分布式数据库系统

讲解人：李鸿岐

分布式数据库查询和优化

分布式数据库查询和优化

查询优化的基础

查询处理问题

- 集中查询处理器必须：
  - ✓ 将演算查询转换为代数操作
  - ✓ 选择最好的执行计划
- 例如:

```
SELECT ENAME
FROM E,G
WHERE RESP = “Manager” and
      E.ENO=G.ENO
```

分布式数据库查询和优化

● 查询优化概述

■关系代数 1:

$$\pi_{ENAME}(\sigma_{RESP="Manager" \wedge E.EMO=G.ENO}(E \times G))$$

■关系代数2:

$$\pi_{ENAME}(E \infty_{ENO}(\sigma_{RESP="Manager"}(G)))$$

■可见： 执行计划 2 好

■对于分布式数据库，查询处理器必须考虑通信代价和选择最佳场地!

如上例：若 G 和 E 是分布的.

简单计划是将所有片段传到查询场地并执行，但通信代价太大。

分布式数据库查询和优化

查询优化概述

优化目标

优化就是寻找执行代价（费用和时间）最小的查询执行策略，使系统执行消耗降到最低。

优化的目标就是指**局部执行代价**和**网络传输代价**的和最小。

- **局部执行代价**：主要指输入/输出次数（I/O代价）及CPU处理代价。
- **网络传输代价**：主要指传输启动代价和数据传输代价。

分布式数据库查询和优化

查询优化概述

优化内容

优化内容体现如下几点：

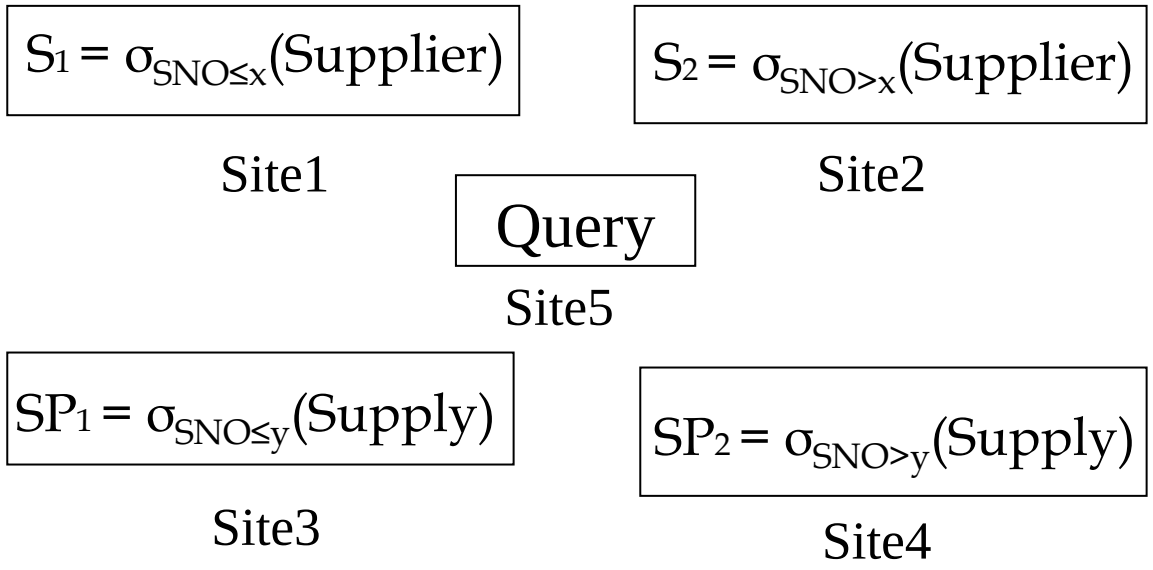
- 执行运算的次序。
- 执行每种运算的方法，不同方法代价不同。
- 所访问的副本场地。如：选择就近的场地，节约传输代价。
- 执行运算的场地的选择，使总的传输代价或总代价最低。

综合考虑，确定出一种**执行代价最小的查询执行策略**。

分布式数据库查询和优化

● 查询优化概述

□ 优化内容—查询案例：



■ 查询：

找出供应**100**号零件的供应者的姓名

[案例] 设一供应关系数据库，含供应者和供应关系

供应者： **Supplier（SNO, SNAME, AREA）**

供应关系：**Supply （SNO, PNO, QTY）**

零件关系：**Part（PNO, PNAME）**

设各场地的数据量如下：场地**1**和场地**2**各有**5000**个**Supplier**元组，场地**3**和场地**4**各有**50000**个**Supply**元组。假设广域网环境下，对于**Supplier**关系的传输速度是**20**个元组**/**秒，对于关系**Supply**的传输速度是**50**个元组**/**秒，通信延迟为**1**秒。局部操作的代价是10^4元组**/**秒。

分布式数据库查询和优化

● 查询优化概述

$S_1 = \sigma_{SNO \le x}(Supplier)$ Site1

$S_2 = \sigma_{SNO > x}(Supplier)$ Site2

Query Site5

$SP_1 = \sigma_{SNO \le y}(Supply)$ Site3

$SP_2 = \sigma_{SNO > y}(Supply)$ Site4

□ 执行计划1：

■ 所有场地的数据分片传到查询场地5并执行选择、连接、投影等运算

$$Q = \Pi_{SNAME}((S_1 \cup S_2) \bowtie_{SNO} \sigma_{PNO=100}(SP_1 \cup SP_2))$$

1. 从场地1、场地2传输S1、S2数据到场地5的总传输时间是：1 + (5000*2)/20 = 501s 。
2. 从场地3、场地4传输SP1、SP2数据到场地5的总传输时间是：1+ (50000*2)/50 = 2001s。
3. 在场地5对SP表执行选择操作的时间是： (50000*2)/10^4 = 10s。
4. 假设SP选择后形成SP’表，有100个元组，则S和SP’连接操作的时间为：（5000*2）×100/ 10^4 = 100s。

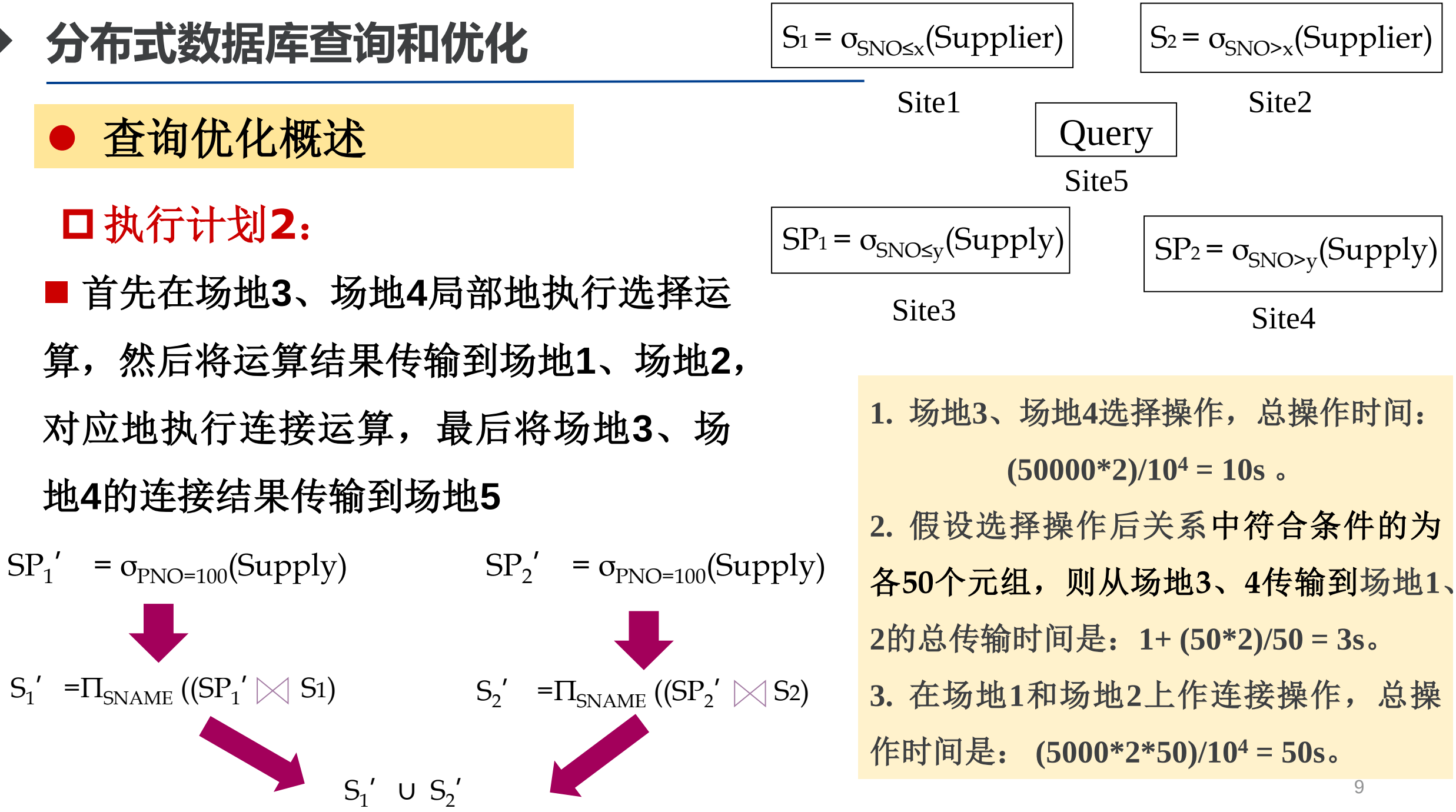
时间总计：501+ 2001+10 + 100 = 2612s ≈ 43.5min

