[A1] \wedge w[A2] <_{P2} v[A2] \wedge w \in \sigma[P1](R)\}$

$= \{w \in R \mid \neg\square v \in R: w[A1]$

$= v[A1] \wedge w[A2] <_{P2} v[A2]\} \cap \sigma[P1](R)\}$

$= \sigma[P1](R) \cap \sigma[P2 \text{ groupby } A1](R) = \sigma[P1 \,\&\, P2](R)$.

Hệ quả 1: *Đơn giản hóa ưu tiên*

$\sigma[P_1 \,\&\, P_2](R) = \sigma[P_2](\sigma[P_1](R))$ nếu P_1 là một chuỗi

Các ưa thích LOWEST và HIGHEST là các chuỗi, i.e. thứ tự tổng bậc. Nếu P_1 và P_2 là các chuỗi, $P_1 \,\&\, P_2$ là một chuỗi.

Gọi lại định nghĩa của ngữ nghĩa thực hiện của truy vấn ưa thích Q làm người đọc có thể từng hỏi chính người đó, tại sao σ_H là được dùng trước $\sigma[P]$ và không ngược

lại versa, và được hay không điều này làm nên một sự khác biệt. Nó trả lại σ[P] có thể trao đổi với σ_H duy nhất ở dưới một tiền điều kiện mạnh hơn

Định lý L5: *Đẩy ưa thích lên phép chọn cứng*

$\sigma[P](\sigma_H(R)) = \sigma_H(\sigma[P](R))$ nếu

$\forall w \in R: (H(w) \wedge \exists v \in R: w[A] <_P v[A]$ đưa đến $H(v))$

Chứng minh: Chúng ta có một bổ đề phụ trợ sau:

$\sigma_H(\sigma[P](R)) \sqsupset \sigma[P](\sigma_H(R))$

Chứng minh cho bổ đề phụ trợ:$w \in \sigma_H(\sigma[P](R))$

nếu $\neg(\sqsupset v \in R: w[A] <_P v[A]) \wedge H(w)$

nếu $\neg(\sqsupset v \in R: w[A] <_P v[A] \wedge H(w))$ // *1*

Trong trường hợp khác chúng ta có:$w \in \sigma[P](\sigma_H(R))$

nếu $w \in \sigma_H(R) \wedge H(w)$

nếu $\neg(\sqsupset v \in R: w[A] <_P v[A] \wedge H(v)) \wedge H(w)$

nếu $\neg(\sqsupset v \in R: w[A] <_P v[A] \wedge H(v) \wedge H(w))$ // *2*

Chúng ta có thể kết luận:$\sigma_H(\sigma[P](R)) \sqsupset \sigma[P](\sigma_H(R))$

nếu $\sqsupset w \in R$: *1* dẫn đến *2*

nếu $\sqsupset w \in R : (\sqsupset v \in R: w[A] <_P v[A] \wedge H(v) \wedge H(w)$

dẫn đến $\sqsupset v \in R: w[A] <_P v[A] \wedge H(w))$

nếu đúng

Bao gồm cả tập nghịch đảo, chúng ta cũng yêu cầu cho định lý L5, có được chỉ duy nhất cho trạng thái điều kiện không quan trọng. Chúng ta bắt đầu qua những dòng đánh nhãn *1* và *2* bên trên:$w \in \sigma H(\sigma[P](R))$

nếu $\neg(\sqsupset v \in R: w[A] <_P v[A] \wedge H(w))w \in \sigma[P](\sigma H(R))$

nếu $\neg(\sqsupset v \in R: w[A] <_P v[A] \wedge H(v) \wedge H(w))$

Chúng ta có thể kết luận:$w \in \sigma[P](\sigma H(R))$ dẫn đến $w \in \sigma H(\sigma[P](R))$

nến $\square$ v $\in$ R: w[A] <P v[A] $\wedge$ H(w) dẫn đến $\square$ v $\in$ R: w[A] <P v[A] $\wedge$ H(v) $\wedge$ H(w)

nếu $\square$ v $\in$ R: w[A] <P v[A] $\wedge$ H(w) dẫn đến H(v)

nến H(w) $\wedge$ $\square$ v $\in$ R: w[A] <P v[A] dẫn đến H(v)

Mệnh đề 2: *Trường hợp đặc biệt của định lý L5*

Cho P_1 := LOWEST(A), P_2 := HIGHEST(A).

a) $\sigma[P_1](\sigma_{A<=c}(R)) = \sigma_{A<=c}(\sigma[P_1](R))$

b) $\sigma[P2](\sigma_{A>=c}(R)) = \sigma_{A>=c}(\sigma[P_2](R))$

Quan hệ kết nối là một lựa chọn bắt buộc qua tích Đề các, lấy σ[P] qua phép kết nối là khó khăn hơn.

Định lý L6 *Đẩy ưa thích lên phép kết nối*

Cho P = (A, <P), A $\subseteq$ attr(R) và X $\subseteq$ attr(R) $\cap$ attr(S).

a) $\sigma[P](R \bowtie_{R.X=S.X} S) = \sigma[P](R) \bowtie_{R.X=S.X} S$,

Nếu mỗi phần tử trong R có ít nhất một đối tác kết nối trong S

Cho $R \bowtie_{R.X=S.X} S$ xác định một thao tác semi-join

b) $\sigma[P](R \bowtie_{R.X=S.X} S) =$

$\sigma[P](R \bowtie_{R.X=S.X} S) \bowtie_{R.X=S.X} S$

c) $\sigma[P](R \bowtie_{R.X=S.X} S)) =$

$\sigma[P](\sigma[P \text{ groupby } X](R) \bowtie_{R.X=S.X} S)$

Xa hơn nữa cho B $\subseteq$ attr(S).

d) $\sigma[P \text{ groupby } B](R \bowtie_{R.X=S.X} S) =$

$\sigma[P \text{ groupby } B](\sigma[P \text{ groupby } X](R) \bowtie_{R.X=S.X} S))$

Chú ý là trong trường hợp luật riêng L6a là có thể áp dụng được, nếu

R.X là một *foreign key* tham chiếu đến S.X. Nếu nó không phải là siêu thông tin chấp nhận được, thì L6b rõ ràng được tính toán tất cả đối tác tham gia. Thỉnh thoảng sự ưa thích phải được tách trước khi một phần của nó có thể được đặt vào phép toán kết nối.

Định lý L7: *Chia tách ưa thích Pareto và đẩy lên phép kết nối*

Let $P_1 = (A_1, <P_1)$ với $A_1 \subseteq$ attr(R), $P_2 = (A_2, <P_2)$

với $A_2 \subseteq$ attr(S). Cho $X \subseteq$ attr(R) $\cap$ attr(S).

$\sigma[P_1 \otimes P_2](R \bowtie_{R.X=S.X} S) =$

$\sigma[P_1 \otimes P_2] (\sigma[P_1 \text{ groupby } X](R) \bowtie_{R.X=S.X} S)$

Chứng minh: Cho T := R ⋈R.X=S.XS.

// Bổ đề 4

σ[P1 □ P2](T) □σ[P1 groupby A2](T) // bổ đề 3

$\sigma[P1 \square P2](R \bowtie_{R.X=S.X} S)$

$= \sigma[P1 \square P2](\sigma[P1 \text{ groupby } A2](R \bowtie_{R.X=S.X} S))$ //Định lý L6d

$= \sigma[P1 \square P2](\sigma[P1 \text{ groupby } A2] (\sigma[P1 \text{ groupby } R.X](R \bowtie_{R.X=S.X} S))$

// Bổ đề 3

$= \sigma[P1 \square P2](\sigma[P1 \text{ groupby } R.X](R) \bowtie_{R.X=S.X} S)$

Định lý L8: *Chia tách thứ tự ưu tiên và đẩy lên phép kết nối*

Cho $P_1 = (A_1, <P_1)$ với $A_1 \subseteq$ attr(R), $P_2 = (A_2, <P_2)$

với $A_2 \subseteq$ attr(R) $\cup$ attr(S) và cho $X \subseteq$ attr(R) $\cap$ attr(S).

a) $\sigma[P_1 \& P_2](R \bowtie_{R.X=S.X} S) =$

$\sigma[P_2 \text{ groupby } A_1](\sigma[P_1](R) \bowtie_{R.X=S.X} S)$,

Nếu mỗi phần tử trong R có ít nhất một đối tác tham gia trong S

b) $\sigma[P_1 \& P_2](R \bowtie_{R.X=S.X} S) =$

$\sigma[P_1 \text{ groupby } A_1](\sigma[P_1](R \ltimes_{R.X=S.X} S) \bowtie_{R.X=S.X} S)$.

Chứng minh: a)

$\sigma[P1 \& P2](R \bowtie_{R.X=S.X} S)$ // định lý L4

$= \sigma[P2 \text{ groupby } A1](\sigma[P1](R \bowtie_{R.X=S.X} S))$ // định lý L6a

$= \sigma[P2 \text{ groupby } A1](\sigma[P1](R \bowtie_{R.X=S.X} S)$.

Chứng minh: b)

$\sigma[P1 \ \& \ P2](R \bowtie_{R.X=S.X} S)$ // định lý L4

$= \sigma[P2 \text{ groupby } A1](\sigma[P1](R \bowtie_{R.X=S.X} S))$ // định lý L6c

$= \sigma[P2 \text{ groupby } A1](\sigma[P1](R \ltimes_{R.X=S.X} S) \bowtie_{R.X=S.X} S)$

Liên quan đến tiền điều kiện dựa trên đối tác thạm gia trong cùng sự chú ý như là cũng áp dụng cho L6.

Tiếp theo chúng ta có một số luật thường sử dụng cho các lựa chọn ưa thích phức tạp.

Định lý L9: *Đơn giản hóa lựa chọn ưa thích nhóm*

Let $P = (A, <P)$, $P_1 = (A_1, <P_1)$, $P_2 = (A_2, <P_2)$ và A, B, B_1,

$B_2 \subseteq$ attr(R).

a) $\sigma[(P \text{ groupby } B_1) \text{ groupby } B_2](R) = \sigma[P \text{ groupby } (B_1 \cup B_2)](R)$

b) $\sigma[(P_1 \text{ groupby } B) \ \& \ P_2](R) = \sigma[(P_1 \ \& \ P_2) \text{ groupby } B](R)$

c) $\sigma[P_1 \otimes (P_2 \text{ groupby } B)](R) = \sigma[(P_1 \otimes P_2) \text{ groupby } B](R)$

d) $\sigma[(P_1 \text{ groupby } B_1) \otimes (P_2 \text{ groupby } B_2)](R) = \sigma[(P_1 \otimes P_2) \text{ groupby } (B_1 \otimes B_2)](R)$

e) $\sigma[P \text{ groupby } B](R) = R$ nếu B là khóa trong R

Tiền điều kiện trong định lý L5 là thường xuyên bị lỗi. Tuy nhiên, trong một số trường hợp chúng ta có thể dùng các luật khác để thay thế sự ưa thích và các lựa chọn bắt buộc.

