数据库系统

第9章 查询处理和查询优化 (2)

胡 敏

合肥工业大学

jsjxhumin@hfut.edu.cn

第 9 章　查询处理和查询优化

9.2 查询优化

查询优化的必要性和重要性

- RDBMS的特点
- 用户不管存取路径，仅需提出what to do
- 查询优化：关键技术、优点所在
- 影响RDBMS性能的关键因素

9.2 查询优化

■查询优化的优点

- ➢用户不必考虑how to do
- ➢系统可以比用户程序的“优化”做得更好
  - ★(1) 优化器可以从数据字典中获取许多统计信息，而用户程序则难以获得这些信息
  - ★(2)如果数据库的物理统计信息改变了，系统可以自动对查询重新优化以选择适应的执行计划。在非关系系统中必须重写程序，而重写程序在实际应用中往往是不太可能的。
- ➢(3)优化器可以考虑数百种不同的执行计划，程序员一般只能考虑有限的几种。
- ➢(4)优化器中包括了很多复杂的优化技术，这些优化技术往往只有最好的程序员才能掌握。系统的自动优化相当于使得所有人都拥有这些优化技术

9.2 查询优化

■查询优化的代价模型

通过某种代价模型计算出各种查询执行方案的执行代价，然后选取代价最小的执行。

磁盘存取块数--I/O代价、处理机时间--CPU代价、查询的内存开销、通信代价

➢在集中式数据库中

总代价=I/O代价 + CPU代价

在多用户环境下：

总代价=I/O代价 + CPU代价+内存代价

➢在分布式数据库中

总代价=I/O代价 + CPU代价+ 内存代价 + 通信代价

■ 查询优化的总目标：

选择有效的大策略，求得给定关系表达式的值，使得查询代价最小（较小）

9.2 .2 查询优化的必要性（1）

例：求选修了课程号为2的学生的姓名

```
SELECT  Student.Sname
FROM    Student, SC
WHERE   Student.Sno=SC.Sno AND    SC.Cno='2';
```