分布式数据库查询和优化

查询优化概述

执行计划2：

■ 首先在场地3、场地4局部地执行选择运算，然后将运算结果传输到场地1、场地2，对应地执行连接运算，最后将场地3、场地4的连接结果传输到场地5



1. 场地3、场地4选择操作，总操作时间：$(50000*2)/10^4 = 10s$ 。
2. 假设选择操作后关系中符合条件的为各50个元组，则从场地3、4传输到场地1、2的总传输时间是：$1+ (50*2)/50 = 3s$。
3. 在场地1和场地2上作连接操作，总操作时间是：$(5000*2*50)/10^4 = 50s$。

分布式数据库查询和优化

查询优化概述

执行计划2：

首先在场地3、场地4局部地执行选择运算，然后将运算结果传输到场地1、场地2，对应地执行连接运算，最后将场地3、场地4的连接结果传输到场地5

$S_1 = \sigma_{SNO \le x}(Supplier)$ Site1

$S_2 = \sigma_{SNO > x}(Supplier)$ Site2

Query Site5

$SP_1 = \sigma_{SNO \le y}(Supply)$ Site3

$SP_2 = \sigma_{SNO > y}(Supply)$ Site4

$SP_1' = \sigma_{PNO=100}(Supply)$

$SP_2' = \sigma_{PNO=100}(Supply)$

$S_1' = \Pi_{SNAME}((SP_1' \bowtie S1)$

$S_2' = \Pi_{SNAME}((SP_2' \bowtie S2)$

$S_1' \cup S_2'$

1. 场地3、场地4选择操作，总操作时间：

4. 假设连接和投影操作后，关系中元组大小缩减为原来一半，则传输速率变为40个元组/秒。从场地1、场地2传输到场地5需要的总传输时间：1+2*50/40=3.5s。

5. 总操作时间：

10 + 3 + 50 + 3.5 = 66.5s ≈ 1.1分钟。

分布式数据库查询和优化

● 查询优化概述

□ 查询处理的目标

- 两方面：

  转换（transformation）和优化（optimization）

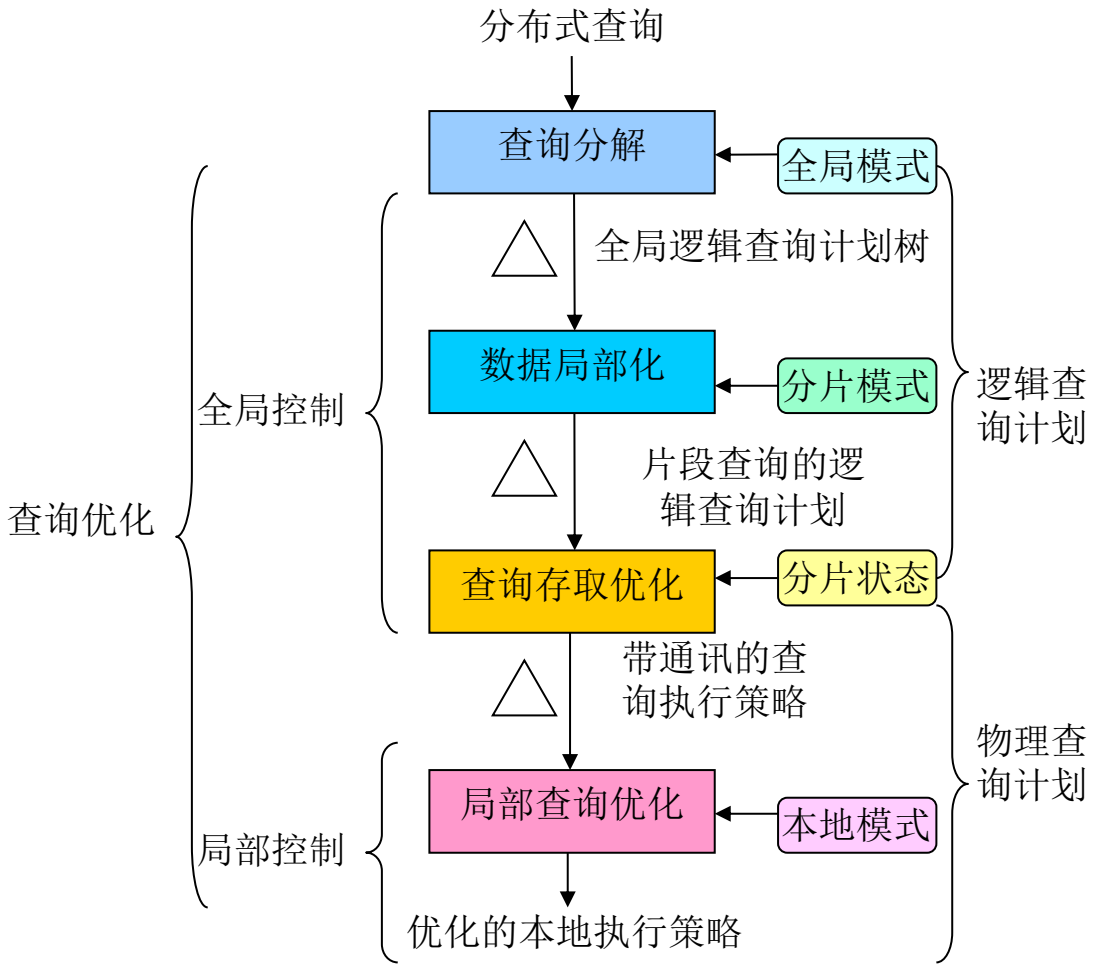
- 考虑的优化代价：
  - ✓ CPU time
  - ✓ I/O time
  - ✓ Communication time
- 在WAN内，侧重通信代价

  在LAN内，三者同等重要

分布式数据库查询和优化

查询处理层次

- 分布式查询处理过程
- 查询分解/查询映射
- 数据本地化
- 片段查询的优化



分布式数据库查询和优化

● 查询处理层次

□ 分布式查询处理过程

1. 查询分解/查询映射（Query Decomposition）

基于全局概念模式将演算查询分解为代数查询。

Step1 – 演算规范化

Step2 – 语义分析，去掉不正确的查询。

Step3 – 简化，去除冗余的部分

Step4 – 将演算查询转化为优化的代数查询。

- □ 不考虑实际的数据分布和复制
- □ 转换很大程度上与集中式DBMS的转换相同

分布式数据库查询和优化

● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

输入：基于全局关系的演算查询

■ 规范化（Normalization）

将查询转换为规范化形式（AND形式或OR形式 ）

对于SQL语句，对Where子句中的谓词表示进行规范化。

词法、语法分析（详见“关系数据库系统的查询优化”）

转换为规范化表示形式

合取范式（Conjunctive normal form） $(p_{11} \vee p_{12} \vee \ldots \vee p_{1n}) \wedge \ldots \wedge (p_{m1} \vee p_{m2} \vee \ldots \vee p_{mn})$

