Quản lý khung nhìn trong CSDL phân tán

- Khung nhìn có thể được dẫn xuất từ các mảnh được lưu ở nhiều trạm khác nhau.
- Việc lưu trữ khung nhìn cần phải được xử lý như là lưu trữ CSDL
- Việc hiệu chỉnh truy vấn là kết quả của một truy vấn phân tán và có thể được xử lý bởi bộ truy vấn phân tán.
- Việc đánh giá khung nhìn có thể tốn kém nếu quan hệ cơ sở được phân tán
  - Sử dụng các khung nhìn cụ thể hóa

12

Khung nhìn cụ thể hóa

- Nguồn gốc: từ những năm 1980
  - Bản sao tĩnh của khung nhìn, tránh dẫn xuất khung nhìn cho mỗi truy vấn
  - Nhưng việc tính toán lại định kỳ khung nhìn có thể gây tốn kém
- Phiên bản thực tế của một khung nhìn
  - Được lưu như là quan hệ CSDL, có thể có các chỉ mục
- Được sử dụng nhiều trong thực tế
  - Hệ quản trị CSDL phân tán: Không cần truy nhập các quan hệ cơ sở từ xa
  - Kho dữ liệu: để tăng tốc độ xử lý phân tích trực tuyến (OLAP)
    - Sử dụng tổng hợp (SUM, COUNT,...) và GROUP BY

13

Duy trì khung nhìn cụ thể hóa

- Quá trình cập nhật (làm mới) khung nhìn để phản ánh những thay đổi đối với dữ liệu cơ sở
  - Tương tự như nhân bản dữ liệu, nhưng có một số khác biệt
    - Các biểu thức khung nhìn thường phức tạp hơn
    - Cấu hình nhân bản tổng quát hơn
- Chính sách duy trì khung nhìn cần xác định:
  - Khi nào cần làm mới
  - Làm mới như thế nào

14

Khi nào cần làm mới lại một khung nhìn?

- Chế độ trực tiếp
  - Là một phần của giao dịch cập nhật, ví dụ thông qua 2PC
  - Khung nhìn luôn luôn nhất quán với dữ liệu cơ sở và các truy vấn nhanh
  - Nhưng tăng thời gian giao dịch để cập nhật dữ liệu cơ sở
- Chế độ trì hoãn (được ưa dùng hơn trong thực tế)
  - Thông qua các giao dịch làm mới riêng
    - Không bị tốn kém chi phí khi cập nhật giao dịch
  - Được kích hoạt vào những thời điểm khác nhau với những yêu cầu khác nhau
    - "Lười": ngay trước khi đánh giá một truy vấn trên khung nhìn
    - Định kỳ: hàng giờ, hàng ngày,...
    - Bắt buộc: sau một số lần cập nhật được xác định trước

15

Làm mới lại một khung nhìn như thế nào?

- Tính toán đầy đủ từ dữ liệu cơ sở
  - Hiệu quả khi có nhiều thay đổi
- Tính toán gia tăng bằng cách chỉ áp dụng các thay đổi tới khung nhìn
  - Tốt hơn nếu một tập con nhỏ đã được thay đổi
  - Sử dụng các quan hệ vi phân để chỉ phản ánh dữ liệu cập nhật

Các quan hệ vi phân

Cho quan hệ R và cập nhật u

R^+ chứa các bộ được chèn bởi u

R^- chứa các bộ bị xóa bởi u

Các loại cập nhật u

chèn (insert)	R^+ rỗng
xóa (delete)	R^- rỗng
sửa đổi (modify)	$R^+ \cup (R - R^-)$

Sau đó, việc làm mới một khung nhìn V được thực hiện bằng cách tính:

$$V^+ \cup (V - V^-)$$

việc tính V^+ và V^- có thể yêu cầu truy nhập dữ liệu cơ sở.

Ví dụ

```
EG =      SELECT DISTINCT ENAME, RESP
          FROM   EMP, ASG
          WHERE  EMP.ENO=ASG.ENO

EG+=      (SELECT DISTINCT ENAME, RESP
          FROM   EMP, ASG+
          WHERE  EMP.ENO=ASG+.ENO) UNION
          (SELECT DISTINCT ENAME, RESP
          FROM   EMP+, ASG
          WHERE  EMP+.ENO=ASG.ENO) UNION
          (SELECT DISTINCT ENAME, RESP
          FROM   EMP+, ASG+
          WHERE  EMP+.ENO=ASG+.ENO)
```


Các kỹ thuật duy trì khung nhìn gia tăng

- Các kỹ thuật khác nhau phụ thuộc vào:
  - Thể hiện khung nhìn
    - Khung nhìn không đệ quy
    - Khung nhìn với kết nối ngoài (outerjoin)
    - Khung nhìn đệ quy
- Trường hợp thường gặp nhất là khung nhìn không đệ quy
  - Vấn đề: một bộ riêng trong khung nhìn có thể được dẫn xuất từ một số bộ cơ sở
    - Ví dụ: bộ ⟨M. Smith, Analyst⟩ trong EG tương ứng với
      - ⟨E2, M. Smith, … ⟩ trong EMP
      - ⟨E2,P1,Analyst,24 ⟩ và ⟨E2,P2,Analyst,6⟩ trong ASG
    - Làm cho việc xóa trở nên khó khăn
  - Giải pháp: đếm

Thuật toán đếm

- Ý tưởng cơ bản
  - Duy trì số lượng dẫn xuất cho mỗi bộ trong khung nhìn
  - Tăng (/giảm) số bộ dựa trên thao tác chèn (/xóa)
  - Một bộ trong khung nhìn có số lượng bằng 0 sẽ bị xóa.
- Thuật toán
  1. Tính V^+ và V^- sử dụng V, quan hệ cơ sở và các quan hệ vi phân
  2. Tính số dương trong V^+ và đếm số âm trong V^-
  3. Tính $V^+ \cup (V - V^-)$, xóa mỗi bộ trong V có số lượng=0
- Tối ưu: tính chính xác các bộ trong khung nhìn được chèn hoặc bị xóa

20

Khai thác dữ liệu nghiêng (Exploiting Data Skew)

- Ý tưởng cơ bản
  - Phân vùng quan hệ theo giá trị nặng/nhẹ cho các thuộc tính kết nối
    - Giá trị ngưỡng phụ thuộc vào kích thước dữ liệu và tham số người dùng
  - Duy trì kết nối các phần khác nhau bằng các kế hoạch khác nhau
    - Hầu hết các trường hợp được thực hiện bằng cách sử dụng xử lý delta (Đếm)
    - Một số ít trường hợp yêu cầu cụ thể hóa trước các khung nhìn phụ trợ
  - Cân bằng lại các phân vùng để phản ánh những thay đổi nặng/nhẹ
    - Lý do thay đổi:
      - Nhiều hơn/ít hơn số lần xuất hiện của một giá trị so với trước đó
      - Ngưỡng nặng/nhẹ thay đổi theo kích thước dữ liệu
    - Cập nhật số lần được thay đổi để tính đến việc cân bằng lại thường xuyên

21

Khung nhìn tự duy trì

- Một khung nhìn có thể tự duy trì nếu không cần truy nhập vào các quan hệ cơ sở.
  - Không phải trường hợp của thuật toán đếm
- Tự duy trì phụ thuộc vào các đặc tả của khung nhìn
  - Hầu hết các khung nhìn SPJ thường tự duy trì phép sửa đổi hoặc xóa, nhưng không duy trì phép chèn.
  - Ví dụ: một khung nhìn V là khung nhìn duy trì cho phép xóa trong R khi khóa của R được bao gồm trong V.

22