假设1：外存：

Student:1000条,SC:10000条, 选修2号课程:50条

假设2：7个内存块

Student：5块，SC：1块，中间结果表：1块

块大小：一块装10个Student元组（或10个student与SC笛卡尔积元组），或50个SC

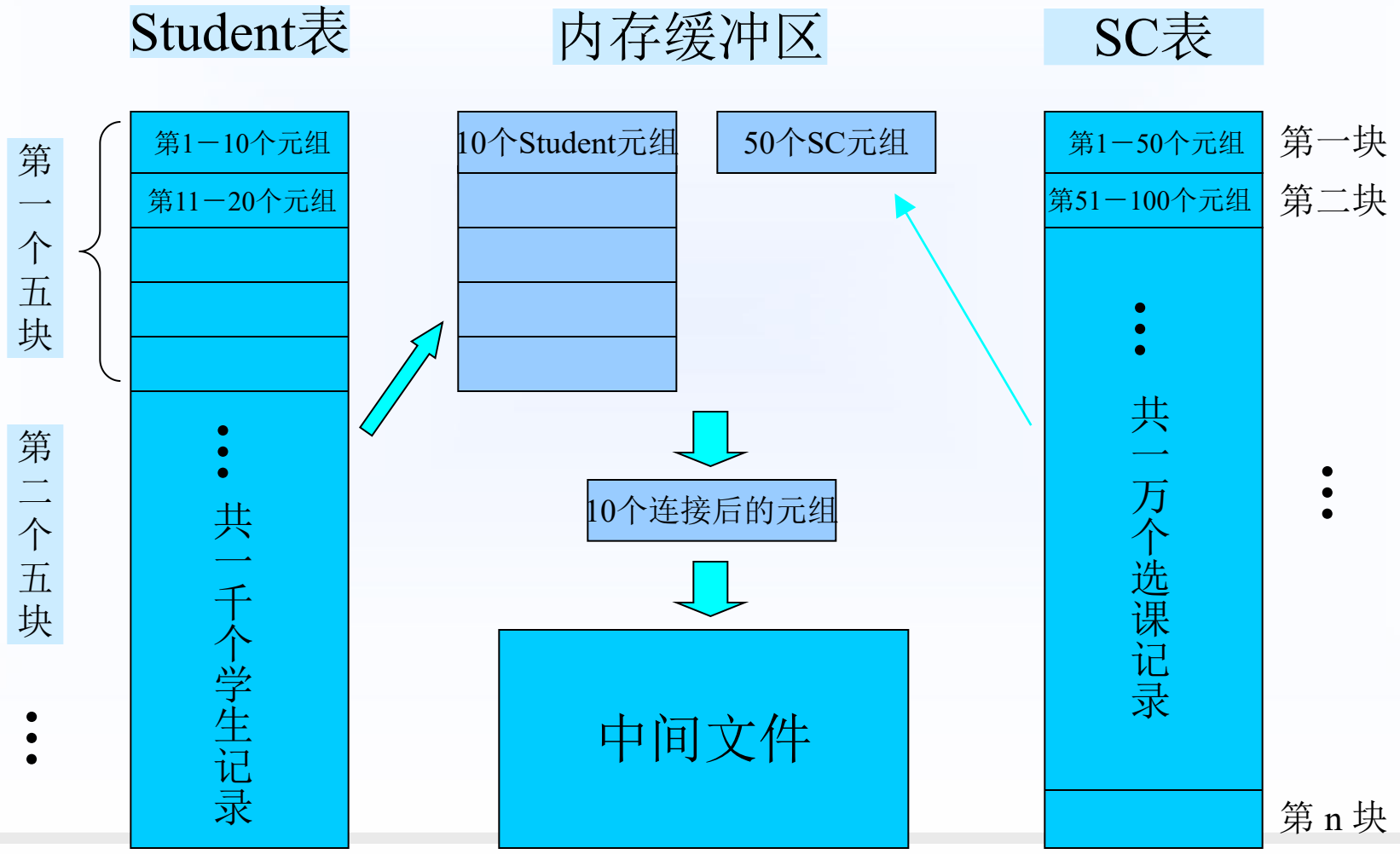
假设3：读写速度：20块/秒

假设4：连接方法：基于数据块的嵌套循环法

student表需内存块：100
SC表需内存块：200
每换一批外表数据，内表要重新装一遍，所以要尽可能分较多内存给外表
--student表做外表、分5块给student表

9.2 .2 查询优化的必要性（2）

- 读取Student和SC表的策略



9.2 查询优化-必要性(3)

- 不同的执行策略,考虑I/O时间
- 执行策略

$$Q1 = \Pi_{S\,name}(\sigma_{Student.Sno=SC.Sno \wedge SC.Cno='2'}(Student \times SC))$$

$$Q2 = \Pi_{S\,name}(\sigma_{SC.Cno='2'}(Student \bowtie SC))$$

$$Q3 = \Pi_{Sname}(Student \bowtie \sigma_{SC.Cno='2'}(SC))$$

$$Q4 = \Pi_{S\,name}(Student \bowtie \sigma_{SC.Cno='2'}(SC)),$$

假设SC表在Cno上有索引，Student表在Sno上有索引

9.2 查询优化-必要性(3)

- 执行策略1

$Q1 = \Pi_{S\,name}(\sigma_{Student.Sno=SC.Sno \wedge SC.Cno='2'}(Student \times SC))$