Định lý L10: *Sự thay thế của ưa thích và lựa chọn bắt buộc*

a) Cho $P = (A, <P)$ và $A, B \subseteq$ attr(R):

$\sigma[P](\sigma_{B=c}(R)) = \sigma_{B=c}(\sigma[P \text{ groupby } B](R))$

$\sigma[P](\sigma_{A=c}(R)) = \sigma_{A=c}(R)$

b) Cho P := BETWEEN(A, $[c_1, c_2]$):

$\sigma[P](\sigma_{A>=c}(R)) = \sigma[\text{LOWEST}(A)](\sigma_{A>=c}(R))$, nếu $c > c_2$

$\sigma[P](\sigma_{A<=c}(R)) = \sigma[\text{HIGHEST}(A)](\sigma_{A<=c}(R))$, nếu $c < c_1$

$\sigma[P](\sigma_{A>=c1 \wedge A<=c2}(R)) = \sigma_{A>=c1 \wedge A<=c2}(R)$

Cho H := '$A = c_1 \vee ... \vee A = c_n$', H-set := $\{c_1, ..., c_n\}$.

c) Cho P := POS(A, Pos-set), Pos-set = $\{v_1, ..., v_m\}$:

Nếu H-set ∩ Pos-set = Ø hoặc H-set ⊆ Pos-set

thì $\sigma[P](\sigma_H(R)) = \sigma_H(R)$

d) Cho P := NEG(A, Neg-set), Neg-set = $\{v_1, ..., v_m\}$:

nếu H-set ∩ Neg-set = Ø hoặc H-set ⊆ Neg-set

thì $\sigma[P](\sigma_H(R)) = \sigma_H(R)$

e) Cho P = POS/NEG(A, POS-set; NEG-set),

POS-set=$\{v_1, ..., v_m\}$, NEG-set = $\{v_{m+1}, ..., v_{m+n}\}$:

Nếu H-set ⊆ POS-set hoặc H-set ⊆ NEG-set hoặc

H-set ∩ (POS-Set ∪ NEG-Set) = Ø

thì $\sigma[P](\sigma_H(R)) = \sigma_H(R)$

f) Nếu P = POS/POS(A, POS1-set; POS2-set),

POS1-set=$\{v_1, ..., v_m\}$, POS2-set = $\{v_{m+1}, ..., v_{m+n}\}$:

Nếu H-set ⊆ POS1-set hoặc H-set ⊆ POS2-set hoặc

H-set ∩ (POS1-Set ∪ POS2-Set) = Ø

thì $\sigma[P](\sigma_H(R)) = \sigma_H(R)$

Khi đó AROUND là một xây dựng phụ ưa thích của BETWEEN

Chúng ta có hệ quả khác như sau.

Mệnh đề 3: *AROUND kế thừa từ BETWEEN*

Định lý L10b áp dụng cho P := AROUND(A, z).

Định lý L11: *Luật giao hoán và kết hợp*

a) $\sigma[P_1 \otimes P_2](R) = \sigma[P_2 \otimes P_1](R)$

b) $\sigma[(P_1 \otimes P_2) \otimes P_3](R) = \sigma[P_1 \otimes (P_2 \otimes P_3)](R)$

c) $\sigma[(P_1 \,\&\, P_2) \,\&\, P_3](R) = \sigma[P_1 \,\&\, (P_2 \,\&\, P_3)](R)$

Luật L11 cũng đưa đến các luật dựa trên ưa thích Paretl giống như L7 là luật giao hoán.

Chú ý là khi đó 'P groupby B' là một ưa thích chính nó, trong mỗi luật trước đây, sự ưa thích có thể được nhóm vào hoặc không, trừ khi bắt buộc được yêu cầu giống như trong luật L9. Cho ví dụ, cho một ưa thích grouping L1 giống như sau:

$\pi_x(\sigma[P \text{ groupby } B](R)) = \sigma[P \text{ groupby } B](\pi_x(R))$ nếu A, B ⊆ X

2.4.2 Tích hợp với tối ưu hóa truy vấn quan hệ.

Bởi vì đại số quan hệ ưa thích là mở rộng của đại số quan hệ, chúng ta xây dựng một sự tối ưu hóa truy vấn ưa thích như là một mở rộng của tối ưu hóa truy vấn quan hệ truyền thống. Một điều rất quan trọng, chúng ta kế thừa tất cả các luật thông thường từ đại số quan hệ. Do đó chúng ta sẽ sử dụng các phương pháp mang tính kinh nghiệm nhằm làm giảm kích cỡ của các quan hệ trung gian, ví dụ. 'đẩy phép chọn cứng' và 'đẩy phép chiếu'. Hãy xem thuật toán leo đồi cơ bản sau, cho một bước phù hợp của những chuyển đổi đại số quan hệ: Thuật toán Pass-1

{ update T:

Step 1-1: <tách các lựa chọn bắt buộc>;

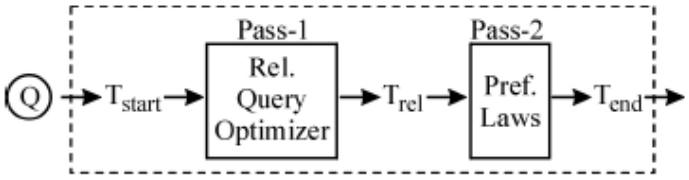
Step 1-2: <thực hiện các lựa chọn bắt buộc nếu có thể>;

Step 1-3: <thực hiện các phép chiếu nếu có>;

Step 1-4: <kết hợp các lựa chọn cứng và tích Decac vào trong phép join>;

return T**}**

Phần triển khai cho ta một bước của chuyển đổi quan hệ bởi những luật mới L1 thành L11 cho ta thấy vấn đề tích hợp chúng vào trong các thủ tục trên như thế nào. Trong trường hợp cá biệt, chúng tôi muốn thêm vào chiến lược sử dụng kinh nghiệm của 'đẩy ưa thích'. Trộn lẫn các biến đối mới và trong chiến lược tối ưu đưa ra sẽ cho một sự mềm dẻo cao nhất. Nhưng từ một điểm kỹ nghệ phần mềm của tầm nhìn có thể có vấn đề. Một khả năng kém mềm dẻo hơn, nhưng cách đưa ra là mở rộng tối ưu truy vấn quan hệ tồn tại bởi phương pháp thứ 2 như minh họa dưới đây.



Hình 4. Nguyên lý two-pass cho tối ưu đại số

Cho một truy vấn ưa thích Q, cây khởi tạo của nó T_{start} là dùng cho Pass-1, ban đầu, công việc của nó tại cây con thuộc đỉnh ưa thích. Cây đầu ra Tre1 là được điều khiển qua Pass-2 mới của chúng ta, hoạt động như thuật toán dưới đây:

Thuật toán Pass-2:

{ repeat <traverse T in a depth-first way>

{ Step 2-1: // *ưu tiên*

Áp dụng hệ quả 1;

Step 2-2: // *đẩy ưa thích lên phép chiếu,*

Tích Đề các hoặc kết hợp

Áp dụng L1a; L1b; L2; L3;

Step 2-3: // *đẩy ưa thích lên phép kết nối*

Áp dụng L6a; L6b;

Step 2-4: // *chia tách ưa thích và đẩy lên phép kết nối*

Áp dụng L7; L9e; L8a; L8b };

Step 2-5: // đẩy phép chiếu nếu có thể được

Áp dụng Step 1-3 bao gồm L1a, L1b trong chế độ 'pull';

return T**}**

Phân tích thuật toán, nếu chuyển đổi 'đẩy ưa thích' là được sử dụng và cập nhật tương ứng với T. Tìm một hạng 'good' của luật chuyển đổi là thường gặp khó khăn, kinh nghiệm là chính. Bắt đầu với hệ quả 1 trong bước 2-1 nên luôn luôn đúng. Cũng như các trường hợp bước 2-3 nên đến trước bước 4 bởi vì phép lọc hiệu quả tốt hơn.

2.4.3 Các vấn đề cần nghiên cứu

Chúng ta hãy xem lại ví dụ 2, thêm vào một số quan hệ khác:

seller(*sid*,name,street,zipcode,city)

Trong used_cars chúng ta giả sử là *ref* và *sid* là khóa ngoại từ category và seller, và Car# là khóa chính.

Ví dụ 3: *Truy vấn ưu thích phức tạp Q1*

SELECT s.name, u.price

FROM used_cars u, category c, seller s

WHERE u.sid = s.sid AND u.ref = c.ref AND

(u.color = 'red’ OR u.color = 'gray’)

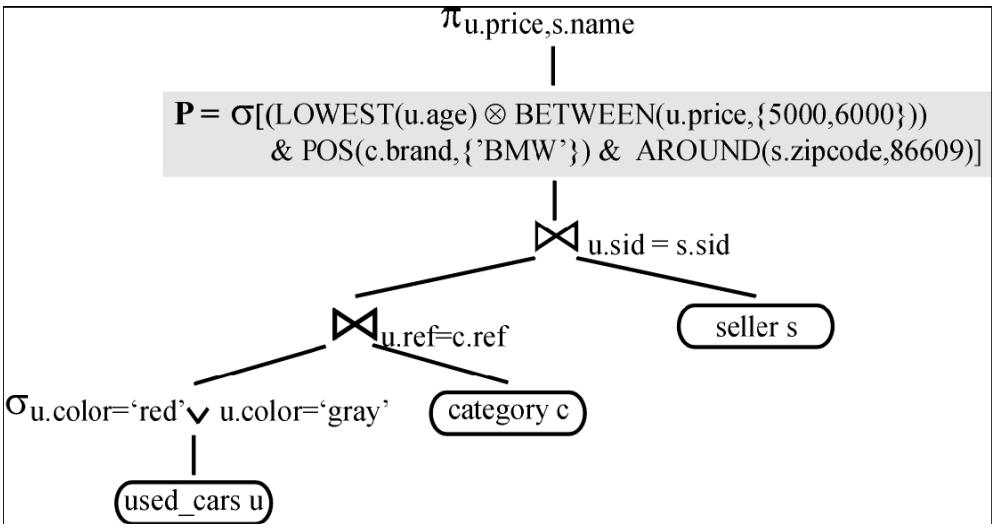
PREFERRING

(LOWEST u.age AND

u.price BETWEEN 5000, 6000)

PRIOR TO c.brand IN ('BMW')

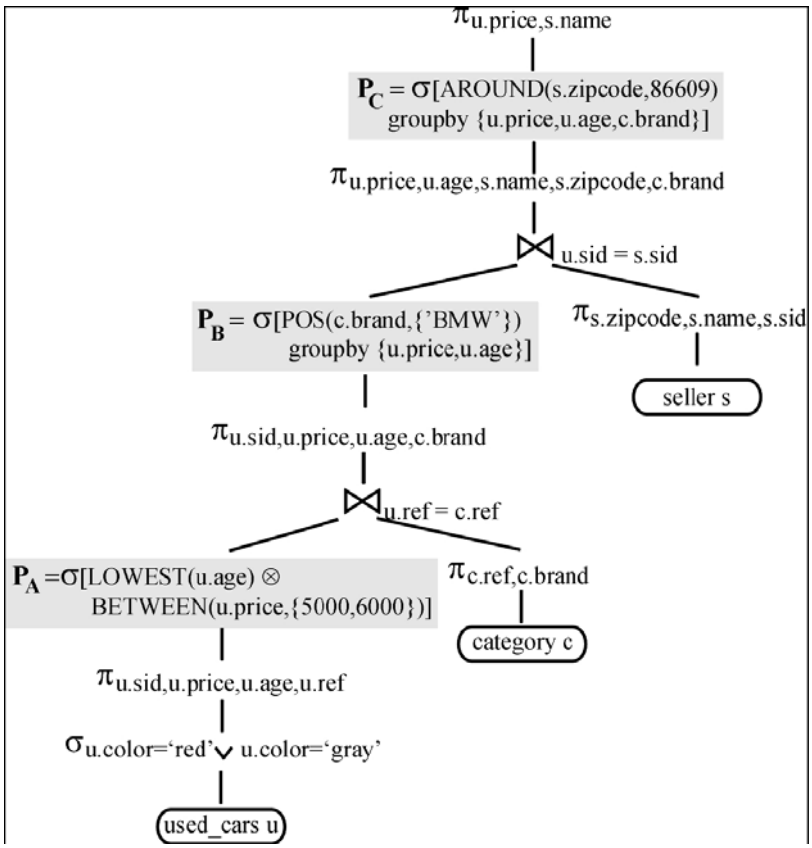
PRIOR TO s.zipcode AROUND 86609;



Hình 5. T_{rel} sau pass 1.