析取范式（Disjunctive normal form ） $(p_{11} \wedge p_{12} \wedge \ldots \wedge p_{1n}) \vee \ldots \vee (p_{m1} \wedge p_{m2} \wedge \ldots \wedge p_{mn})$

OR映射为并操作（union）

AND映射为连接（join）或选择（selection）

分布式数据库查询和优化

● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

输入：基于全局关系的演算查询

■ 规范化（Normalization）

将查询转换为规范化形式（AND形式或OR形式 ）

■分析（Analysis）

检查不正确的查询 & 得到关系演算子集

目的: 拒绝类型不正确或语义不正确的查询

```
SELECT E#          !undefined attribute
FROM  E
WHERE ENAME>200  !type mismatching
```

■ 约简（Simplification）

删除冗余谓词

■ 查询重写（Restructuring）

→ 将演算查询转换为关系代数查询

→ 可能得到多种转换结果

→ 使用转换规则

分布式数据库查询和优化

● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

■ 查询重写

✓ 直接将关系演算转换为关系代数；

✓ 重写关系代数查询以提高性能。

■ 通常使用操作树来表示关系代数查询。

✓ **关系代数树**定义为：

- 根节点表示**查询结果**
- 叶子节点表示**关系**
- 非叶子节点表示**中间结果关系**
- 叶子到根的边表示**序列操作**

✓ SQL查询转换为代数树：

- 叶子来自于FROM子句；
- 根节点是结果关系，由所需要属性的投影操作生成，包含在SELECT子句中；
- WHERE子句中的条件被转换成从叶子节点到根节点的关系操作序列，操作序列由操作符和谓词出现的顺序直接生成。

分布式数据库查询和优化

● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

■ 查询重写

查询重写是将用户请求构成的查询树进行等价变换

规则1: 连接、笛卡尔积的交换律

$R\times S \Leftrightarrow S\times R$，$R\infty S \Leftrightarrow S\infty R$

规则2: 连接、笛卡尔积的结合律

$(R\times S)\times T \Leftrightarrow R\times(S\times T)$， $(R\infty S)\infty T \Leftrightarrow R\infty(S\infty T)$

规则3: 投影的串接定律

$\pi_{A1,A2,...,An}(\pi_{B1,B2,...,Bn}(E)) \Leftrightarrow \pi_{A1,A2,...,An}(E)$

规则4: 选择的串接定律

$\sigma_{P1}(\sigma_{P2}(R)) \Leftrightarrow \sigma_{P1\wedge P2}(R)$

规则5: 选择和投影的交换律

$\sigma_P(\pi_{A1,A2,...,An}(R)) \Leftrightarrow \pi_{A1,A2,...,An}(\sigma_P(R))$

规则6: 选择与笛卡尔积的分配律

$\sigma_P(R\times S) \equiv \sigma_P(R)\times S$

规则7: 选择与并的分配律

$\sigma_P(R\cup T) \Leftrightarrow \sigma_P(R)\cup \sigma_P(T)$

规则8: 选择与差的分配律

$\sigma_P(R-T) \Leftrightarrow \sigma_P(R)-\sigma_P(T)$

规则9: 投影与笛卡尔积的分配律

$\pi_{A1,A2,...,An,\ B1,B2,...,Bn}(R\times S) \equiv \pi_{A1,A2,...,An}(R)\times\pi_{B1,B2,...,Bn}(S)$

规则10: 投影与并的分配律

$\pi_{A1,A2,...,An}(R\cup F) \equiv \pi_{A1,A2,...,An}(R)\cup\pi_{A1,A2,...,An}(F)$

分布式数据库查询和优化

● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

■ 查询重写

✓ 应用等价变换规则，一棵查询树可等价转换为多棵查询树，需要能生成最优查询树的等价变换。

✓ 关系查询优化的基本思想是先做能使中间结果变小的操作，尽量减少查询执行代价。

运算类型	执行代价（n为关系元组个数）
σ、Π（不消重复项）	O（n）
Π（消重复项）、GROUP	O（nLOGn）
X（笛卡尔积）	O（n^2）
∞、∪、∩、-、∝、÷	O（nLOGn）

□ 关系代数

■ 一元运算 U：σ（选择）/Π（投影）

■ 二元运算

b：∞（联接）/X（笛卡儿积）/∪（并）/∩（交）/-（差）/∝（半联接）

变换的通用准则为：

准则1 尽可能将一元运算移到查询树的底部（树叶部分），优先执行一元运算。

准则2 利用一元运算重复律，缩减每一关系，减少关系尺寸，降低网络传输量和I/O大小。

分布式数据库查询和优化

● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

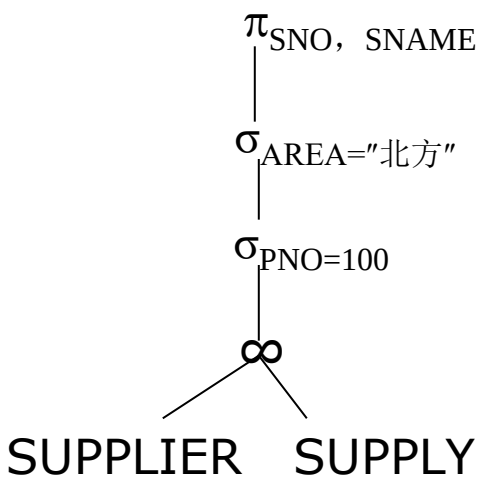
■ 查询重写

■ 案例

SUPPLIER{SNO，SNAME，AREA}
SUPPLY{SNO，PNO，QTY}

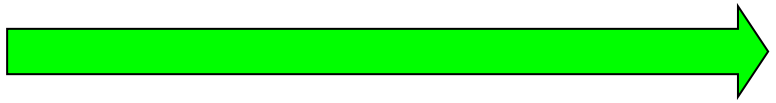
Q1： $\pi_{SNO,SNAME}(\sigma_{AREA="北方"}(\sigma_{PNO=100}(SUPPLIER \infty SUPPLY)))$

查询树：



□全局优化

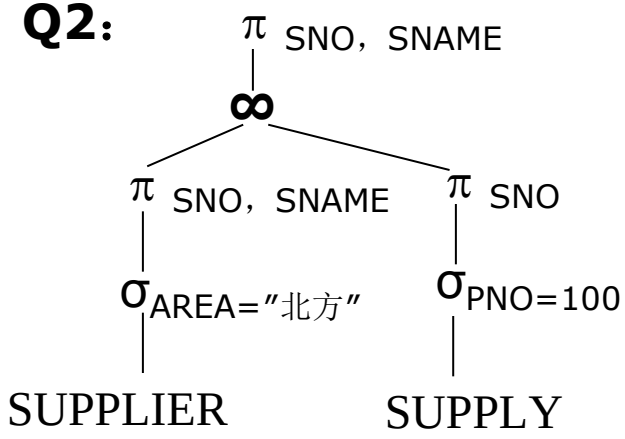
基于查询重写准则（在查询树Q1的基础上进行全局优化）



根据分配律，将一元运算向下移。

得到全局优化后的查询树：

Q2:



分布式数据库查询和优化

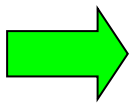
● 查询处理层次

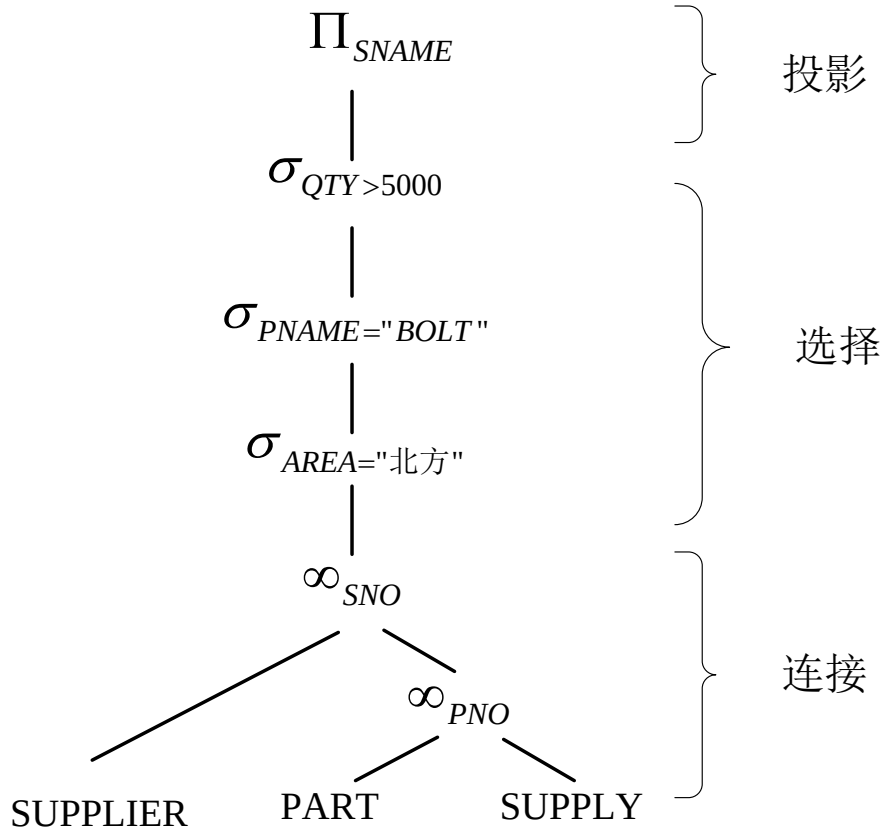
1. 查询分解/查询映射（Query Decomposition）

■ 查询重写

■ 案例 2

```
SELECT SNAME
FROM  SUPPLIER S，SUPPLY SP，PART P
WHERE S.SNO = SP.SNO
AND SP.PNO = P.PNO
AND P.PNAME = “BOLT”
AND S.AREA = “北方”
AND SP.QTY > 5000;
```