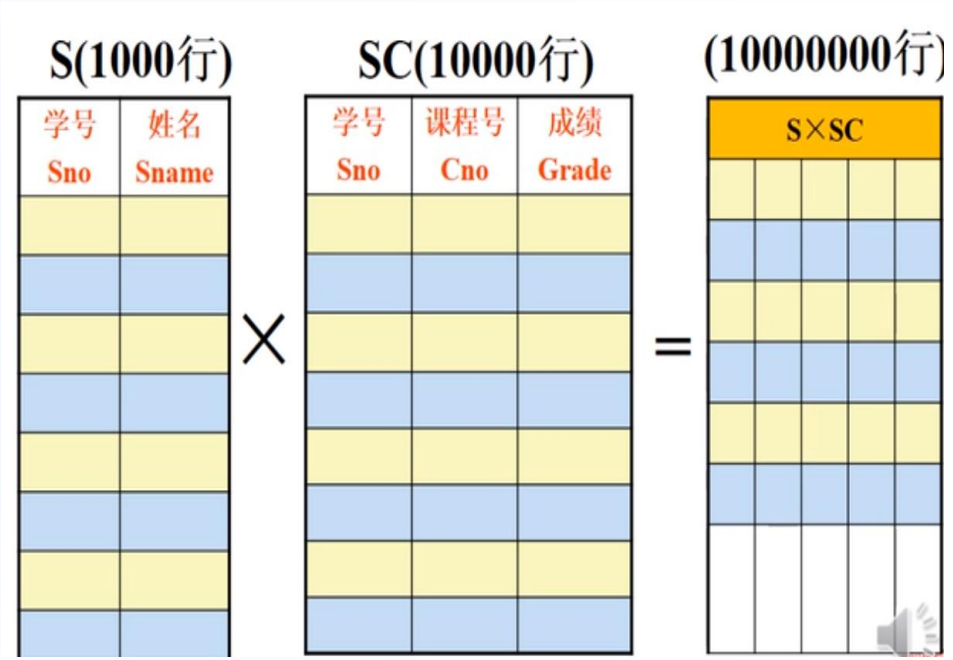
① Student×SC

读取总块数= 读Student表块数 + 读SC表遍数 *每遍块数

=1000/10+(1000/(10×5)) ×(10000/50)

=100+20×200=4100

读数据时间=4100/20=205秒



中间结果大小 = 1000*10000 = 10^7 (1千万条元组)

写中间结果时间 = 10000000/10/20 = 50000秒

②σ 读数据时间 = 50000秒

③Π 总时间 =205＋50000＋50000 = 100205（秒）

= 27.8小时

9.2 查询优化-必要性(4)

■执行策略2

Q2＝ $\Pi_{S\,name}(\sigma_{SC.Cno='2'}(Student \bowtie SC))$

①⋈

读取总块数= 4100块

读数据时间=4100/20=205秒

（自然连接）中间结果大小=10000 （减少1000倍）

写中间结果时间=10000/10/20=50秒

②б 读中间结果

读数据时间=50秒

③Π 50个元组

时间忽略不计

总时间＝205＋50＋50＝305（秒）=5.08分

S(1000行)

学号 Sno	姓名 Sname
05001	李勇
05002	刘晨

⋈

SC(10000行)

学号 Sno	课程号 Cno	成绩 Grade
05001	1	92
05001	2	85
05001	3	88
05002	2	90

＝ 10000行

9.2 查询优化-必要性(5)

■执行策略3

Q 3＝ $\Pi_{Sname}(Student \bowtie \sigma_{SC.Cno='2'}(SC))$

①б

读SC表总块数= 10000/50=200块

读数据时间=200/20=10秒

中间结果大小=50条 不必写入外存

② ⋈

读Student表总块数= 1000/10=100块

读数据时间=100/20=5秒

③ Π

总时间＝10＋5＝15（秒）

S(1000行)

学号 Sno	姓名 Sname
05001	李勇
05002	刘晨

⋈

$\sigma_{SC.Cno='2'}(SC)$ (50行)

学号 Sno	课程号 Cno	成绩 Grade
05001	2	85
05002	2	90

＝ 50行

9.2 查询优化-必要性(6)

■执行策略4

Q4＝ $\Pi_{S\,name}$(Student ⋈$\sigma_{SC.Cno='2'}$ (SC))

假设SC表在Cno上有索引，Student表在Sno上有索引

①б

读SC表索引=

读SC表总块数= 50/50=1块

读数据时间

中间结果大小=50条 不必写入外存

② ⋈

读Student表索引=

读Student表总块数= 50/10=5块

读数据时间

③ Π

总时间<10秒

S(1000行)

学号 Sno	姓名 Sname
05001	李勇
05002	刘晨

⋈

$\sigma_{SC.Cno='2'}$(SC) (50行)

学号 Sno	课程号 Cno	成绩 Grade
05001	1	92
05001	2	85
05002	2	90
05001	3	88

＝ 50行

图7－3

9.2 查询优化-必要性(7)