Cây T_{rel} được xem là đầu ra của Pass-1 như trong hình 5

5. Cho T_{rel} Pass-2 thực hiện tuần tự các chuyển đổi ưa thích để đi đến kết quả cây T_{end} như trong hình 6: **L8a; L8a; L1b; L1a; L1a**



Hình 6: T_{end} sau pass 2.

Hãy so sánh T_{end} với T_{rel}:

- *Chi phí phép kết nối*: Toán hạng của phép kết nối bên trái thấp hơn trong T_{end} là bị giảm bớt bởi lựa chọn ưa thích thực hiện P_A như trong hình 6. Sự quy vòng này, được tăng lên bởi lựa chọn ưa thích P_B, làm giảm kích thước của toán hạng kết nối bên trái cao hơn trong T_{end}.
- *Chi phí ưa thích*: P_A là đơn giản hơn so với cây ban đầu P trong T_{rel}, nhưng nó vẫn là sự ưa thích ưu tiên hơn. Tuy nhiên, cả hai P_A và P_C là đơn giản hơn, nhóm ưa thích cơ bản.

S*kyline queries* đã được nghiên cứu trong ưa thích Pareto. Đây là một dạng cú pháp SQL ưa thích.

Ví dụ 4: *Skyline query Q2*

SELECT s.name, u.price

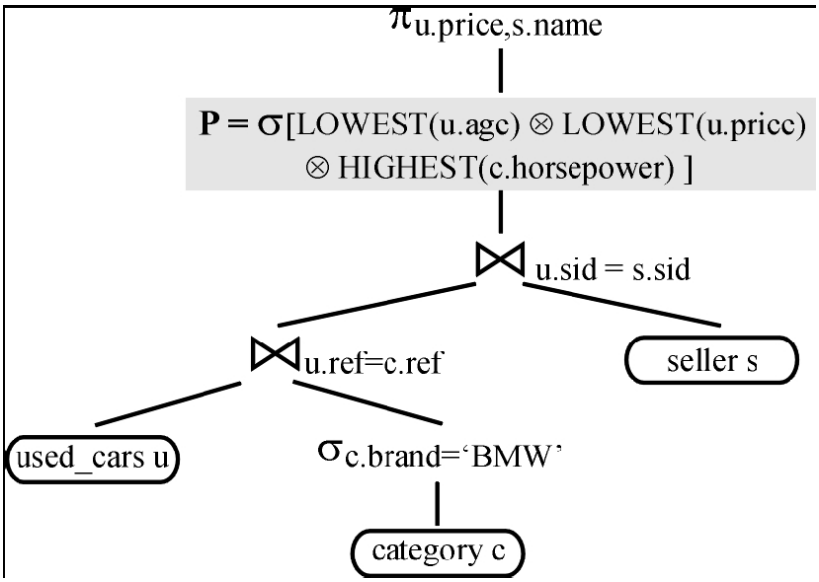
FROM used_cars u, category c, seller s

WHERE u.sid = s.sid AND u.ref = c.ref AND

c.brand = 'BMW'

PREFERRING LOWEST u.age AND

LOWEST u.price AND HIGHEST c.horsepower;



Hình 7

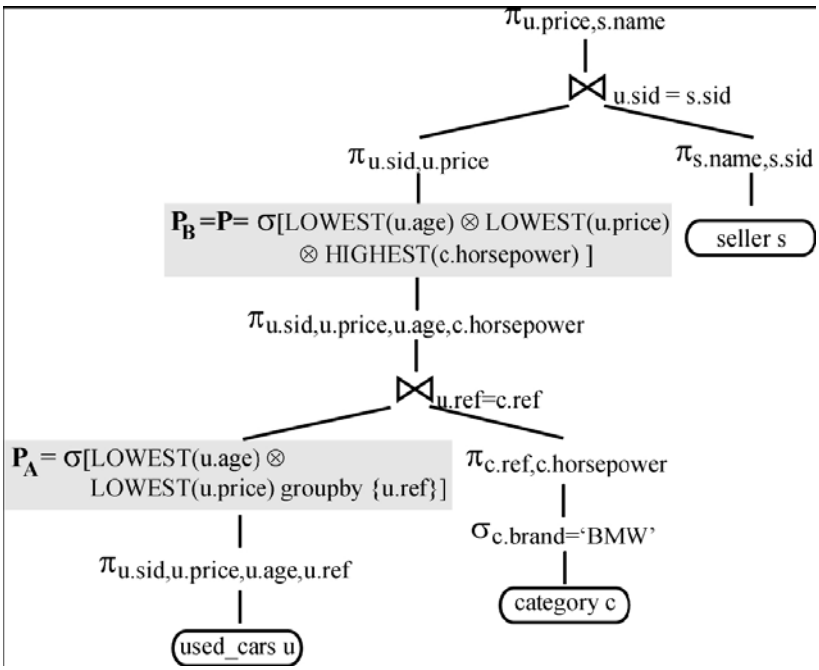
Cây T_{rel} đầu ra bởi Pass-1 là được mô tả như trong hình 7.

Cho Trel-2 thực hiện biến đổi ưa thích để đi đến cây cuối cùng T_{end} như trong hình 8: L6a; L7; L7; L9e; L1b; L1a

So sánh giữa các trường trong T_{end} và $T_{rel.}$

- *Chi phí phép kết nối*: Các đối số giống nhau như là sử dụng ở trên.

Chi phí ưa thích: Ban đầu skyline P đã từng được dựng lên và chia cắt thành hai skylines P_A và P_B = P.



Hình 8. T_{end} sau bước 2.

2.4.4 Tối ưu thứ tự phép kết nối

Một vấn đề chính trong tối ưu bài toán leo đồi mở rộng là thực hiện phép toán kết nối được định trước bởi đối số khởi tạo của các quan hệ cơ sở. Thực hiện phép toán kết nối có một tác động rất lớn trên ứng dụng của các luật ưa thích giống như L6. Do đó, phép toán kết nối cho phép các ứng dụng trước đây của toán hạng σ[P]. Trong ví dụ tiếp theo, chúng tôi sẽ mở rộng tối ưu cơ bản bởi thuật toán bottom up, mà được tính toán như phép toán kết nối

Ví dụ 4: Truy vấn kết nối phức tạp Q2

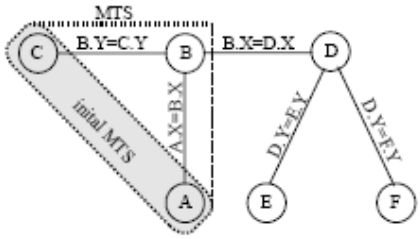
SELECT a.x, b.x

FROM a, b, c, d, e, f

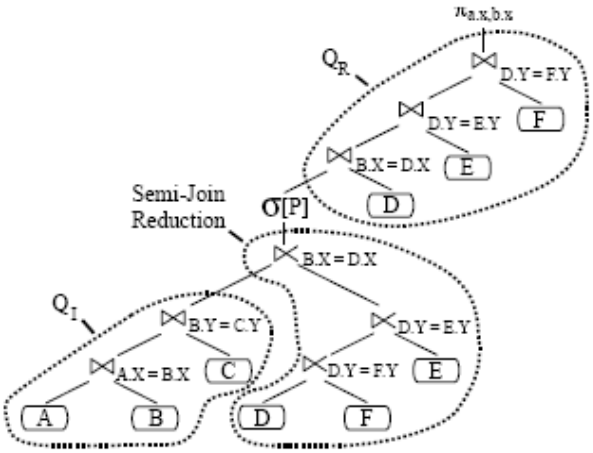
WHERE a.x=b.x and b.y=c.y and b.x=d.x and

d.y=e.y and d.y=f.y

PREFERRING a.val lowset and c.val highest



Với khởi tạo đầu vào, thuật toán nhận được đồ thị truy vấn Q=(V,E) những đỉnh của nó là những quan hệ của truy vấn và những đỉnh là bao gồm bởi các điều kiện join. Như bước đầu tiên khởi tạo bảng tối thiểu là được tính toán. Điều này bao gồm tất cả các quan hệ mà những thuộc tính của nó là bị liên quan bởi lựa chọn ưa thích σ[P]. Tập các bảng tối thiểu bao gồm đồ thị con QMTS của Q với VMTS=MTS. Nếu đồ thị này là có mối liên hệ với nhau chúng ta sẽ đi đến bước tiếp theo. Các trường hợp khác, chúng tôi thêm vào những đỉnh mới từ Q vào QMTS cho đến khi đồ thị kết quả là có mối liên hệ với nhau. Điều này là rất quan trọng, với những đỉnh không kết nối sẽ cho kết quả trong một tích Đề cac. Sau bước này, đồ thị truy vấn Q là bị sụp đổ, nhờ đó đồ thị kết quả là một cây hoặc không. Trong trường hợp đầu tiên, Giới hạn thuật toán. Các trường hợp khác chúng ta phải khử vòng xẩy ra trong đồ thị. Điều này được thực hiện bởi những nút liền kề sụp đổ lặp lại của QP với QP.. Cho những đỉnh lựa chọn là được xem xét, với đường đi dựa trên đường vòng tròn trong Q. Đồ thị truy vấn kết quả là thế cho thử nghiệm cho các cây thành viên của nó. Nếu sự thử nghiệm này trở thành thực, giới hạn thuật toán và các trường hợp khác lặp đi lặp lại thu được..



