Tối ưu hóa truy vấn II

Tối thiểu hóa DISTINCT

- DISTINCT loại bỏ các bộ dữ trùng lặp liệu từ kết quả truy vấn.
- Mục tiêu: tránh DISTINCT nếu có thể

➔Làm sao để biết khi nào DISTINCT là cần thiết?

Sử dụng: bảng đặc quyền và reachability để kiểm tra xem có sự trùng lặp trong kết quả truy vấn hay không

Bảng đặc quyền

- Bảng đặc quyền: các thuộc tính được trả về với mệnh đề SELECT chứa một key.
- VD: lấy mã an ninh của tất cả các nhân viên làm việc trong phòng công nghệ

```
SELECT ssnum
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
```

- Employee là một bảng ưu tiên:
  - Mệnh đề SELECT thực hiện phép chiếu đến các thuộc tính ssnum
  - ssnum là key của Employee

Reachability

- R và S là bảng
- R gọi là reach S nếu:
  - R và S được join như nhau
  - Thuộc tính join trong R là key của R
- Intuition(trực giác): A bộ dữ liệu từ S chỉ được join nhiều nhất 1 bộ dữ liệu từ R
- Reachability có tính bắc cầu: nếu A reach B và B reach C thì A reach C.

Reachability VD

- VD trước: lấy mã an ninh của tất cả các nhân viên phòng công nghệ

```
SELECT ssnum
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
```

- Techdept reach Employee:
  - Techdept và Employee được join như nhau
  - dept là key của Techdept

Đảm bảo tính không trùng lặp

- ✓Một truy vấn không có bản sao nếu theo những điều kiện sau:
  - Mỗi thuộc tính trong mệnh đề SELECT là từ bảng đặc quyền.
  - Mỗi bảng không ưu tiên reach ít nhất một bảng ưu tiên.

VD: câu truy vấn sau có trả lại trùng lặp không?

```
SELECT ssnum
FROM Employee, Techdept
WHERE Employee.manager = Techdept.manager
```

→Có, vì:

- Manager không phải là key của Techdept
- Nên Techdept không reach bảng đặc quyền Employee.

```
SELECT ssnum, Techdept.dept
FROM Employee, Techdept
WHERE Employee.manager = Techdept.manager
```

→Không, vì nó khó khác với VD trước.
Cả Techdept và Employee đều là bảng đặc quyền

➢ VD: các câu truy vấn sau có trả lại trùng lặp không?

```
SELECT ssnum, Techdept.dept
FROM Employee, Techdept
```

→ Không, bảng Techdept và Employee là bảng đặc quyền.

```
SELECT Student.ssnum
FROM Student, Employee, Techdept
WHERE Student.name = Employee.name
AND Employee.dept = Techdept.dept
```

(Student.name không phải là key)

→ Không, vì:

- Thuộc tính: Employee.name là key nên Employee reach bảng đặc quyền Student (1)
- Thuộc tính: Techdept.dept là key nên Techdept reach bảng đặc quyền Employee (2)

(1),(2) → Techdept reach Student.

VD: câu truy vấn sau có trả lại trung lặp hay không?

(Student.name không phải là key)

```
SELECT Student.ssnum
FROM Student, Employee, Techdept
WHERE Student.name = Employee.name
AND Employee.manager = Techdept.manager
```

→ Có, vì:

- Thuộc tính Techdept.manager không phải là key nên Tchdept không reach Employee (và Student)

❑ Thử các câu truy vấn VD theo CSDL sau:

Employee(ssnum, name, manager, dept)

ssnum	name	manager	dept
1	Peter	John	IT
2	Rose	Mary	Development

Techdept(dept, manager)

dept	manager
IT	John
Development	Mary
Production	John

Students(ssnum, name)

ssnum	name
5	Peter
6	Peter

Viết lại các truy vấn lồng nhau

- Truy vấn con không tương quan:
  - Có các phép toán tập hợp ở truy vấn con:

```
SELECT ssnum
FROM Employee
WHERE salary > (SELECT AVG(salary) FROM Employee)
```

→Không rắc rối:

- Kết quả của truy vấn con là một đơn giá trị (hằng số)
- Hầu hết các hệ thống sẽ thực thi truy vấn con trước rồi thay thế nó với hằng số kết quả.

  - Không có phép toán tập hợp ở truy vấn con:

```
SELECT ssnum
FROM Employee
WHERE dept IN (SELECT dept FROM Techdept)
```

→ Vài hệ thống sẽ không sử dụng index trên Employee.dept

- Truy vấn con tương quan:
  - Với các phép toán tập hợp trong truy vấn con:

```
SELECT ssnum
FROM Employee e1, Techdept
WHERE salary = (SELECT AVG(e2.salary)
                FROM Employee e2, Techdept
                WHERE e2.dept = e1.dept
                AND e2.dept = Techdept.dept)
```

  - Không có phép toán tập hợp trong truy vấn con (uncommon)
- Truy vấn không lồng:

```
SELECT ssnum
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
```

- Chiến lược không lồng:
  1. Kết hợp các tham số của 2 mệnh đề FROM
  2. AND các mệnh đề WHERE
  3. Thay thế “outer.attr1 IN (SELECT inner.attr2 …)” bằng “outer.attr1 = inner.attr2” trong mệnh đề WHERE.
  4. Giữ lại mệnh đề SELECT từ khối bên ngoài
- Chiến lược này hoạt động hiệu quả với các truy vấn lồng ở bất cứ độ sâu nào
- Chú ý: nếu bảng bên trong không reach bảng bên ngoài trong điều kiện join mới, sự trùng lặp dữ liệu mới có thể xuất hiện.