■把代数表达式Q1变换为Q2、 Q3：代数优化

- ✓ 选择和连接操作时，先做选择操作
- ✓选择与其后面的连接同时做；

■在Q3=》Q4：物理优化

SC表的选择操作算法有全表扫描和索引扫描2种方法，经过初步估算，索引扫描方法较优

- I/O对比

请思考：在同一环境下，以SC为外表，会有什么不同？

第 9 章　查询处理和查询优化

9.3 代数优化

- 代数优化对查询进行等效变换，以减少执行开销。

- 代数优化的原则是尽量减小查询过程中间结果的大小。
- 选择、投影操作通常能够有效地减小关系的大小。
- 连接、笛卡尔乘积和并操作容易生成较大的查询中间结果。

因此，先做选择、投影；先做小关系间的连接，再做大关系的连接；甚至需要先找出查询中的公共表达式，以避免重复运算。

9.3.1 代数优化-等价变换

- 代数优化策略：通过对关系代数表达式的等价变换来提高查询效率

- 关系代数表达式的等价：指用相同的关系代替两个表达式中相应的关系所得到的结果是相同的

- 两个关系表达式*E1*和*E2*是等价的，可记为*E1*≡*E2*

9.3.1 代数优化-等价变换规则（1）

设E1、E2等是关系代数表达式，F是条件表达式

- I. 连接、笛卡尔积交换律
- $E1 \times E2 \equiv E2 \times E1$
- $E1 \bowtie E2 \equiv E2 \bowtie E1$
- $E1 \bowtie_F E2 \equiv E2 \bowtie_F E1$

9.3.1 代数优化-等价变换规则（2）

- 2. 连接、笛卡尔积的结合律

$$(E1 \times E2) \times E3 \equiv E1 \times (E2 \times E3)$$

$$(E1 \bowtie E2) \bowtie E3 \equiv E1 \bowtie (E2 \bowtie E3)$$

$$(E1 \underset{F}{\bowtie} E2) \underset{F}{\bowtie} E3 \equiv E1 \underset{F}{\bowtie} (E2 \underset{F}{\bowtie} E3)$$

9.3.1 代数优化-等价变换规则（3）

- 3. 投影的串接定律

$$\pi_{A1,A2,\cdots,An}(\pi_{B1,B2,\cdots,Bm}(E)) \equiv \pi_{A1,A2,\cdots,An}(E)$$

假设：

1) E是关系代数表达式

2) $A_i(i=1，2，…，n)$, $B_j(j=l，2，…，m)$是属性名

3)$\{A_1, A_2, \ldots, A_n\}$构成$\{B_l，B_2，…，B_m\}$的子集

9.3.1 代数优化-等价变换规则（4）

- 4. 选择的串接定律

$$б_{F1}（б_{F2}（E））\equiv б_{F1 \wedge F2}(E)$$

  - 选择的串接律说明选择条件可以合并
  - 这样一次就可检查全部条件

9.3.1 代数优化-等价变换规则（5）

■ 5. 选择与投影操作的交换律

(1)假设: 选择条件F只涉及属性A1，…，An

$$б_F(\pi_{A1,A2,\cdots,An}(E)) \equiv \pi_{A1,A2,\cdots,An}(б_F(E))$$

(2)假设: F中有不属于A1, …,An的属性B1,…,Bm

$$\pi_{A1,A2,\cdots,An}(б_F(E)) \equiv \pi_{A1,A2,\cdots,An}(б_F(\pi_{A1,A2,\cdots,An,B1,B2,\cdots,Bm}(E)))$$

9.3.1 代数优化-等价变换规则（6）

6. 选择与笛卡尔积的交换律

(1) 假设：F中涉及的属性都是E1中的属性

$$б_F(E1 \times E2) \equiv б_F(E1) \times E2$$

(2) 假设：$F=F1 \wedge F2$，并且F1只涉及E1中的属性，F2只涉及E2中的属性

则由上面的等价变换规则1，4，6可推出：

$$б_F(E1 \times E2) \equiv б_{F1}(E1) \times б_{F2}(E2)$$

9.3.1 代数优化-等价变换规则（7）

- 7. 选择与并的分配律
- 8. 选择与差运算的分配律
- 9. 选择对自然连接的分配律
- 10. 投影与笛卡尔积的分配律
- 11. 投影与并的分配律

9.3.2 查询树的启发式优化 （1）

■典型的启发式规则：

➢1. 选择运算应尽可能先做！！！

减少中间结果集

➢2. 把投影运算和选择运算同时进行

如有若干投影和选择运算，并且它们都对同一个关系操作，则可以在扫描此关系的同时完成所有的这些运算以避免重复扫描关系。

➢ 3. 把投影同其前或其后的双目（二元）运算结合起来

避免重复扫描关系

9.3.2 查询树的启发式优化 （2）

■典型的启发式规则：

➢4. 把某些选择同在它前面要执行的笛卡尔积结合起来成为一个连接运算

减少中间结果、减少扫描次数

$$\sigma_{S.sno=SC.sno}(S \times SC) \Rightarrow S \bowtie SC$$

➢5. 找出公共子表达式

如果这种重复出现的子表达式的结果不是很大的关系，并且从外存中读入这个关系比计算该子表达式的时间少得多，则先计算一次公共子表达式并把结果写入中间文件。

当查询的是视图时，定义视图的表达式就是公共子表达式的情况。

保存中间结果，避免重复计算

9.3.2查询树的启发式优化 （3）

■优化算法

算法：关系表达式的优化

输入：一个关系表达式的语法树。

输出：计算该表达式的程序。

方法：

（1）分解选择运算

利用规则4 ～8(选择的串接)把形如$б_{F1} \wedge_{F2} \wedge \ldots \wedge_{Fn} (E)$变换为 $б_{F1}(б_{F2}(\ldots(б_{Fn}(E))\ldots))$