Đồ thị truy vấn kết quả là một cây với gốc QP. Từ đồ thị truy vấn này, truy vấn mới là được xây dựng trong bốn bước. Bước thứ nhất, một truy vấn QP là được xây dựng. Bước thứ hai, một full-semi-join thu nhỏ cho Q là được ước tính mà nhận được QI là đầu vào. Full semi-join giảm bớt được đảm bảo, làm cho tất cả các phần tử kết quả có ít nhất một đối tác kết nối trong truy vấn kết quả, nó là phù hợp cho các ứng dụng của σ[P]. , toán tử σ[P] là được thêm vào câu truy vấn và bước cuối cùng duy trì các đỉnh của Q là được tham gia vào theo cách thích hợp với truy vấn hiện tại (QR). Semi-joins giới thiệu ở bước thứ 2, có thể tinh lọc sử dụng tồn tại hai khóa của ràng buộc khóa ngoại.

2.5 Ứng dụng thực tế.

Bây giờ chúng ta nghiên cứu mô hình ưa thích phù hợp với cơ sở dữ liệu thực tế.

2.5.1 Tích hợp vào SQL và XML.

Nguồn gốc nội dung trong luận văn này được trích dẫn từ [1]. Suy diễn đến gần với lập trình với ưa thích như là thứ tự một phần nghiêm ngặt. Lý thuyết nêu trong phần [1] có thể cung cấp dạng cơ bản tồn tại thực tế của mô hình lý thuyết và tương ứng các lý thuyết chính với gộp vòa trong cơ sở dữ liệu suy diễn. Sự gộp này nhìn chung là rất hũu ích của mô hình Herbrand cho thứ tự một phần tùy ứng. Quay trở lại, mô hình Herbrand, miêu tả chính xác mô hình truy vấn phù hợp, là trường hợp đặc biệt của mô hình gộp. Như là hệ quả sự gặp gỡ này điểm quyết định từ sự giới thiệu của chúng tôi của ước nguyện. Chúng ta có thể mở rộng ngôn ngữ truy vấn cơ sở dữ liệu dưới một mô hình chính xác, bao gồm cơ sở dữ liệu SQL đối tượng quan hệ và cơ sở dữ liệu XML, bởi sự ưa thích dùng mô hình truy vấn BMO.

- **SQL ưa thích.**

SQL ưa thích được giới thiệu đầu tiên vào năm 1999, và là trường hợp đầu tiên mở rộng của SQL, trong đó xem sự ưa thích như là thứ tự một phần nghiêm ngặt. Nó không được giới thiệu rộng rãi và chỉ dừng lại ở mức nghiên cứu. SQL ưa thích thực hiện thêm vào ứng dụng tích hợp bởi viết lại khéo léo của truy vấn SQL ưa

thích vào trong mã SQL92, làm cho nó có giá trị trên DB2, Oracle 8i và MS SQL Server. SQL ưa thích được dùng trong thương mại như việc tìm kiếm ưa thích cho các ứng dụng thương mại điện tử. Mô hình ưa thích thực hiện bao trùm tất cả cấu trúc ưa thích cơ sở trước đây, Tích lũy Pareto (‘AND’) và ưa thích lan truyền. Ưa thích có thể được dùng cho mệnh đề PREFERRING mới. Dưới đây là 2 ví dụ giải thích:

*SELECT * FROM car WHERE make = ‘Opel’*
PREFERRING(category = 'roadster' ELSE category <> 'passenger' AND
price AROUND 40000 AND HIGHEST(power))
CASCADE color = 'red' CASCADE LOWEST(mileage);
*SELECT * FROM trips*
PREFERRING start_date AROUND '2001/11/23' AND duration AROUND 14
BUT ONLY DISTANCE(start_date)<=2 AND DISTANCE(duration)<=2;

Hàm đặc trưng LEVEL và DISTANCE có thể giám sát các mức độ đặc trưng yêu cầu (mệnh đề BUT ONLY) và có thể được khai thác cho giải thích truy vấn. Kinh nghiệm cho thấy các cửa hàng lớn dùng SQL ưa thích, hiệu năng từ các yêu cầu của khách hàng mà điển hình kích cỡ kết quả của ưa thích Pareto dùng mô hình truy vấn BMO từ vài cho đến vài tá, nó sẽ cho kết quả chính xác trong tình huống shopping.

- **XPATH ưa thích.**

XPATH ưa thích là một ngôn ngữ truy vấn dùng cho xây dựng tìm kiến thông tin hướng người dùng trong môi trường XML. Nó thực hiện đầy đủ biểu diễn mô hình ưa thích. Ứng dụng đầu tiên chạy trên hệ thống cơ sở dữ liệu XML là phần mềm AG. XPATH chuẩn được mở rộng như sau: Sản xuất “LocationStep: axis nodetest predicate*” là được cập nhật như là “LocationStep: axis nodetest(predicate|preference)*”. Để phân định ranh giới giữa lựa chọn bắt buộc XPATH dùng biểu trưng ‘[’ và ‘]’. Cho các lựa chọn mềm, XPATH dùng biểu trưng ‘#[’ và ‘]#’. Dưới đây là 2 ví dụ, tích lũy Pareto là được viết như ‘AND’ và tích lũy ưu tiên được viết là ‘PRIOR TO’:

Q1: /CARS/CAR #[(@fuel_economy)highest and (@horsepower)highest]#

Q2: /CARS/CAR #[(@color)in("black", "white")prior to(@price)around 10000]#
#[(@mileage)lowest]#

XPATH ưa thích có thể được áp dụng trong các công nghệ then chốt XML khác giống như XSLT, Xpointer hoặc Xquery.

2.5.2 Mô hình truy vấn ranking.

Ràng buộc mềm được thực hiện bởi tích lũy số rank(F) là được dùng rộng rãi trong cơ sở dữ liệu và các ứng dụng xử lý thông tin.

- *Bộ máy truy vấn đa tích năng.*

Đặc trưng cơ bản dùng trong bộ máy truy vấn đa tính năng để phân cấp bậc các đối tượng tùy thuộc vào các tính năng công nghệ phức tạp, hỗ trợ cho việc truy vấn bởi nội dung là hình ảnh dựa trên màu sắc, chữ hoặc hình khối. Khi đó rank(F) xây dựng ưa thích chuỗi trong hầu hết các trường hợp. Ngữ nghĩa BMO- truy vấn sẽ trả lại chính xác đối tượng phù hợp nhất. Rõ ràng, đây là tập quá nhỏ cho việc lựa chọn từ tổng quát. Cho nhiều lựa chọn hơn, mô hình truy vấn ‘k-best’ là được sử dụng, trả lại k đối tượng với một k xác định. Trong BMO , số lượng này lấy ra một số đối tượng không lớn nhất. Có đề xuất SQL/MM cho truy vấn đa tính năng trong SQL. Các thuật toán giống như Quick-Combine có thể được dùng cho việc tăng tốc độ tính toán của rank(F) dùng phương pháp ‘k-best’.

- Bộ máy tìm kiến văn bản đầy đủ.

Một lĩnh vực áp dụng khác là tìm kiếm đầy đủ văn bản, từ khóa tìm kiếm có thể được hiểu như là sự ưa thích đặc biệt, mỗi giá trị điểm số xác định sự thích hợp của chúng. Kết hợp chức năng F cho rank(F) là điển hình cho phạm vi nhỏ sản xuất dùng hàm cosin. Nếu mô hình không gian vector từ thông tin có được là dùng SQL đã từng được mở rộng bởi hệ quản trị cơ sở dữ liệu Orcale 8i, DB2 hoặc cơ sở dữ liệu văn bản (Informix), thực thi mô hình truy vấn k-best. Dạng XXL là được biểu diễn của cung cấp ngữ nghĩa k-best trong ngữ cảnh XML.

Sử dụng một số từ khóa đưa ra của không số và xếp hạng số hoặc tìm kiếm cơ bản. Tìm kiếm đầy đủ văn bản” Nếu thực hiện hiệu qủa của mô hình ưa thích đầy

đủ là thực thi, sau đó có nhiều lựa chọn hơn cho mô hình ưa thích trong ứng dụng, phạm vi từ không số thuần túy tới số thuần túy và bất cú sự kết hợp giữa chúng. Cho ví dụ, một kết hợp của tìm kiếm thuộc tính và tìm kiếm đầy đủ văn bản sẽ được tổng hợp của ưa thích XPATH và XXL. Tương tự như vậy kết hợp SQL ưa thích tốt với SQL và văn bản xếp hạng. Văn bản xếp hạng chính nó có thể là một tính năng trong truy vấn đa tính năng. Lý thuyết cho thấy nó có thể là trường hợp mà tính tổng xếp hạng số tất cả các ưa thích khác. Tuy nhiên, nó là sự ưa thích được hỗ trợ phần lớn của cấu trúc ưa thích: Xác định như là nhiều cấu trúc ưa thích có thể thường xuyên xẩy ra trong thế giới thực, có một ngữ nghĩa trực quan và các thuật toán đánh giá hiệu quả.

2.6. Tổng kết chương.

Chúng ta đã tìm hiểu nền tảng cơ bản cho xử lý và tối ưu truy vấn ưa thích và cũng thấy nó như là một mở rộng của công nghệ tối ưu truy vấn cho cơ sở dữ liệu quan hệ. Các truy vấn ưa thích có thể được phân tích bởi đại số quan hệ ưa thích, chương này cũng cung cấp nhiều luật biến đổi cho đại số quan hệ ưa thích mà là vấn đề then chốt cho tối ưu đại số. Tối ưu truy vấn ưa thích có thể được xây dựng như là một mở rộng của tối ưu SQL hiện tại, thêm vào một số suy diễn ngữ nghĩa. Bằng cách này chúng ta kế thừa được đầy đủ từ tối ưu truy vấn quan hệ.

Tóm lại, chúng ta đã nêu ra được những tính chất cốt yếu trợ giúp cho vấn đề tối ưu truy vấn ưa thích. Chúng ta cũng khẳng định rằng công nghệ cơ sở dữ liệu có thể một phần nào hỗ trợ tốt cho ưa thích và cá nhân hóa.

Chương III. SQL HƯỚNG NGƯỜI DÙNG.

Các bộ máy tìm kiếm hiện tại thường gặp phải nhiều vấn đề khó khăn khi thực hiện câu truy vấn hướng người dùng. Vấn đề khó khăn lớn nhất của bộ máy tìm kiếm hiện tại thực hiện với SQL chuẩn là SQL không hiểu được ý muốn rõ ràng về sự ưa thích của người dùng. SQL ưa thích là mở rộng của SQL chuẩn bởi mô hình ưa thích dựa trên thứ tự một phần nghiêm ngặt, Với những cấu truy vấn ưa thích xử lý giống như những ràng buộc lựa chọn mềm. Sự đa dạng của các loại truy vấn hướng người dùng cơ bản và tích lũy Pareto xây dựng nên ưa thích phức tạp, kết hợp với sự tham gia của các ngôn ngữ SQL hiện có, nó sẽ giúp cho ngôn ngữ lập trình ngày càng hiệu quả và thân thiện với người dùng. Tối ưu SQL ưa thích hiện tại thực hiện bằng SQL chuẩn, bao gồm thực hiện ở mức cao của thao tác skyline cho tập Pareto tối ưu. Đây là tiền xử lý tiến lại gần khả năng tích hợp ứng dụng không có dấu hiệu nối thêm, là cho SQL ưa thích tương tích với nền SQL bao gồm IBM DB2, Oracle, Microsoft SQL Server và Sybase. Lợi ích của ông nghệ SQL ưa thích bao gồm truy vấn có sự tương tác với người dùng cao, phù hợp với thương mại điện tử và thời gian phát triển ngắn cho các ứng dụng e-service.