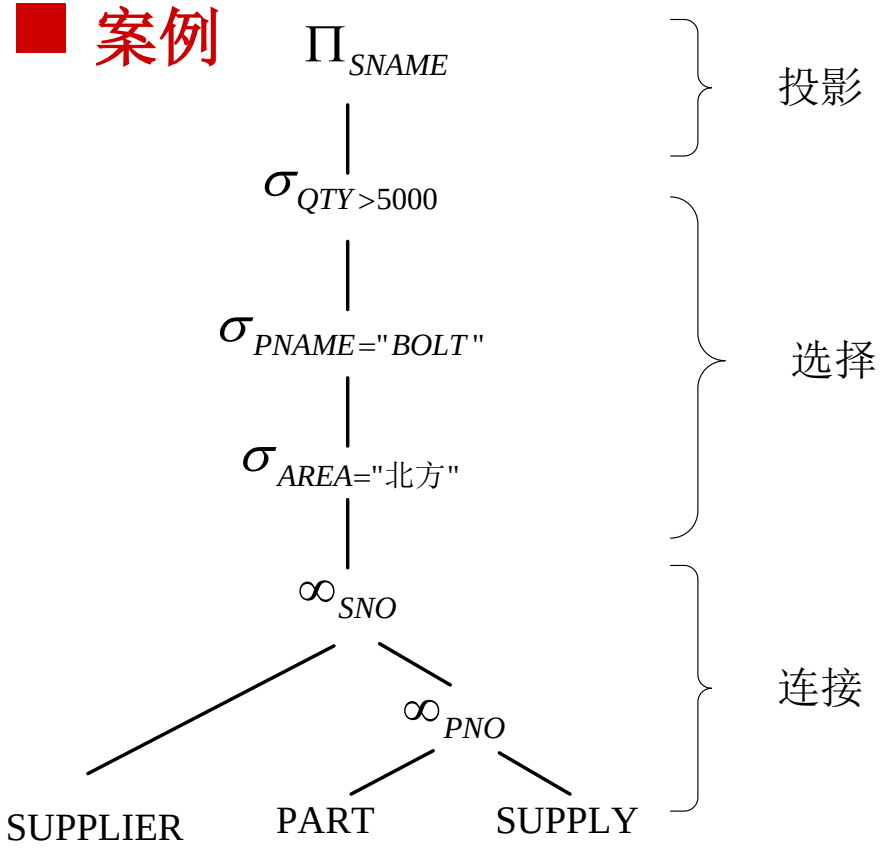


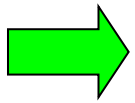
分布式数据库查询和优化

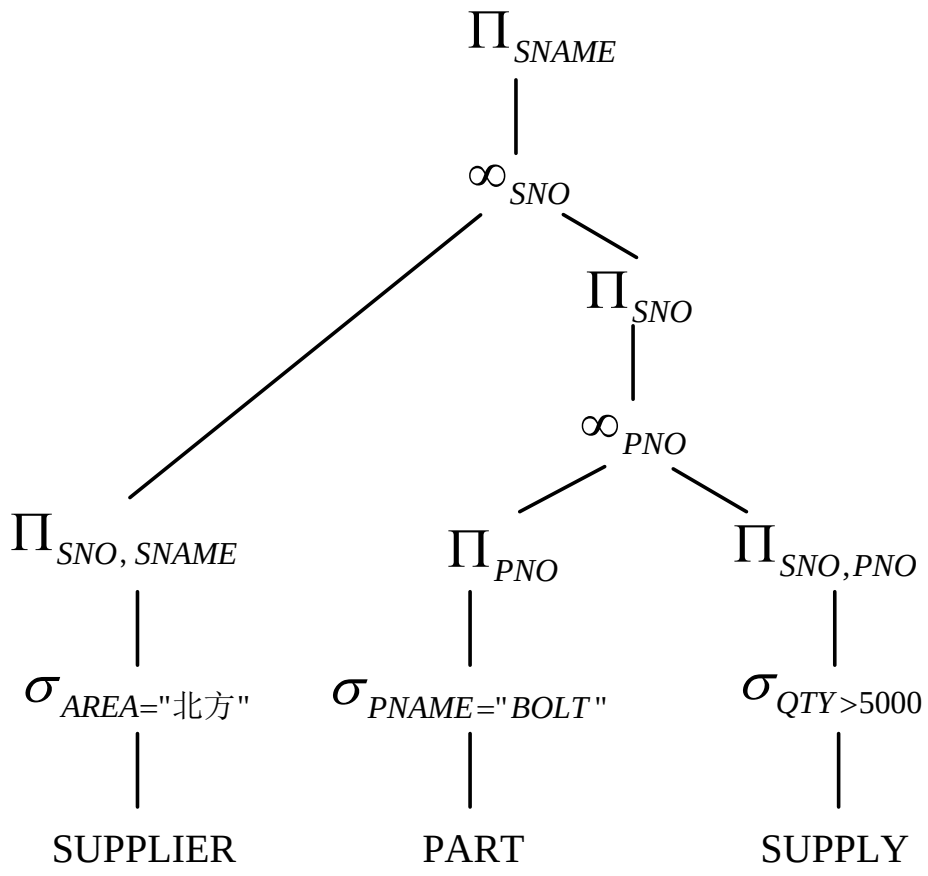
● 查询处理层次

1. 查询分解/查询映射（Query Decomposition）

- 查询重写
- 案例







● 查询处理层次

□ 分布式查询处理过程

2. 数据本地化（Data Localization）

分布查询映射为片段查询，简化、重组为优化的查询。

- 查询的处理过程是从全局关系到片段关系，最后再到实际操作的副本关系。
- 数据本地化将介绍全局查询到片段查询的变换。

即利用全局关系与其片段关系的等价变换，将分布查询中的全局关系替换为对片段关系的查询，变换后的查询称为片段查询。对应于片段查询的查询树，称为片段查询树。

分布式数据库查询和优化

● 查询处理层次　□ 分布式查询处理过程

2. 数据本地化（Data Localization）

□ 分布查询与片段查询的等价关系

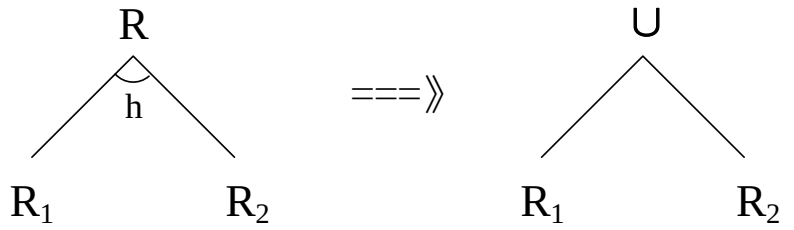
- 对于全局关系R的水平分片R_1，R_2，…，R_n，表示为：

$$R=R_1 \cup R_2 \cup \ldots \cup R_n$$

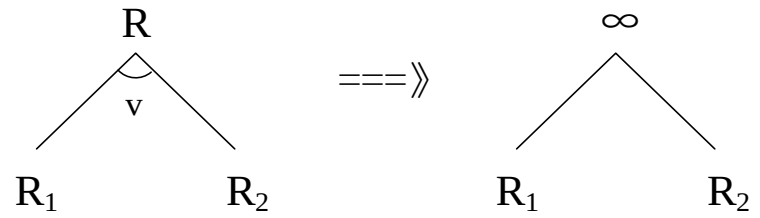
- 对于全局关系R的垂直分片R_1，R_2，…，R_n，表示为：

$$R=R_1 \infty R_2 \infty \ldots \infty R_n$$

水平h节点分片树　　转换后的∪节点



垂直v节点分片树　　转换后的∞节点



□片段查询树的生成步骤

- 将分片树的h（水平）节点转换为查询树的∪（并集）节点。
- 将分片树的v（垂直）节点转换为查询树的∞（联接）节点。
- 用替换后的分片树代替全局查询树中的全局关系，得到片段查询树。

分布式数据库查询和优化

● 查询处理层次　□ 分布式查询处理过程

2. 数据本地化（Data Localization）

□ **例：** 假设SUPPLIER和SUPPLY的分片如下：

- SUPPLIER水平分片为S_1和S_2，具体如下：

  $S_1=\sigma_{AREA=\text{"北方"}}$（SUPPLIER）

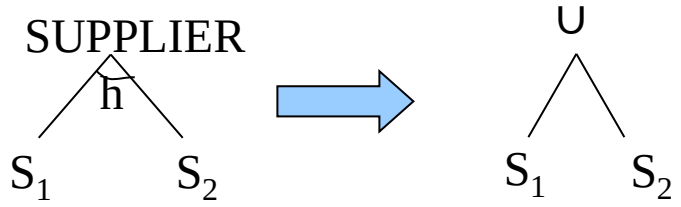
  $S_2=\sigma_{AREA=\text{"南方"}}$（SUPPLIER）

- SUPPLY水平分片为$SUPPLY_1$和$SUPPLY_2$，具体如下：

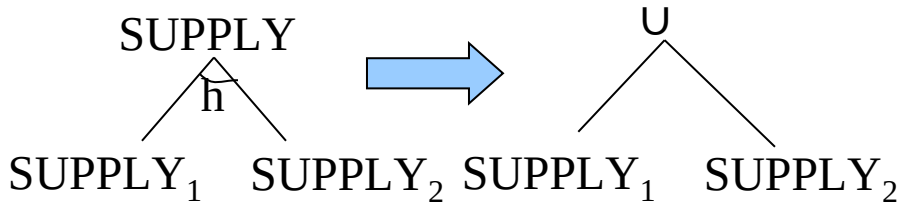
  $SUPPLY_1=\sigma_{AREA=\text{"北方"}}$（SUPPLY）

  $SUPPLY_2=\sigma_{AREA=\text{"南方"}}$（SUPPLY）

SUPPLIER的分片树为：　转换后的∪节点：



SUPPLY的分片树为：　转换后的∪节点：



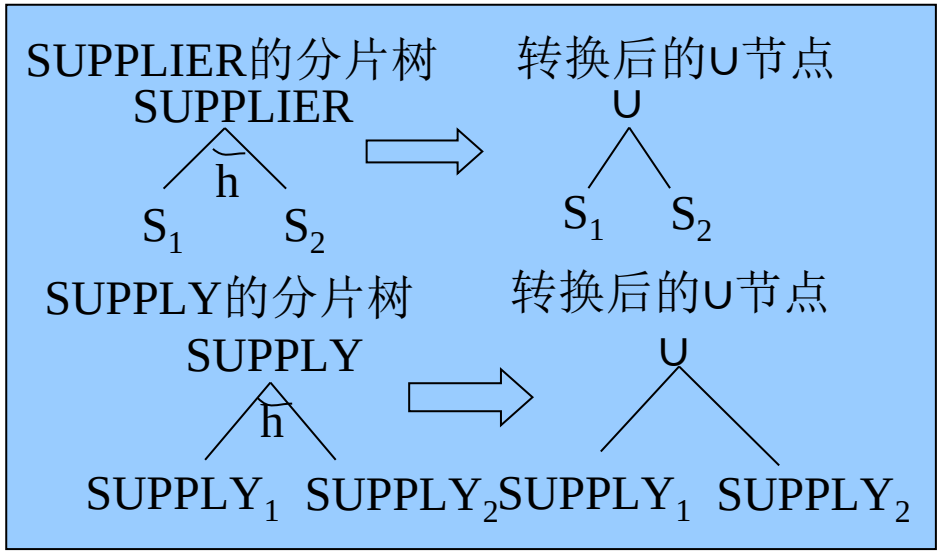
分布式数据库查询和优化

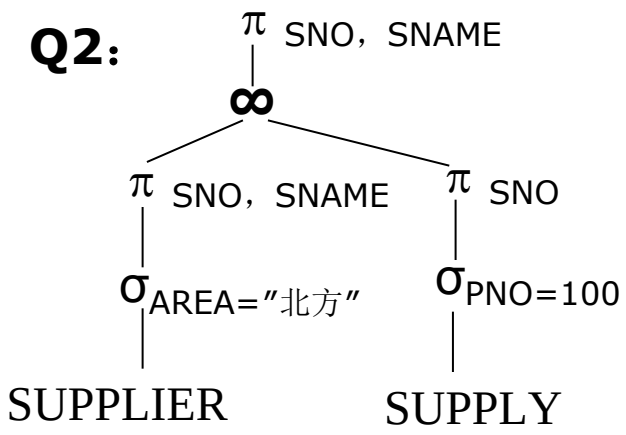
● 查询处理层次　□ 分布式查询处理过程

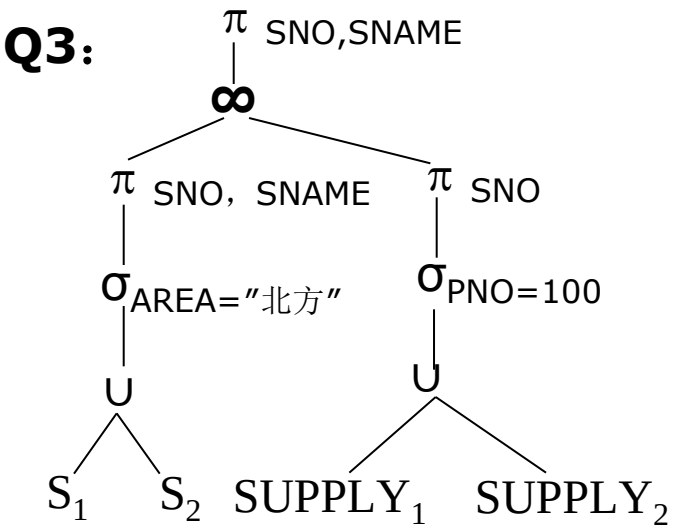
2. 数据本地化（Data Localization）

■ 转换为片段查询树Q3

在Q2基础上，用SUPPLIER分片树替换查询树Q2的全局关系SUPPLIER，用SUPPLY分片树替换查询树Q2的全局关系SUPPLY，即得到转换后的片段查询树为Q3。