（2）通过交换选择运算，将其尽可能移到树的叶端

（3）通过交换投影运算，将其尽可能移到叶端

注意： 等价变换规则3使一些投影消失

规则5把一个投影分裂为两个，其中一个有可能被移向树的叶端

目的：尽早丢弃既不是投影又不是连接条件的多余属性

9.3.2查询树的启发式优化（4）

■优化算法

（4）合并串接的选择和投影，以便能同时执行或在一次扫描中完成

- 利用规则3～5把选择和投影的串接合并成单个选择、单个投影或一个选择后跟一个投影。
- 使多个选择或投影能同时执行，或在一次扫描中全部完成。
- 尽管这种变换似乎违背“投影尽可能早做”的原则，但这样可以减少扫描遍数，效率更高。

9.3.2查询树的启发式优化 (5)

（5）对内结点分组

每一双目运算(⋈，×，∪，-)和它所有的直接祖先为一组(这些直接祖先是б，π运算)。

如果其后代直到叶子全是单目运算，则也将它们并入该组，但当双目运算是笛卡尔积(×)，而且其后的选择不能与它结合为等值连接时除外。把这些单目运算单独分为一组。

（6）生成程序

生成一个程序，每组结点的计算是程序中的一步。

各步的顺序是任意的，只要保证任何一组的计算不会在它的后代组之前计算。

优化的一般步骤

1. 把查询转换成某种内部表示
2. 代数优化：把语法树转换成标准（优化）形式
3. 物理优化：选择低层的存取路径
4. 生成查询计划，选择代价最小的

优化的一般步骤-例 (1)

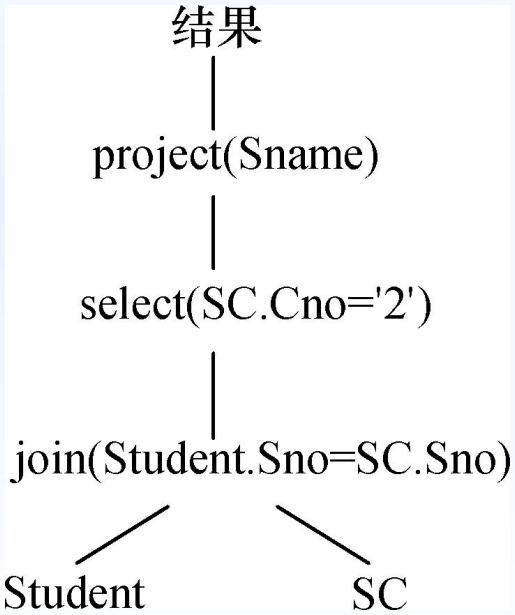
例：求选修了课程号为2的学生姓名

```
SELECT  Student.Sname
 FROM    Student, SC
 WHERE  Student.Sno=SC.Sno
 AND     SC.Cno='2';
```