3.1. Thiết kế ngôn ngữ SQL hướng người dùng.

Các bộ máy tìm kiếm trực tiếp thực hiện với SQL chuẩn với thực tế rằng SQL đó không ngay lập tức hiểu khái niệm về sự ưa thích, do đó không đủ khả năng để hỗ trợ cho các ràng buộc mềm. Do đó, bất cứ truy vấn ưa thích nào phải được dịch ra dạng chuẩn trong mệnh đề WHERE, thực tế, người dùng thường khó khăn khi nghĩ giống như cơ sở dữ liệu SQL. Cần có một sự mở rộng cần thiết của SQL chuẩn thêm vào ưa thích như là cấu trúc ngôn ngữ cơ sở. Vấn đề nan giải để cố gắng là lựa chọn mô hình thích hợp của ưa thích. Một mô hình phải thỏa mãn về trực quan cho người dùng và phải phù hợp với dạng ngữ nghĩa mà cho phép tích hợp hiệu quả vào SQL chuẩn. Từ đó chúng ta sẽ giới thiệu cơ sở của mô hình ưa thích dựa trên thứ tự một phần nghiêm ngặt.

3.1.1 Một mô hình cho sự ưa thích.

Ưa thích có ít nhất hai khía cạnh khác biệt nhau, cả hai có thể biểu diễn các tình huống trong thế giới thực:

1. Lựa chọn trong thế giới chính xác: Tất cả con người đều muốn ước muốn của họ trở thành hiện thực hoặc thoả mãn một phần nào đó. Thông thường sự lựa chọn của con người là cố gắng xác định rõ sự mục tiêu mong muốn. Một ví dụ cụ thể là cấu hình phần mềm tùy thuộc vào thông tin người sử dụng. Trong giới hạn về công nghệ, các truy vấn trong ngữ cảnh này là những truy vấn chính xác với những lựa chọn bắt buộc. Thể hiện chính xác đối tượng mong muốn nếu nó là tồn tại thực và những trường hợp khác sẽ từ chối yêu cầu của người dùng.
2. Lựa chọn trong thế thế giới thực.

   Các lựa chọn có cách cư xử khá khác nhau. Chúng được thực hiện bởi ưa thích cá nhân trong những ước muốn: Những ước muốn là tự nhiên, nhưng không có sự ràng buộc mà họ có thể thỏa mãn mọi thời điểm. Trong trường hợp lỗi xẩy ra cho người dùng là không thường xuyên, nhưng thường phải chấp nhận những kết quả xấu hoặc giàn xếp sự thỏa hiệp. Do đó, sự ưa thích trong thế giới thực yêu cầu thay đổi mô hình từ sự phù hợp chính xác tới phù hợp thực tiến, có nghĩa là tìm vị trí phù hợp nhất giữa ước muốn và thực tế. Hay nói cách khác, sự ưa thích ở đây sẽ chú ý nhiều đến ràng buộc mềm.

Do đó, thành công khi xây dựng chương trình tìm kiếm hướng người dùng yêu cầu như sau:

1. Các ràng buộc bắt buộc trong thế giới thực chính xác: Đây là một điểm mạnh của SQL chuẩn.
2. Ưa thích chọn lựa trong thế giới thực: Cho những mô hình không phù hợp với sự chính xác trong thế giới thực tại ngày nay.

Chúng ta tập trung vào xử lý các thiếu sót trong SQL chuẩn bằng SQL ưa thích. Đó là vấn đề cơ bản để đưa ra mô hình phù hợp của sự ưa thích. Sự ưa thích trong thế giới thực là khác với sự hiểu biết của mọi người về nó. Tuy nhiên, kiểm tra cẩn

thận về sự thể hiện tự nhiên khác nhau mà chúng chia sẻ nền tảng nguồn gốc phổ biến, Hãy xem cuộc sống hàng ngày của chúng ta với sự phong phú của sự ưa thích mà có thể đến từ sự cảm nhận trong đầu hoặc sự tác động của trực quan khác. Trong sư tương tự cài đặt nó ngoài sự kiểm soát mà con người biểu diễn những ước muốn thường xuyên trong điều kiện của "tôi thích A hơn B". Bằng cách này con người diễn tả một xếp hạng không số giữa A và B. Đây là một loại mô hình ưa thích mà được sử dụng rộng rãi và trực quan dễ hiểu đối với con người. Thực tế, mỗi đứa trẻ học hiểu biết về thế giới xung quanh ngay khi chúng còn rất trẻ. Con người qua trực giác thường xuyên thể hiện sự ưa thích của mình, điển hình với nó là không diễn tả trong điều kiện của điểm số và tính toán. Nghĩ về sự ưa thích trong điều kiện "tốt hơn" có một bản sao tự nhiên trong toán học: Chúng ta có thể ánh xạ chúng vào cuộc sống thực một cách trung thực dựa trên thứ tự một phần nghiêm ngặt. Toán học biểu diễn sự ưa thích P=(A,<P) là không thực sự mềm dẻo.

3.1.2 Tổng quan về ngôn ngữ SQL ưa thích.

Con người thường xuyên mong muốn diễn tả sự ước muốn của họ theo cách diễn tả đầy đủ. Họ đơn giản không muốn gặp rắc rối với kỹ thuật chi tiết là diễn tả ước muốn của họ như thế nào là đúng. Điều ước muốn thật sự có được thoả mãn. SQL ưa thích mang lại sự thuận tiện này cho người dùng. Bây giờ, chúng ta giới thiệu những tính năng then chốt của ngôn ngữ truy vấn SQL ưa thích. SQL ưa thích là một sự mở rộng của SQL chuẩn. Nó được giới thiệu như là cấu trúc ngôn ngữ mới, trong đó xem sự ưa thích như là vấn đề cốt lõi.

SQL ưa thích = SQL chuẩn + Sự ưa thích

Sự ưa thích có thể được xây dựng trên truy vấn cơ sở, hoặc chúng có thể được định nghĩa như là những đối tượng sử dụng ngôn ngữ định nghĩa ưa thích. Sự ưa thích là diễn tả cú pháp bên trong một khối truy vấn SQL, sử dụng từ khóa PREFERRING.

3.1.2.1 Xây dựng các loại ưa thích.

Có khá nhiều tiêu chuẩn lựa chọn khác nhau mà dùng cho sự ưa thích trong thứ tự một phần nghiêm ngặt. Mục đính của SQL ưa thích cung cấp nhiều loại ưa thích cơ bản, điển hình dùng cho xây dựng các bộ máy tìm kiếm.

- Phép tính xấp xỉ: AROUND, BETWEEN

Sự ưa thích 'AROUND' ý muốn nói giá trị đến gần với giá trị mục tiêu số. Được sử dụng khi nhắm vào giá trị mục tiêu không phải là băt buộc hoặc vị trí khó khăn. Truy vấn sau đây trả về chuyến hành trình có thời gian khoảng 14 ngày. Hoặc chính xác 14 ngày.

*SELECT * FROM trips PREFERRING duration AROUND 14;*

BETWEEN[low, up] là loại ưa thích có cách thể hiện tương tự. Giá trị bên trong [low, up] là thoản mãn, ngoài ra là bị loại bỏ.

- Nhỏ nhất/Lớn nhất: LOWEST, HIGHEST

Ưa thích thường xuyên xẩy ra là hỏi về giá trị cao nhất và thấp nhất, nếu là đúng, các trường hợp khác trả vè giá trị gần với giá trị lớn nhất hoặc nhỏ nhất là được chấp nhận. Câu truy vấn ưa thích sau cho giá trị căn hộ lớn nhất.

*SELECT * FROM apartments PREFERRING HIGHEST(area);*

Truy vấn này có một bản sao SQL chuẩn. Nhưng chú ý là thay vì thuộc tính đơn (giống như area) lại có một diễn tả qua nhiều thuộc tính hoặc các sự kiện thông qua stored procedure.

- Ưa thích, không thích: POS, NEG

Ưa thích POS diễn tả một điều kiện mềm mà giá trị mong muốn nên là một thay vì một danh sách các giá trị. Truy vấn sau đây tìm kiếm một người lập trình thành thạo Java hoặc C++. Nếu không làm việc với hai ngôn ngữ trên, người lập trình sẽ bỏ qua và thay bằng điều kiện khác.

*SELECT * FROM programmers PREFERRING exp IN ('java', 'C++');*

"Không nên có" tiêu chuẩn được hỗ trợ bởi ưa thích NEG. Truy vấn sau đây diễn tả sự ưa thích cho một khách sạn ngoài thành phố. Nếu chỉ những khách sạn trong thành phố có phòng, đề nghị tìm khách sạn có phòng hơn là không có gì.

*SELECT * FROM hotels PREFERRING location <> 'downtown';*

3.1.2.2 Tập hợp các ưa thích phức hợp.

Nhìn chung, sự quyết định không dựa trên sự ưa thích đơn lẻ, nhưng dựa trên sự kết hợp phức hợp có thể của ưa thích. SQL ưa thích đề nghị có những ưa thích phức hợp.

- **Quan trọng ngang nhau:** Tích lũy Pareto (AND)

Tích lũy Pareto của ưa thích P1, ... , Pn trong ưa thích phứp hợp P là được định nghĩa như:

v = (v1, …, vn) là tốt hơn w = (w1, …, wn) nếu

vi là tốt hơn wi và v là cân bằng hoặc tốt hơn w trong bất kỳ giá trị thành phần khác

Ngữ nghĩa trực quan của tích lũy Pareto là một sự kết hợp không phân biệt đối xử của ưa thích quan trọng tương đương nhau P1, ... , Pn. Tích lũy Pareto tạo thành một thứ tự một phần nghiêm ngặt trở lại, sau đó là một sự ưa thích trong sự cảm nhận. Giá trị lớn nhất của P được gọi là tập Pareto tối ưu. Nguồn gốc của Pareto tối ưu đã từng được sử dụng và nghiên cứu ít nhất cách đây khoảng 50 năm cho những vấn đề ra quyết định đa thuộc tính trong xã hội và khoa học kinh tế. Cú phps của SQL ưa thích cho tích lũy Pareto là phép ‘AND’ của ưa thích cơ bản.

Hãy tưởng tượng, khi mua một máy tính thì khách hàng thường quan tâm đến kích cỡ bộ nhớ lớn nhât và tốc độ CPU là quan trọng ngang nhau. Sự ưa thích này có thể được diễn tả như sau:

*SELECT * FROM computers*
PREFERRING HIGHEST(main_memory) AND HIGHEST(cpu_speed);

- **Thứ tự quan trọng**:Lan truyền của sự ưa thích (CASCADE)

Cascading của sự ưa thích phân ra các mức khác nhau của sự quan trọng cho ưa thích hợp thành, sử dụng ưa thích một lần sau cái khác. Cú pháp SQL ưa thích cho thứ tự quan trọng là ‘CASCADE’.