- Truy vấn lồng nhau:

```
SELECT AVG(salary)
FROM Employee
WHERE dept IN (SELECT dept FROM Techdept)
```

- Truy vấn không lồng nhau:

```
SELECT AVG(salary)
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
```

- Truy vấn không lồng nhau là đúng:
  - Techdept reach Employee, nên không có dữ liệu bị trùng lặp
  - Mỗ salary xuất hiện 1 lần trong phép tính trung bình

- Truy vấn lồng:

```
SELECT AVG(salary)
FROM Employee
WHERE manager IN (SELECT manager FROM Techdept)
```

- Truy vấn không lồng:

```
SELECT AVG(salary)
FROM Employee, Techdept
WHERE Employee.manager = Techdept.manager
```

- Truy vấn không lồng là sai:
  - Techdept không reach Employee, nên có thể xảy ra trung lặp dữ liệu
  - Một vài salary có thể xuất hiện nhiều lần trong phép tính trung bình
- Chú ý: trung lặp dữ liệu không ảnh hưởng trong các phép toán tập hợp như MIN, MAX.

- Giải pháp cho câu truy vấn sau:

```
SELECT AVG(salary)
FROM Employee
WHERE manager IN (SELECT manager FROM Techdept)
```

- Tạo bảng tạm:

```
SELECT DISTINCT manager INTO Temp
FROM Techdept

SELECT AVG(salary)
FROM Employee, Temp
WHERE Employee.manager = Temp.manager
```

- Truy vấn con tương quan với các phép toán tập hợp bên trong:

```
SELECT ssnum
FROM Employee e1, Techdept
WHERE salary = (SELECT AVG(e2.salary)
                FROM Employee e2, Techdept
                WHERE e2.dept = e1.dept
                AND e2.dept = Techdept.dept)
```

- Không hiệu quả trong nhiều hệ thống

Chiến lược viết lại truy vấn:

```
SELECT ssnum
FROM Employee e1, Techdept
WHERE salary = (SELECT AVG(e2.salary)
                FROM Employee e2, Techdept
                WHERE e2.dept = e1.dept
                AND e2.dept = Techdept.dept)
```

1. Tạo bảng tạm

- GROUP BY trên các thuộc tính tương quan trong truy vấn con
- Sử dụng các cấp độ không tương quan của truy vấn bên trong mệnh đề WHERE.

```
SELECT AVG(salary) as avsalary, Employee.dept INTO Temp
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
GROUP BY Employee.dept
```

Chiến lược viết lại truy vấn:

```
SELECT ssnum
FROM Employee e1, Techdept
WHERE salary = (SELECT AVG(e2.salary) ... )
```

```
SELECT AVG(salary) as avsalary, Employee.dept INTO Temp
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
GROUP BY Employee.dept
```

2. Join bảng tạm với truy vấn ngoài:

   - Điều kiện về các nhóm thuộc tính thay thế các điều kiện tương quan.
   - Dựa vào thuộc tính của nhóm thay thế các truy vấn con.
   - Tất cả mức của các truy vấn ngoài vẫn được giữ nguyên.

```
SELECT ssnum
FROM Employee, Temp
WHERE salary = avsalary
AND Employee.dept = Temp.dept;
```

The count bug

- Truy vấn con tương quan với phép toán tập hợp COUNT bên trong truy vấn lồng:

```
SELECT ssnum
FROM Employee e1, Techdept
WHERE numfriends = COUNT(SELECT e2.ssnum
                         FROM Employee e2, Techdept
                         WHERE e2.dept = e1.dept
                         AND e2.dept = Techdept.dept)
```

- Viết lại với bảng tạm:

```
SELECT COUNT(ssnum) as numcolleagues, Employee.dept INTO Temp
FROM Employee, Techdept
WHERE Employee.dept = Techdept.dept
GROUP BY Employee.dept

SELECT ssnum
FROM Employee, Temp
WHERE numfriends = numcolleagues
AND Employee.dept = Temp.dept;
```

→ Có gì đang sai?

The count bug

- Xem xét ví dụ một employee Jane:
  - Jane không phải trong một technical department (Techdept)
  - Jane không có bạn bè (Employee.numfriends = 0)
- Truy vấn lồng:
  - Kể từ khi Jane không trong một technical department, truy vấn lồng là rỗng
  - Nhưng COUNT(∅) = 0 nên Jane sẽ nằm trong kết quả cuối cùng
- Truy vấn viết lại với bảng tạm:
  - Jane không trong technical department và không tồn tại trong join nên Jane không trong kết quả cuối cùng