● 查询处理层次　□ 分布式查询处理过程

2. 数据本地化（Data Localization）

□ 片段查询优化规则

准则1：对于一元运算，根据一元运算的重复律，将叶子节点之前的选择运算作用于片段，如果不满足片段的限定条件，则置为空关系。

准则2：对于联接运算的树，若联接条件不满足，则将其置为空关系。

准则3：在查询树中，将联接运算（∞）下移到并运算（∪）之前执行。

准则4：消去不影响查询运算的垂直片段。

分布式数据库查询和优化

● 查询处理层次

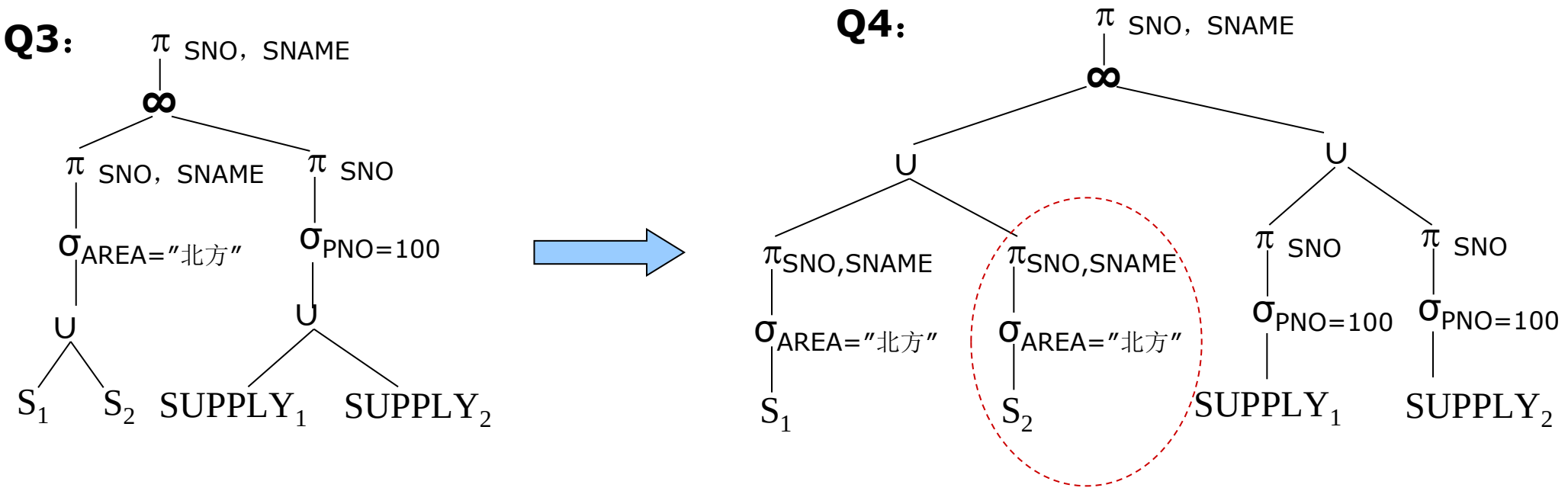
□ 分布式查询处理过程

2. 数据本地化（Data Localization）

准则1：对于一元运算，根据一元运算的重复律，将叶子节点之前的选择运算作用于片段，如果不满足片段的限定条件，则置为空关系。

■接上例，以片段查询树**Q3**为基础，

（1）根据片段查询优化准则1，按限定条件化简，得到Q4。



分布式数据库查询和优化

● 查询处理层次

2. 数据本地化（Data Localization）

□ 片段查询优化

（2）根据

$S_1=\sigma_{AREA="北方"}$（SUPPLIER）

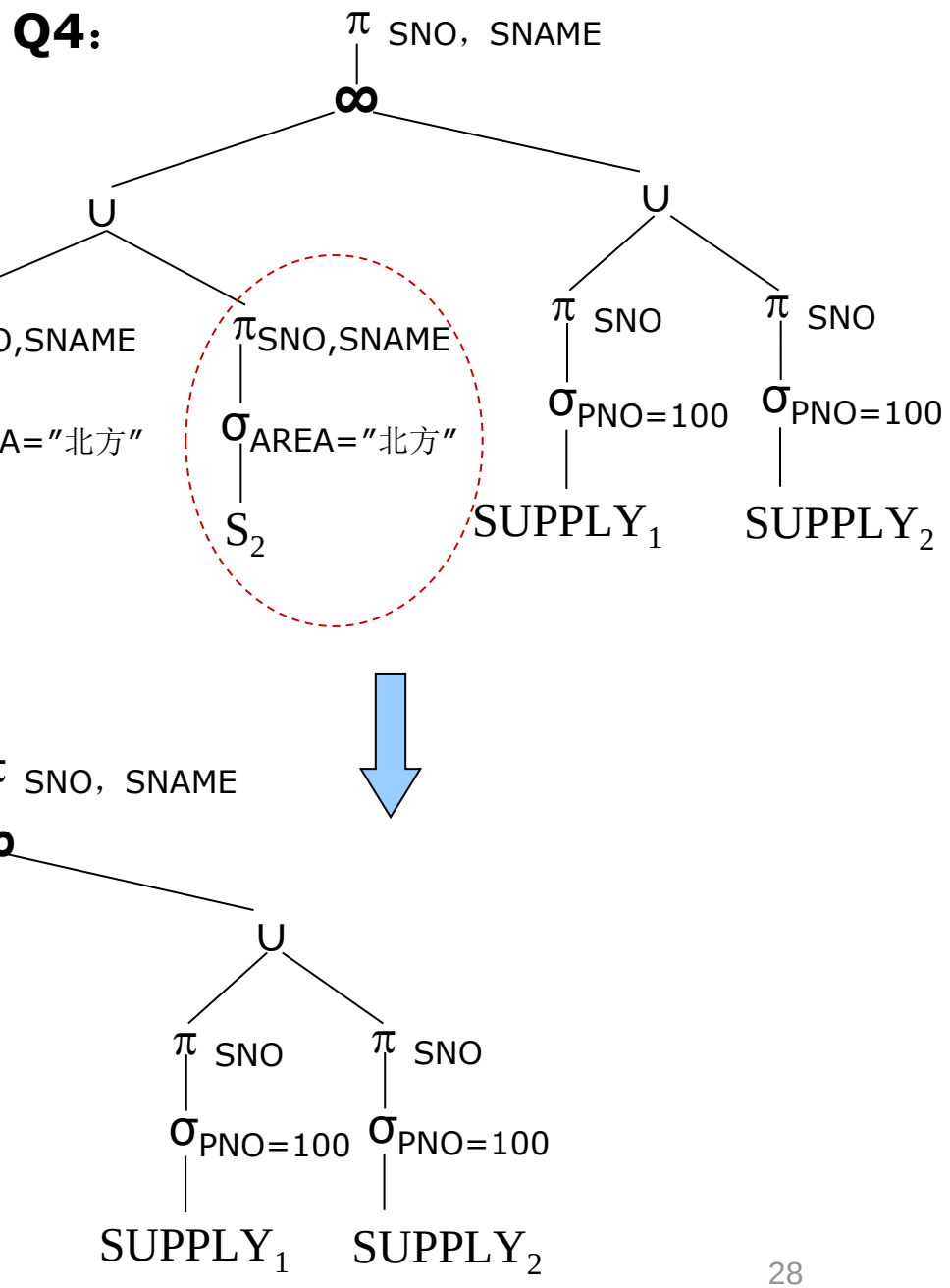
$S_2=\sigma_{AREA="南方"}$（SUPPLIER）

$SUPPLY_1=\sigma_{AREA="北方"}$（SUPPLY）

$SUPPLY_2=\sigma_{AREA="南方"}$（SUPPLY）

所以，Q4中的虚线部分为空关系，

可将其去掉，得到Q5查询树。



分布式数据库查询和优化

● 查询处理层次

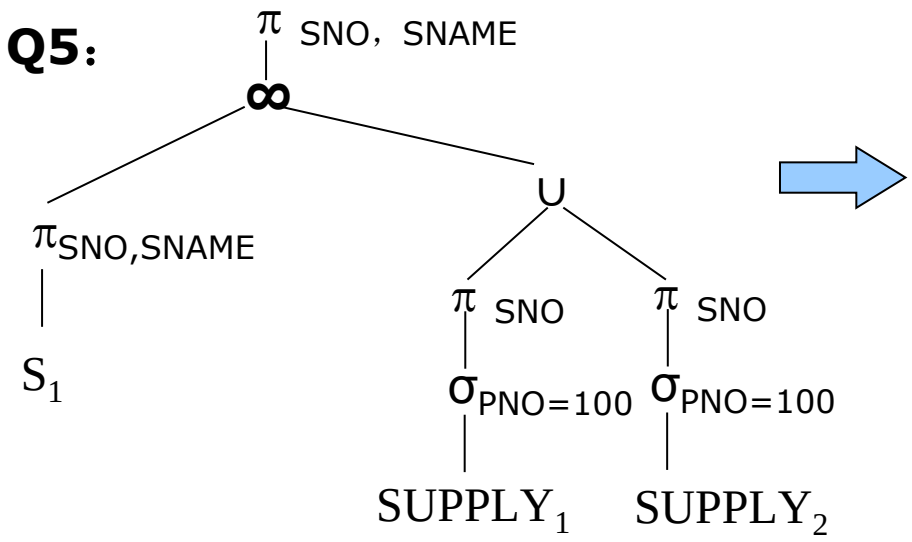
2. 数据本地化（Data Localization）

$S1=\sigma_{AREA="北方"}(SUPPLIER)$
$S2=\sigma_{AREA="南方"}(SUPPLIER)$
$SUPPLY1=\sigma_{AREA="北方"}(SUPPLY)$
$SUPPLY2=\sigma_{AREA="南方"}(SUPPLY)$