Giả sử là một số người muốn mua một máy tính, màu của nó nên là màu đen hoặc nâu, thuộc tính này xem là ít quan trọng hơn kích cỡ lớn nhất của bộ nhớ chính. Điều mong này có thể được diễn tả như sau:

*SELECT * FROM computers*

PREFERRING HIGHEST(main_memory) CASCADE color IN ('black','brown');

- **Kết hợp tích lũy Pareto và cascading.**

Sức mạnh của SQL ưa thích khi kết hợp những cấu trúc cơ bản của nó thành ước muốn phức hợp. Cho một ví dụ cụ thể diễn đạt những ước muốn của khách hàng bằng ngôn ngữ tự nhiên:

"Loại xe ưa thích của tôi phải là loại Opel. Nó nên là một xe không mui 2 chỗ ngồi, nhưng nếu không có, không nên có là xe khách. Sự quan trọng ngang nhau tôi muốn chi khoảng 40000DM và xe nên có động cơ mạnh. Sự ít quan trọng hơn tôi thích xe màu đỏ. Nếu có thêm nhiều lựa chọn, chọn xe ít được đi".

Các tính năng truy vấn SQL ưa thích bao gồm điều kiện cứng và mềm. Kết hợp ưa thích POS/NEG để lọc loại xe, ưa thích AROUND để chọn giá cả, ưa thích POS cho màu xe và LOWEST cho số km đã sử dụng. Ta có câu truy vấn sau:

*SELECT * FROM car WHERE make = 'Opel'*

PREFERRING (category = 'roadster' ELSE category <> 'passenger' AND

price AROUND 40000 AND HIGHEST(power))

CASCADE color = 'red' CASCADE LOWEST(mileage);

3.1.2.3 Giải thích câu trả lời.

Khi một phần từ được lựa chọn hoặc từ chối bởi điều kiện WHERE trong SQL chuẩn, lý do là hiển nhiên ngay lập tức từ những thuộc tính của phần tử: Nếu chúng gặp điều kiện nó thuộc vào kết quả, các trường hợp khác sẽ không. Biểu diễn của phần tử trong tập kết quả của truy vấn SQL ưa thích không chỉ tùy thuộc vào chất lượng của chính bản thân nó., nhưng cũng vì sự so sánh. Sự tiến triển này cần sự điều chỉnh kết quả của truy vấn ưa thích, giống như trong thế giới thực.

Cho mục đích SQL ưa thích hỗ trợ về các hàm chất lượng, báo cáo với tiêu chuân mềm là được cho kết quả với mở rộng:

- TOP(A) là một báo cáo dự đoán Boolean với giá trị thuộc tính A là một giá trị phù hợp hoặc không.
- LEVEL(A) thông báo giá trị A trả về từ truy vấn là một phần từ giá trị A lớn nhất .
- DISTANCE (A) thông báo cho biết giá trị số A của kết quả trả về là một phần từ giá trị A lớn nhất .

Nghiên cứu cơ sở dữ liệu Car và truy vấn SQL ưa thích. Kết hợp Pareto một ưa thích POS/POS với ưa thích AROUND.

oldtimer(	ident,	color,	age)
	Maggie	white	19
.	Bart	green	19
.	Homer	yellow	35
.	Selma	red	40
.	Smithers	red	43
.	Skinner	yellow	51

SELECT ident, color, age, LEVEL(color), DISTANCE(age)
FROM oldtimer
PREFERRING color = ‘white‘ else color = ‘yellow‘ AND age AROUND 40;

Tối ưu Pareto cho kết quả là:

Selma	red	40	3	0
Homer	yellow	35	2	5
Maggi	white	19	1	21

Do đó bạn có thể nhìn thoáng qua thấy tiêu chuẩn là được thấy như kết quả và có bao nhiêu sự khác biệt từ điều kiện thuận lợi nhất. Đây là một điểm quan trọng cải

tiến qua hành vi không thỏa mãn của nhiều bộ máy tìm kiếm mà kết quả xếp hạng bởi điểm số không cho bất cứ thông tin nào mà có tính điểm.

3.1.2.4 Điều khiển đặc trưng.

Nếu kết quả trả về từ truy vấn ưa thích là không được chấp nhận, cần phải thực hiện xử lý để tìm ra câu trả lời tốt nhất. Các hàm đặc trưng SQL ưa thích là rất hữu ích, nếu người dùng muốn có hiệu lực chắc chắn tiêu chuẩn chất lượng nhỏ nhất một kết quả phải thỏa mãn.Giả sử là một khách hàng trên mạng đang tìm kiếm một chuyến du lịch bắt đầu khởi hành vào tháng 7 ngày 3 và đi khoảng 2 tuần. Nhưng họ không chấp nhận sự thay đổi trên 2 ngày cho mỗi tiêu chuẩn. Sự giới hạn có thể được diễn tả dùng 'BUT ONLY' của SQL ưa thích:

*SELECT * FROM trips*
PREFERRING start_day AROUND '1999/7/3' AND duration AROUND 14
BUT ONLY DISTANCE(start_day)<=2 AND DISTANCE(duration)<=2;

Rõ ràng, kết quả rỗng có thể xẩy ra, nhưng sự tương quan này với sự rõ ràng của người dùng được tăng cao.

3.1.2.5 Khối truy vấn SQL ưa thích.

Cú pháp hoàn chỉnh của SQL ưa thích phiên bản 1.3 hiện tại là được cho trong. Nhìn chung, truy vấn SQL ưa thích đề nghị những lựa chọn sau, cho phép các lựa chọn cứng và mềm cùng tồn tại trong một câu truy vấn đơn

SELECT <selection>
FROM <table_references>
WHERE <where-conditions>
PREFERRING <soft-conditions>
GROUPING <attribute_list>
BUT ONLY <but_only_condition>
ORDER BY <attribute_list>

Các phần tử là mở rộng cuả SQL chuẩn là xuất hiện trong mệnh đề <selection> . Giống như SQL chuẩn, WHERE và mệnh đề ORDER BY là được lựa chọn. Nếu không có mệnh đề PREFERRING, nó không phải là truy vấn ưa thích. Mệnh đề GROUPING và BUT ONLY là tùy chọn. Sự ưa thích chỉ dùng cho những phần tử đáp ứng điều kiện WHERE. Điều kiện của mệnh đề BUT ONLY là hợp lý được kiểm tra sau khi dùng ưa thích của mệnh đề PREFERRING. Truy vấn SQL ưa thích có thể được cầu khẩn như là truy vấn phụ của trạng thái INSERT. Như là hạn chế hiện tại truy vấn phụ trong mệnh đề WHERE có thể không chứa mệnh đề PREFERRING. Ngữ nghĩa trả lời cùa SQL ưa thích đi theo một mô hình truy vấn BMO.

Tìm tất cả các giá trị phù hợp, ưa thích P trong mệnh đề PREFERRING. Nếu tập giá trị là không rỗng, chúng ta thực hiện được.

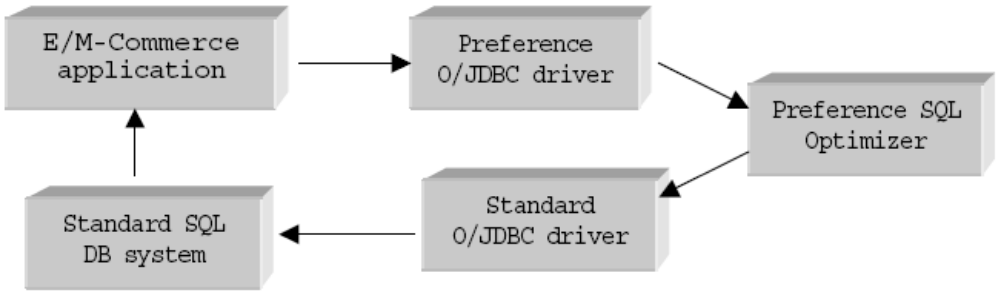
Các trường hợp khác, xem xét đến tất cả các giá trị khác ở trong ngưỡng chất lượng BUT ONLY, nhưng .

3.2. Môi trường thực thi của SQL ưa thích.

SQL ưa thích đã từng được thiết kế và thử nghiệm bắt đầu từ năm 1997 và nó dùng trong sản phẩm thương mại từ năm 1999.

3.2.1 Tích hợp vào các ứng dụng sẵn có.

SQL ưa thích thực hiện như là một lớp thực thi giữa ứng dụng và cơ sở dữ liệu. Nó xử lý các truy vấn ưa thích bởi dịch chúng thành các truy vấn SQL chuẩn và đưa chúng vào cơ sở dữ liệu. Các truy vấn không dùng ưa thích là được đưa vào hệ thống cơ sở dữ liệu mà không cần phải xử lý. Các ứng dụng SQL kế thừa chạy mà không bị bất cứ sự hạn chế nào.



Trong thi hành hiện tại của SQL ưa thích 1.3 một điều khiển Preference ODBC hoặc JDBC là được đặt trực tiếp đằng trước của mô đun tối ưu SQL ưa thích để dịch Preference SQL thành SQL chuẩn, tói ưu trước là chức năng tương đối mới lạ. Tiêu điều khiển ODBC hoặc JDBC chuẩn thực hiện truyền chương trình SQL vào hệ thống cơ sở dữ liệu SQL, ở đó nó được thực hiện tối ưu lần thứ 2 bởi tối ưu SQL chuẩn. Chương trình thực hiện là chạy trở lại cơ sở dữ liệu SQL tồn tại. Bất cứ đoạn mã thêm vào, nhìn chung cho truy vấn viết lại bởi tối ưu SQL ưa thích, đầy đủ dùng trong SQL92. Do đó SQL ưa thích có thể chạy kết hợp với bất cứ hệ thống cơ sở dữ liệu dùng SQL 92.

3.2.2 Tối ưu SQL ưa thích.

SQL ưa thích phức tạp có thể được trình bày rõ ràng nêu ra trong truy vấn ưa thích đơn giản. Nó trở thành vấn đề chính của tối ưu SQL ưa thích để tìm ra câu trả lời đúng sử dụng mô hình truy vấn BMO.

Các phương thức khác nhau gần đây đã từng được đề nghị cho việc thực hiện thao tác skyline. Trong phạm vi luận văn này chúng ta chỉ nêu ra sự hiệu quả tính toán của tối ưu pareto trong được viết lại cho phù hợp, sự tối ưu cũng phải tính đến hệ thống SQL hiện có. Thuật toán trừu tượng được trình bày như sau:

Lựa chọn thuật toán cho lấy lại tất cả các phần tử từ quan hệ R, thứ tự một phần nghiêm ngặt P:

(1) Khởi đầu cho tập phần tử lớn nhất Max là rỗng.

(2) Lựa chọn một phần tử t1 từ R.

(3) Chèn t1 vào Max nếu không có phần tử t2 trong R mà là tốt hơn t1.

(4) Lặp lại các bước (2) và (3) cho tới khi tất cả phần tử t1 từ R đã từng được lựa chọn.

(5) Thuật toán kết thúc. Max chứa các phần tử lớn nhất từ R. Thuật toán kết thúc.