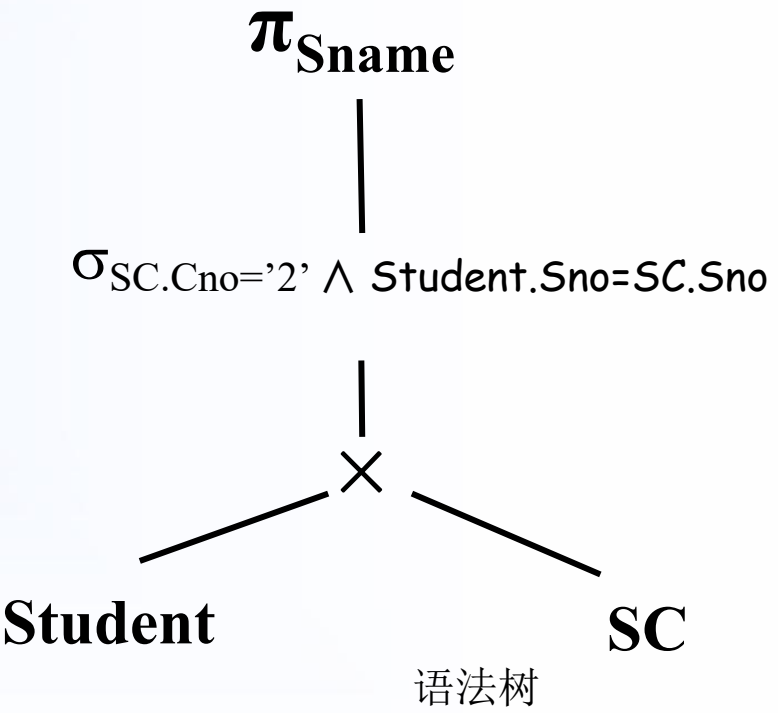
（1）把查询转换成某种内部表示（语法树）

建立语法树的规则：

对一个关系表达式进行语法分析，将关系作为叶子节点，而对关系的操作作为非叶子节点。



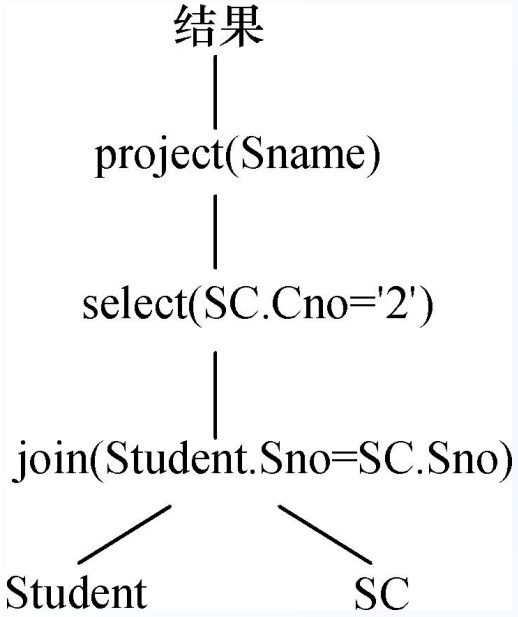
查询树



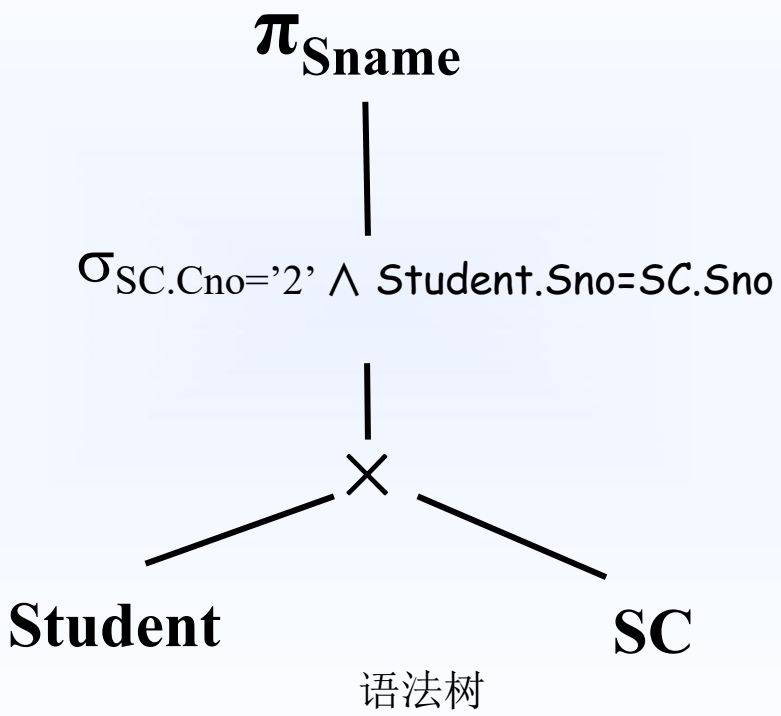
语法树

优化的一般步骤-例 (2)