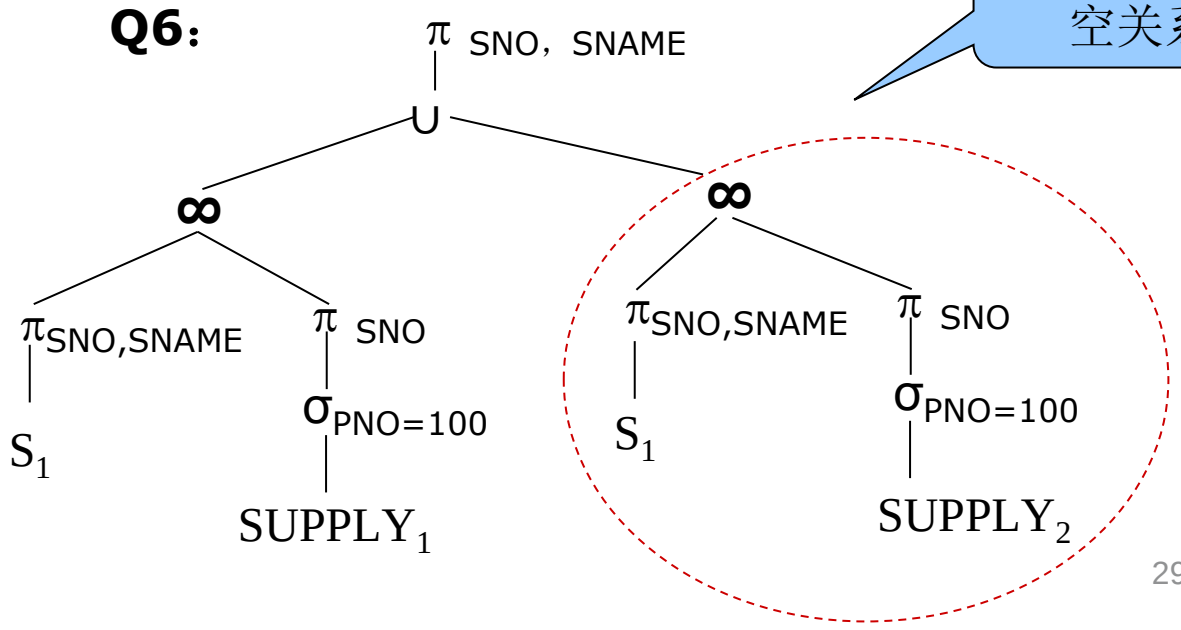
□ 片段查询优化

（3）根据准则3，将联接运算(∞)下移到并运算(∪)之前，得到Q6。

Q5：



Q6：



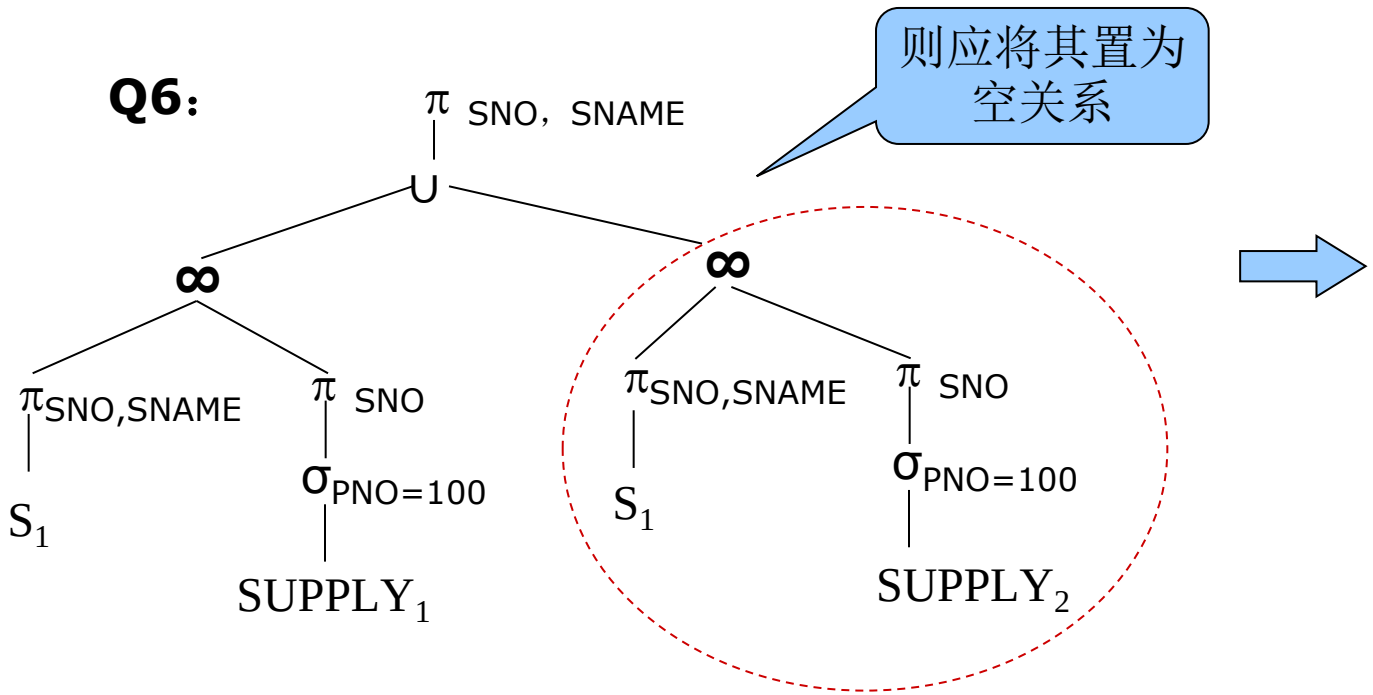
则应将其置为空关系

分布式数据库查询和优化

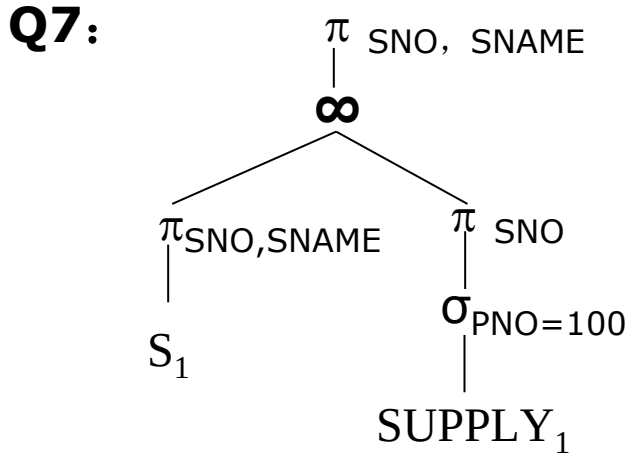
● 查询处理层次

2. 数据本地化（Data Localization）

□ 片段查询优化



Q7为化简后的最终的优化查询。



分布式数据库查询和优化

查询处理层次

例

■ 假设：一雇员关系 EMP {ENO，ENAME，BIRTH，SALARY，DNO}

✓ 其分片为：

$E1=\prod_{ENO，ENAME，BIRTH}（EMP）$

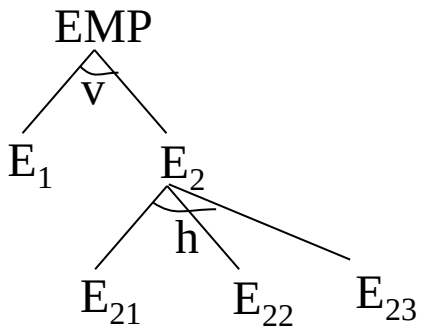
$E2=\prod_{ENO，SALARY，DNO}（EMP）$

$E21=\sigma_{Dno=201}（E2）$

$E22=\sigma_{Dno=202}（E2）$

$E23=\sigma_{Dno〈〉201\ AND\ Dno〈〉202}（E2）$

✓ 分片树：

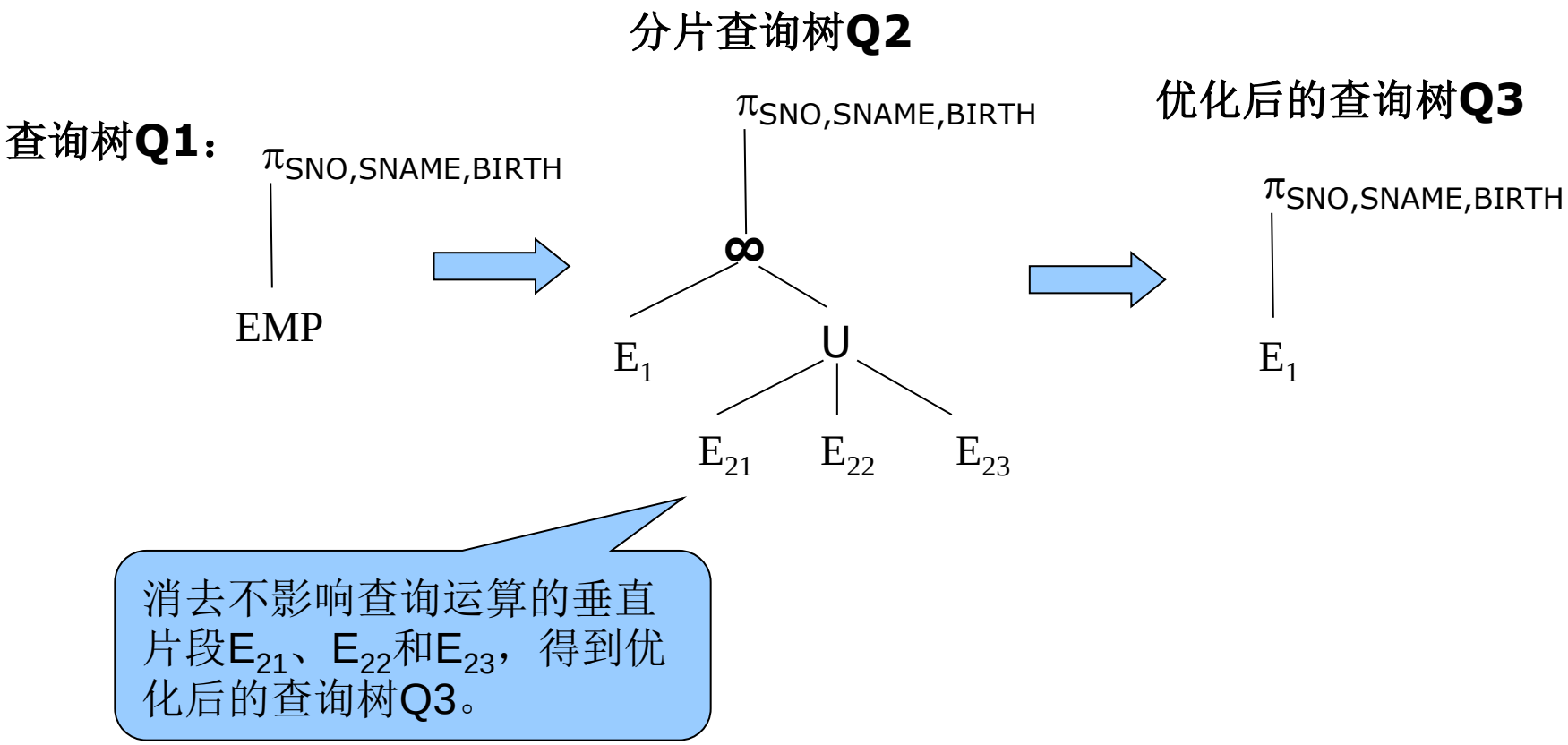


要求：查询SQL：SELECT ENO，ENAME，BIRTH

FROM EMP

分布式数据库查询和优化

查询处理层次



分布式数据库查询和优化

● 查询处理层次

□ 分布式查询处理过程

3. 全局查询优化（Global Query Optimization）

- 找接近于最优的执行策略；
- 找片段查询中最佳的操作顺序，包括通信操作；
- 需要实时定义代价函数。

4. 局部查询优化 （Local Query Optimization）

集中的系统算法.

- INGRES –动态优化（ dynamic optimization）
- System R – 基于穷举法的静态优化（static optimization based on exhaustive search）

分布式数据库查询和优化

● 查询处理层次

□ 分布式查询处理过程

3. 全局查询优化（Global Query Optimization）

- 找接近于最优的执行策略；
- 找片段查询中最佳的操作顺序，包括通信操作；
- 需要实时定义代价函数。

4. 局部查询优化 （Local Query Optimization）

集中的系统算法.

- INGRES –动态优化（ dynamic optimization）