Chúng tôi đưa ra truy vấn SQL ưa thích này ngược với quan hệ Car tiếp sau:

SELECT * FROM Cars PREFERRING Make = ‘Audi’ AND Diesel = ‘yes’;

Identifier	Make	Model	Price	Mileage	AirBag	Diesel
1	Audi	A6	40000	15000	Yes	No
2	BMW	Series	35000	30000	Yes	Yes
3	Volkswagen	Beetle	20000	10000	Yes	no

Thuật toán có thể được cài đặt bởi SQL92, sau khi tạo ra một quan hệ tạm thời Max:

```
CREATE VIEW Aux AS
SELECT *, CASE WHEN Make = 'Audi' THEN 1 ELSE 2 END AS Makelevel,
CASE WHEN Diesel = 'yes' THEN 1 ELSE 2 END AS Diesellevel
FROM Cars;
INSERT INTO Max
SELECT Identifier, Make, Model, Price, Mileage, Airbag, Diesel
FROM Aux A1
WHERE NOT EXISTS (SELECT 1 FROM Aux A2
WHERE A2.Makelevel <= A1.Makelevel AND
A2.Diesellevel <= A1.Diesellevel AND
(A2.Makelevel < A1.Makelevel OR
A2.Diesellevel < A1.Diesellevel));
```

Tối ưu câu truy vấn hiện tại có thể được thực hiện bằng ngôn ngữ SQL này với câu truy vấn phụ tương đương NOT EXISTS.

3.3 Tổng kết chương.

Chúng ta đã nghiên cứu một cách tổng quát về SQL ưa thích, và thấy nó là mở rộng của SQL chuẩn với ưa thích dựa trên ngữ nghĩa thứ tự một phần chặt. Các thuộc tính của SQL ưa thích bao gồm các toán tử ưa thích và đã được xây dựng sẵn. Trong chương này cũng đã đưa ra một số phương pháp tối ưu hóa SQL ưa thích. Hy vọng rằng SQL ưa thích sẽ mang lại nhiều lợi ích cho các nhà phát triển ứng dụng và góp phần làm cho các phần mềm ứng dụng trở nên thân thiện với người dùng.

KẾT LUẬN VÀ ĐỊNH HƯỚNG TƯƠNG LAI.

Truy vấn ưa thích là một lựa chọn tốt cho nhiều ứng dụng cơ sở dữ liệu. Với nhiều tiết mục phong phú của cấu trúc ưa thích trực quan có thể làm cho thuận tiện với thiết kế của các ứng dụng hướng người dùng. Khi đó các ưa thích số là bị giới hạn bởi tính diễn cảm, cấu trúc cho ưa thích pareto và ưu tiên là được yêu cầu. Một trong những tranh cãi là mô hình đưa ra có thời gian thực hiện thiếu khả thi không. Tuy nhiên, những đóng góp của mô hình đưa ra cho một bằng chứng mạnh mẽ là truy vấn ưa thích quan hệ dùng mô hình BMO là nó không phải là thừa.

Chúng ta đã từng sắp đặt nền tảng của bộ khung cho tối ưu truy vấn ưa thích là một sự mở rộng được thiết lập cho công nghệ tối ưu truy vấn từ cơ sở dữ liệu quan hệ. Truy vấn ưa thích có thể được đánh giá bởi đại số quan hệ ưa thích, mở rộng của đại số quan hệ cổ điển. Chúng ta đã từng cung cấp một số các luật biến đổi cho đại số quan hệ ưa thích mà là những nhân tố then chốt cho tối ưu đại số. Tối ưu truy vấn ưa thích có thể được xây dựng như là một mở rộng của tối ưu SQL tồn tại, thêm vào một chút suy luận dựa trên kinh nghiệm giống như ‘ưa thích thúc đẩy ’. Trong trường hợp này một thập kỷ đầu tư và nghiên cứu với tối ưu truy vấn ưa thích có thể được kế thừa đầy đủ. Chúng ta biểu diễn một mẫu đơn giản về tối ưu truy vấn ưa thích và hiệu năng của nó. Thử nghiệm cài đặt trên SQL thực tại thì hiệu năng đạt được là khả thi.

Nghiên cứu mở ra cho nhiều ứng dụng tính thân thiện cao với người sử dụng, và cũng dễ dàng cài đặt bằng SQL ưa thích hoặc XPATH ưa thích. Xây dựng dựa trên nền tảng sáng suốt phát triển cho tối ưu truy vấn ưa thích. Một vấn đề quan trọng đưa ra của tối ưu giá trị cơ sở có thể được giải quyết trong thời gian tới. Chúng ta cũng sẽ mở rộng nghiên cứu tối ưu cho truy vấn ưa thích đệ quy, lấy cơ sở dựa trên kết quả nghiên cứu ưa thích pushing. Tóm lại chúng ta tin rằng chúng ta có đóng góp những bước cơ bản để cho những nghiên cứu về tối ưu truy vấn ưa thích ngày càng tốt hơn. Công nghệ cơ sở dữ liệu có thể hỗ trợ tốt cho ưa thích và cá nhân hóa, cả hai như là điều kiện không bị ràng buộc cho mô hình và hiệu quả cao.

Trong phạm vi luận văn này, chúng ta đã nghiên cứu SQL ưa thích, nó tương thích mở rộng của SQL chuẩn với ưa thích dựa trên ngữ nghĩa thứ tự một phần chặt. Các tính năng quan trọng bao gồm sự đa dạng của các toán tử ưa thích được xây dựng sẵn. Cấu trúc tích lũy Pareto được lắp gép thành ưa thích phức hợp và hàm điều khiển chất lượng Chúng ta đã đưa ra một số hiểu biết sâu sắc về tối ưu hóa SQL ưa thích và cũng thử nghiệm cho thấy SQL ưa thích cho hiệu năng cao. SQL ưa thích là một trường hợp của nghiên cứu của tìm kiếm toàn diện nỗ lực trong sự phát triển.

TÀI LIỆU THAM KHẢO.

Tiếng Anh

1. Werner Kießling. *Foundations of Preferences in Database Systems*
2. Bernd Hafenrichter, Werner Kießling. *Optimization of Relational Preference Queries*
3. Jan Chomicki. *Semantic Optimization Techniques for Preference Queries*.
4. Jan Chomicki . *Preference Formulas in Relational Queries*.
5. Werner Kießling, Bernd Hafenrichter. *Algebraic Optimization of Relational Preference Queries*
6. Werner Kießling, Gerhard Kostler. *Preference SQL - Design, Implementation, Experiences*

PHỤ LỤC

Tất cả các chứng minh dùng định nghĩa cho ưa thích P = (A, <P), ưa thích chọn lọc σ[P](R) và ưa thích chọn lọc nhóm σ[P groupby B](R) .

Bổ đề 1: Cho P = (A, <P) và A □ X □ attr(R). Sau đó P xác định như là thứ tự một phần nghiêm ngặt <X onto R: v, w ∈ R: v <X w □ v[A] <P w[A]

Bổ đề 2:Cho f(r, s) là tổng trái hàm Boolean, cho mỗi r có tồn tại ít nhất một giá trị s thỏa mãn f(r, s). Mở rộng của R bởi S qua f là được định nghĩa như:

R extf S = {(r, s) | r∈R, s∈S, f(r, s)}

Cho P = (A, <P) và A □ attr(R), khi đó:

σ [P](R extf S) = σ[P](R) extf S

Chứng minh:σ [P](R extf S)

= {w∈ R extf S | ¬□v ∈ R extf S: w[A] <P v[A]}

= {w∈ R extf S | ¬□v ∈ R : w[A] <P v[A]}

= {w∈ R | ¬□v ∈ R : w[A] <P v[A]} extf S = σ[P](R) extf S

Bổ đề 3:Cho P1 = (A, <P1), P2 = (A, <P2) và σ[P1](R) □ σ[P2](R):

σ[P1](R) = σ[P1](σ[P2](R))

Chứng minh:σ[P1](R)

= σ[P1](R) ∩ σ[P2](R) = {w ∈ R | ¬□v ∈ R: w[A] <P1 v[A]} ∩ σ[P2](R)

// cho σ[P1](R) □ σ[P2](R) chúng ta có:

// ¬□v ∈ R: w[A] <P1 v[A] dẫn đến

// ¬□v ∈ σ[P2](R): w[A] <P1 v[A] = {w ∈ R | ¬□v ∈ σ[P2](R): w[A] <P1 v[A]} ∩ σ[P2](R) = {w ∈ σ[P2](R) | ¬□v ∈ σ[P2](R): w[A] <P1 v[A] } = σ[P1](σ[P2](R))

Bổ đề 4:σ[P1 □ P2](R) □ σ[P1 groupby A2](R)

Chứng minh : Let P = (A, <P) := (A1 ∪ A2, < P1 □ P2).σ[P1 □ P2](R)

= {w ∈ R | ¬□v ∈ R : w[A] <P v[A] }

// Def '□' = {w ∈ R | ¬(□v ∈ R: (w[A1] <P1 v[A1] ∧ (w[A2]

$= v[A2] \vee w[A2] <P2 v[A2])) \vee (w[A2] <P2 v[A2] \wedge (w[A1]$

$= v[A1] \vee w[A1] <P1 v[A1]))))\} = \{w \in R \mid \neg(\Box v \in R: (w[A1] <P1 v[A1] \wedge$

$w[A2]$

$= v[A2]) \vee (w[A1] <P1 v[A1] \wedge w[A2] <P2 v[A2]) \vee (w[A2] <P2 v[A2] \wedge$

$w[A1] = v[A1]))\}$

$= \{w \in R \mid \neg((\Box v \in R: (w[A1] <P1 v[A1] \wedge w[A2]$

$= v[A2])) \vee (\Box v \in R: (w[A1] <P1 v[A1] \wedge w[A2] <P2 v[A2])) \vee (\Box v \in R:$

$(w[A2] <P2 v[A2] \wedge w[A1] = v[A1]))\}$

$= \{w \in R \mid (\neg\Box v \in R: (w[A1] <P1 v[A1] \wedge w[A2]$

$= v[A2])) \wedge (\neg\Box v \in R: (w[A1] <P1 v[A1] \wedge w[A2] <P2 v[A2])) \wedge (\neg\Box v \in R:$

$(w[A2] <P2 v[A2] \wedge w[A1] = v[A1]))\}$

$= \sigma[P1 \text{ groupby } A2](R) \cap \sigma[P2 \text{ groupby } A1](R) \cap \{w \in R \mid \Box v \in R: (w[A1] <P1$

$v[A1] \wedge w[A2] <P2 v[A2])\} \Box \sigma[P1 \text{ groupby } A2](R)$

Bổ đề 5:Let P=(A,<P), attrib(P) $\Box$ R, X $\Box$ attribs (R) $\cap$ attirb(S)

khi đó $\sigma[P \text{ groupby } X](R \rhd\!\lhd_{R.X=S.X} S) \bowtie_{R.X=S.X} S$

$= \sigma[P \text{ groupby } X](R) \bowtie_{R.X=S.X} S$

Chứng minh:$T' = R \rhd\!\lhd_{R.X=S.X} S \sigma[P \text{ groupby } X](R \rhd\!\lhd_{R.X=S.X} S) = \{ w \in T' \mid \neg\Box v \in T' :$