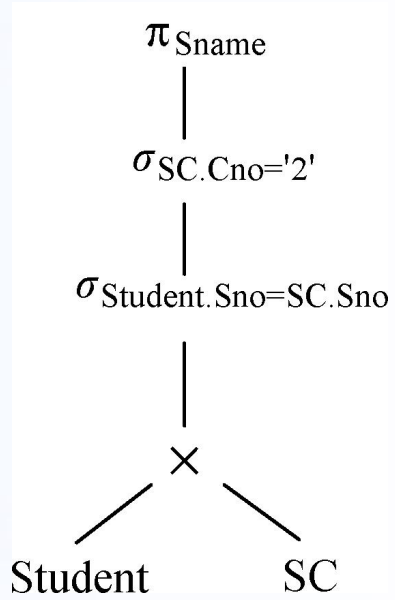
（2）利用优化算法把语法树转换成标准（优化）形式



查询树



语法树

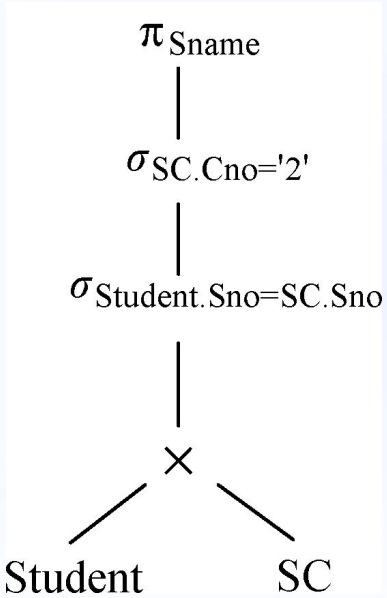


关系代数语法树

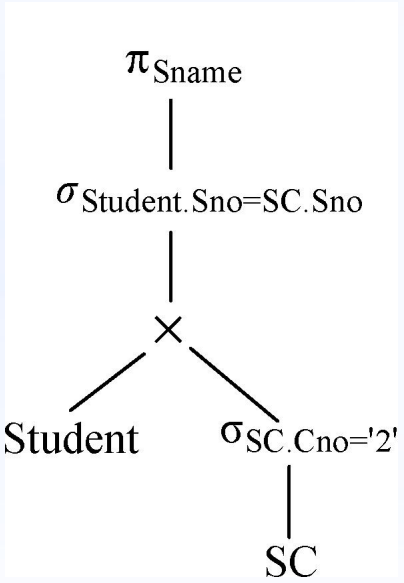
优化的一般步骤-例 (3)

(3) 对查询树进行优化

利用规则4、6把选择$\sigma_{SC.Cno='2'}$移到叶端，查询树便转换成下图所示的优化的查询树。这就是上节中Q3的查询树表示



关系代数语法树



优化后的查询树

优化的一般步骤-例 (1)

例：选修了数据库课程的女生学号与姓名

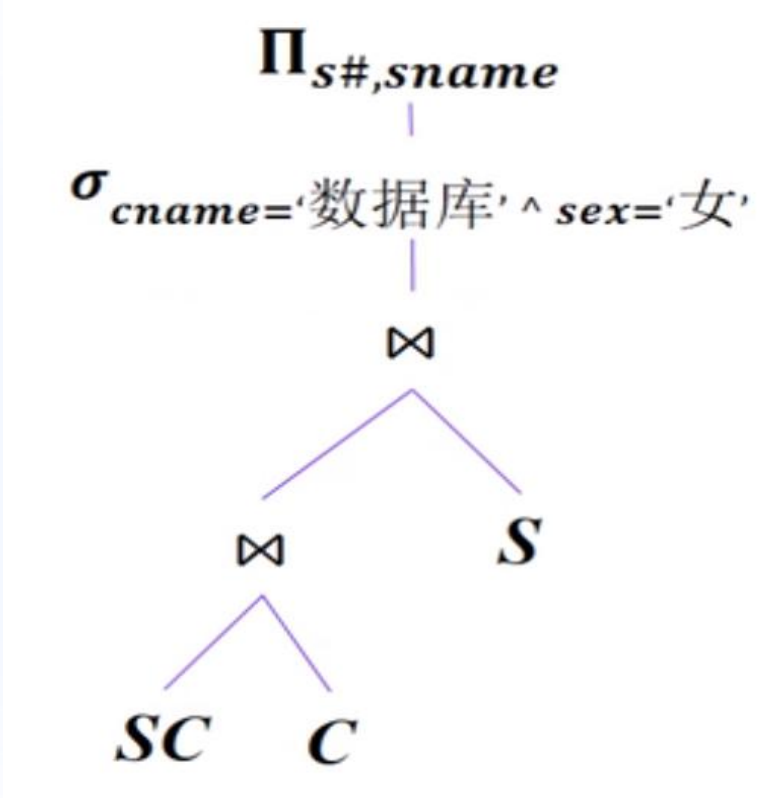
S(S#, Sname, sex, age, dept)