- System R – 基于穷举法的静态优化（static optimization based on exhaustive search）

分布式数据库查询和优化

● 查询优化概述

□ 优化类型

- ■ 穷举法（**Exhaustive search**）– workable for small solution space.
- ■ 启发式（**Heuristics**） – σ，π perform first, semi-join, etc. for large solution space.

□ 优化时机

- ■ 静态的 **(Static) – 在编译时进行优化，基于统计信息进行穷举优化，优化一次，执行多次**
- ■ 动态的 **(Dynamic) –在执行中优化，准确性好，但每次执行都进行优化，代价高**

分布式数据库查询和优化

● 查询优化概述

□ 统计数据（Statistics）

元组基数, 属性值分布，关系大小等。信息提供给查询优化器并定期修改。

□ 决策场地（Decision Site）

查询优化场地可以：

- ■单场地：集中的方法， 简单，需要整个分布库的知识。
- ■分布式：涉及所有的场地协调确定调度，只需本地知识，需要协调代价。
- ■混合型： 一个场地确定全局调度，其他场地优化局部子查询。

□ 复制的片段

最小化通信代价。

□ 使用半连接

分布式数据库查询和优化

[案例] 设一供应关系数据库，含供应者和供应关系

供应者： Supplier（SNO, SNAME, AREA）

供应关系：Supply （SNO, PNO, QTY）

零件关系：Part（PNO, PNAME）

案例

```
SELECT      SNAME
FROM        SUPPLIER S，SUPPLY SP，PART P
WHERE       S.SNO=SP.SNO
AND         SP.PNO=P.PNO
AND         P.PNAME=“BOLT”
AND         S.AREA=“北方”
AND         SP.QTY>5000;
```

分布式数据库事务管理

分布式事务

分布式事务的定义

从宏观上来看，分布式事务是由一系列分布在多个场地上执行的数据库操作所组成的。

分布式事务：是指分布式数据库应用中的事务，也称为全局事务。

子事务：一个分布式事务在执行时将被分解为若干个场地上独立执行的操作序列，即一个分布式事务在某个场地上操作的集合。

显然，分布式事务是典型的嵌套事务。

分布式数据库并发控制

分布式恢复

典型分布式数据库介绍与研究展望

关于本讲内容

祝各位学习愉快！

感谢观看！

讲解人：李鸿岐