$w < v \wedge w[x] = v[x] \}$

$= \{ w \in R \mid \neg\Box v \in R: w < v \wedge w[x] = v[x] \wedge w \in T' \wedge v \in T' \}$

// w[x]=v[x] , transitivity= $\{ w \in R \mid \neg\Box v : w < v \wedge w[x] = v[x] \wedge w \in T'\}$

$= \{ w \in R \mid \neg\Box v : w < v \wedge w[x] = v[x] \} \cap T'$

$= \sigma[P \text{ groupby } X](R) \cap R \rhd\!\lhd_{R.X=S.X} S$

$\sigma[P \text{ groupby } X](R \rhd\!\lhd_{R.X=S.X} S) \bowtie_{R.X=S.X} S$

$= (\sigma[P \text{ groupby } X](R) \cap R \bowtie_{R.X=S.X} S) \bowtie_{R.X=S.X} S$

$= (\sigma[P \text{ groupby } X](R) \bowtie_{R.X=S.X} S) \cap ((R \rhd\!\lhd_{R.X=S.X} S) \bowtie_{R.X=S.X} S)$

$= \sigma[P \text{ groupby } X](R) \bowtie_{R.X=S.X} S \cap (R \bowtie_{R.X=S.X} S)$

$= \sigma[P \text{ groupby } X](R) \bowtie_{R.X=S.X} S$

(A) Định lý cho P1&P2:

(a) P1&P2 ≡ P1 nếu P1 = (A, <P1) và P2 = (A, <P2)

(b) P1&P2 ≡ P1 + (A1↔&P2) nếu A1 ∩ A2 = □

Chứng minh: (a) Cho P1 = (A, <P1) và P2 = (A, <P2). thì P1&P2 = (A , <P1&P2). cho x, y ∈ dom(A) chúng ta có: x <P1&P2 y nếu x <P1 y ∨ (x = y ∧ x <P2 y) nếu x <P1 y ∨ false nếu x <P1 y

(b) Cho P1 = (A1, <P1) và P2 = (A2, <P2) với A1 ∩ A2 = □. cho x = (x1, x2), y = (y1, y2) ∈ dom(A1) × dom(A2) cho x <P1* y nếu x1 <P1 y1. Sau đó P1 là một thứ tự được nhúng vào P1* = (A1 ∪ A2, <P1*). Khi đó A1 ∩ A2 = □, P1* và P2 là ưa thích không giao nhau, do đó P1* và A1↔&P2 là cũng không giao nhau.

Do đó P1* + (A1↔ &P2) = (A1 ∪ A2, <P1*+(A1↔&P2)) là ưa thích kết hợp không giao nhau. Chúng ta có: x <P1+(A1↔&P2) y nếu x <P1*+(A1↔&P2) y nếu x1 <P1 y1 ∨ (x <A1↔&P2 y) nếu x1 <P1 y1 ∨ (x1 = y1 ∧ x2 <P2 y2) nếu x <P1&P2 y Q.e.d. (B)

Định lý "Non-discrimination": P1 □ P2 ≡ (P1 & P2) ♦ (P2 & P1)

Chứng minh: Cho P1 = (A1, <P1) và P2 = (A2, <P2).

Khi đó: P1 □ P2 = (A1 ∪ A2, <P1□P2) P1 & P2 = (A1 ∪ A2, <P1&P2), P2 & P1 = (A2 ∪ A1, <P2&P1) (P1 & P2) ♦ (P2 & P1) = (A1 ∪ A2, <(P1&P2)♦(P2&P1))

Cho x = (x1, x2) and y = (y1, y2) ∈ dom(A1) × dom(A2).

Rút gọn: B := 'x1 <P1 y1', C := 'x1 = y1', D := 'x2 <P2 y2', E := 'x2 = y2' If x = y, sau đó ¬B ∧ ¬D được thực hiện. Các trường hợp khác, nếu x ≠ y, sau đó ¬C ∨ ¬D thực hiện. (*) (1) x <P1□P2 y nếu (B ∧ (E ∨ D)) ∨ (D ∧ (C ∨ B))

nếu ((B∧ E)∨(B ∧ D)) ∨ ((D∧ C) ∨ (D ∧ B))

nếu (B ∧ E) ∨ (B ∧ D) ∨ (D ∧ C) (2) x <(P1&P2)♦(P2&P1) y

nếu (B ∨ (C ∧ D)) ∧ (D ∨ (E ∧ B))

nếu (B ∧ (D ∨ (E ∧ B))) ∨ ((C ∧ D) ∧ (D∨ (E ∧ B)))

nếu $(B \wedge D) \vee (B \wedge E \wedge B) \vee (C \wedge D \wedge D) \vee ((C \wedge E) \wedge D \wedge B)$ nếu $x <P1 \square P2\ y \vee ((C \wedge E) \wedge D \wedge B)$

(**) Hãy chú ý vào $H := C \wedge E \wedge D \wedge B$ trong (**) bên trên: Trong cả hai trường hợp $x \neq y$ hoặc $x = y$, dẫn đến (*) $\neg H$ bị ảnh hưởng. Do đó, ngay lập tức từ đại số Boolean, chúng ta có thể tiếp tục (**): khi và chỉ khi $x <P1 \square P2\ y$

(C) Định lý: $\sigma[P1+P2](R) = \sigma[P1](R) \cap \sigma[P2](R)$

Chứng minh: Nghiên cứu $P1+P2 = (A, <P1+P2)$, ưa thích Pareto $(P1+P2)^R$ và $w \in R[A]: w \in Nmax((P1+P2)^R)$ khi và chỉ khi $\square\ v \in R[A]: w <P1+P2\ v$ khi và chỉ khi $\square\ v \in R[A]: w <P1\ v \vee w <P2\ v$ khi đó P1 và P2 phải là ưa thíc không kết hợp, chúng ta có thể kết hợp:

khi và chỉ khi $(\square\ v \in R[A]: w <P1\ v) \vee (\square\ v \in R[A]: w <P2\ v)$

khi và chỉ khi $w \in Nmax(P1^R) \vee w \in Nmax(P2^R)$ Do đó: $Nmax((P1+P2)^R) = Nmax(P1^R) \cup Nmax(P2^R)$ thì:

$\sigma[P1+P2](R) = \{t \in R: t[A] \in max((P1+P2)^R)\}$

$= \{t \in R: t[A] \in R[A] - Nmax((P1+P2)^R)\}$

$= \{t \in R: t[A] \in R[A] - (Nmax(P1^R) \cup Nmax(P2^R))\}$

$= \{t \in R: t[A] \in (R[A] - Nmax(P1^R)) \cap (R[A] - Nmax(P2^R))\}$

$= \{t \in R: t[A] \in max(P1^R) \cap max(P2^R)\} = \sigma[P1](R) \cap \sigma[P2](R)$

(D) Định lý: $\sigma[P1 \blacklozenge P2](R) = \sigma[P1](R) \cup \sigma[P2](R) \cup YY(P1, P2)^R$

Chứng minh: Nghiên cứu $P1 \blacklozenge P2 = (A, <P1 \blacklozenge P2)$, ưa thích cơ sở dữ liệu $(P1 \blacklozenge P2)^R$ và $w \in R[A]: w \in Nmax((P1 \blacklozenge P2)^R)$

Khi và chỉ khi $\square\ v \in R[A]: w <P1 \blacklozenge P2\ v$

Khi và chỉ khi $\square\ v \in R[A]: w <P1\ v \wedge w <P2\ v$ tại điểm này chúng ta phải quan tâm khi thuộc tính xác định số lượng ở trong sự kết hợp:

nếu $\square\ v, v' \in R[A]: w <P1\ v \wedge w <P2\ v' \wedge (v \in P1{\uparrow}w \wedge v' \in P2{\uparrow}w \wedge v = v')$

nếu $(\square v \in R[A]: w <_{P1} v) \wedge (\square v' \in R[A]: w <_{P2} v') \wedge (\square v \in Nmax(P1^R), \square v' \in Nmax(P2^R): v \in P1\uparrow w \wedge v' \in P2\uparrow w \wedge v = v')$

nếu $w \in Nmax(P1^R) \wedge w \in Nmax(P2^R) \wedge (\square v \in Nmax(P1^R), \square v' \in Nmax(P2^R): v \in P1\uparrow w \wedge v' \in P2\uparrow w \wedge v = v')$ dẫn đến $XX(P1, P2)^R := \{w \in R[A]: \square v \in Nmax(P1^R), \square v' \in Nmax(P2^R): v \in P1\uparrow w \wedge v' \in P2\uparrow w \wedge v = v'\}$ chúng ta có:

nếu $w \in Nmax(P1^R) \wedge w \in Nmax(P2^R) \wedge w \in XX(P1, P2)^R$ do đó:

$Nmax((P1\blacklozenge P2)^R) = Nmax(P1^R) \cap Nmax(P2^R) \cap XX(P1, P2)^R$

sau đó chúng ta có: $\sigma[P1\blacklozenge P2](R) = \{t \in R: t[A] \in max((P1\blacklozenge P2)^R)\}$

$= \{t \in R: t[A] \in R[A] - Nmax((P1\blacklozenge P2)^R)\}$

$= \{t \in R: t[A] \in R[A] - (Nmax(P1^R) \cap Nmax(P2^R) \cap XX(P1, P2)^R)\}$

$= \{t \in R: t[A] \in (R[A] - Nmax(P1^R)) \cup (R[A] - Nmax(P2^R)) \cup (R[A] - XX(P1, P2)^R)\}$

$= \{t \in R: t[A] \in max(P1^R) \cup max(P2^R) \cup (R[A] - XX(P1, P2)^R)\}$

$= \sigma[P1](R) \cup \sigma[P2](R) \cup \{t \in R: t[A] \in R[A] - XX(P1, P2)^R\}$

Chúng ta có: $t[A] \in R[A] - XX(P1, P2)^R$

nếu $t[A] \square XX(P1, P2)^R$ nếu $t[A] \in \{w \in R[A]: \neg(\square v \in Nmax(P1^R), \square v' \in Nmax(P2^R): v \in P1\uparrow w \wedge v' \in P2\uparrow w \wedge v = v')$

nếu $\neg(\square v \in Nmax(P1^R), \square v' \in Nmax(P2^R): v \in P1\uparrow t[A] \wedge v' \in P2\uparrow t[A] \wedge v = v')$

nếu $\neg(t[A] \in Nmax(P1^R) \cap Nmax(P2^R): P1\uparrow t[A] \cap P2\uparrow t[A] \neq \square)$

nếu $(t[A] \in Nmax(P1^R) \cap Nmax(P2^R): P1\uparrow t[A] \cap P2\uparrow t[A] = \square)$ dẫn đến $YY(P1, P2)^R := \{t \in R : t[A] \in Nmax(P1^R) \cap Nmax(P2^R) \wedge P1\uparrow t[A] \cap P2\uparrow t[A] = \square\}$

kết quả là: $\sigma[P1\blacklozenge P2](R) = \sigma[P1](R) \cup \sigma[P2](R) \cup YY(P1, P2)^R$.