C(C#, Cname)

SC(S#, C#, Grade)

该查询的关系代数表达式如下：

$$\Pi_{S\#,Sname}(\sigma_{cname='数据库' \wedge sex='女'}(SC \bowtie C \bowtie S))$$



语法树

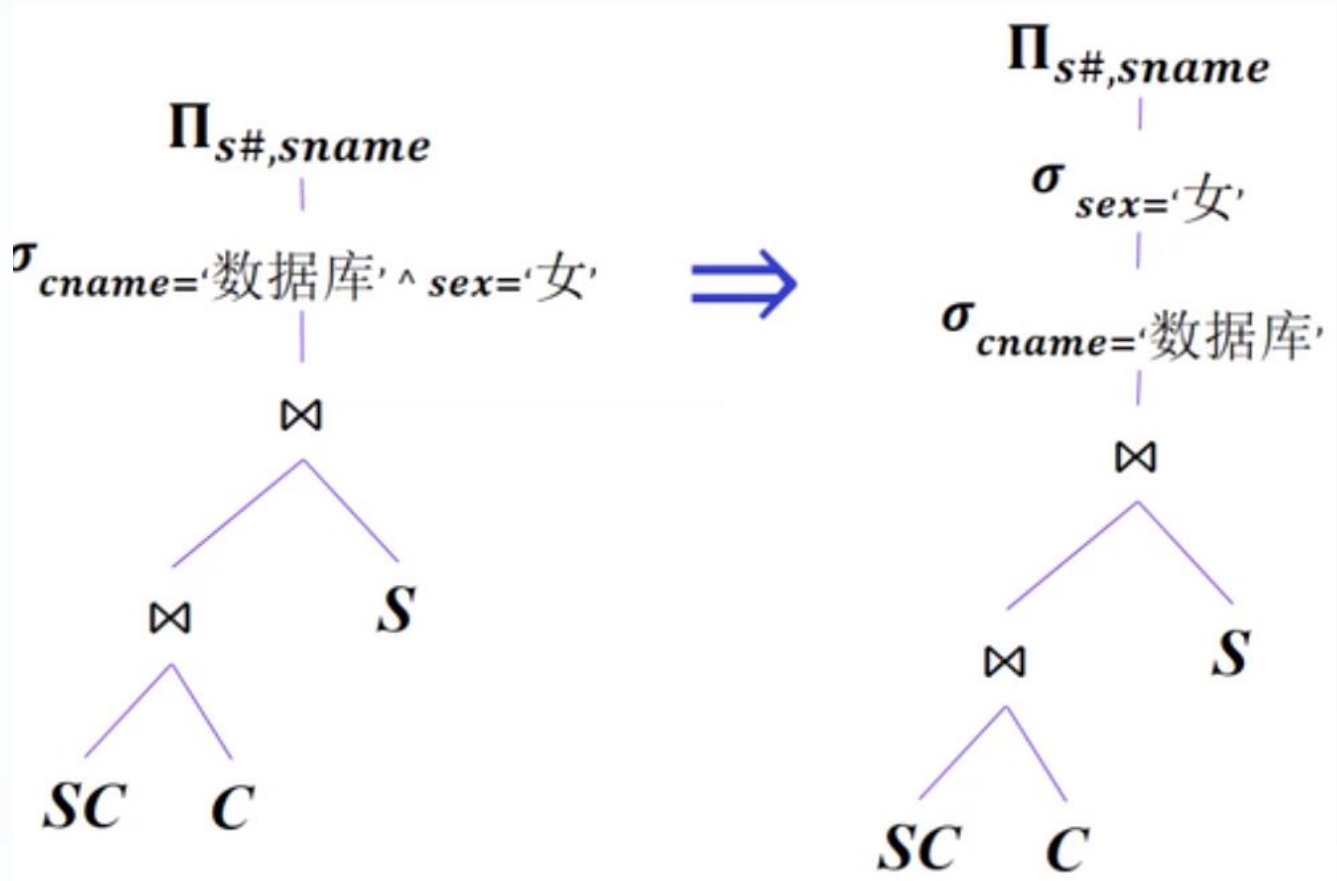
（1）把查询转换成某种内部表示（语法树）

建立语法树的规则：

对一个关系表达式进行语法分析，将关系作为叶子节点，而对关系的操作作为非叶子节点。

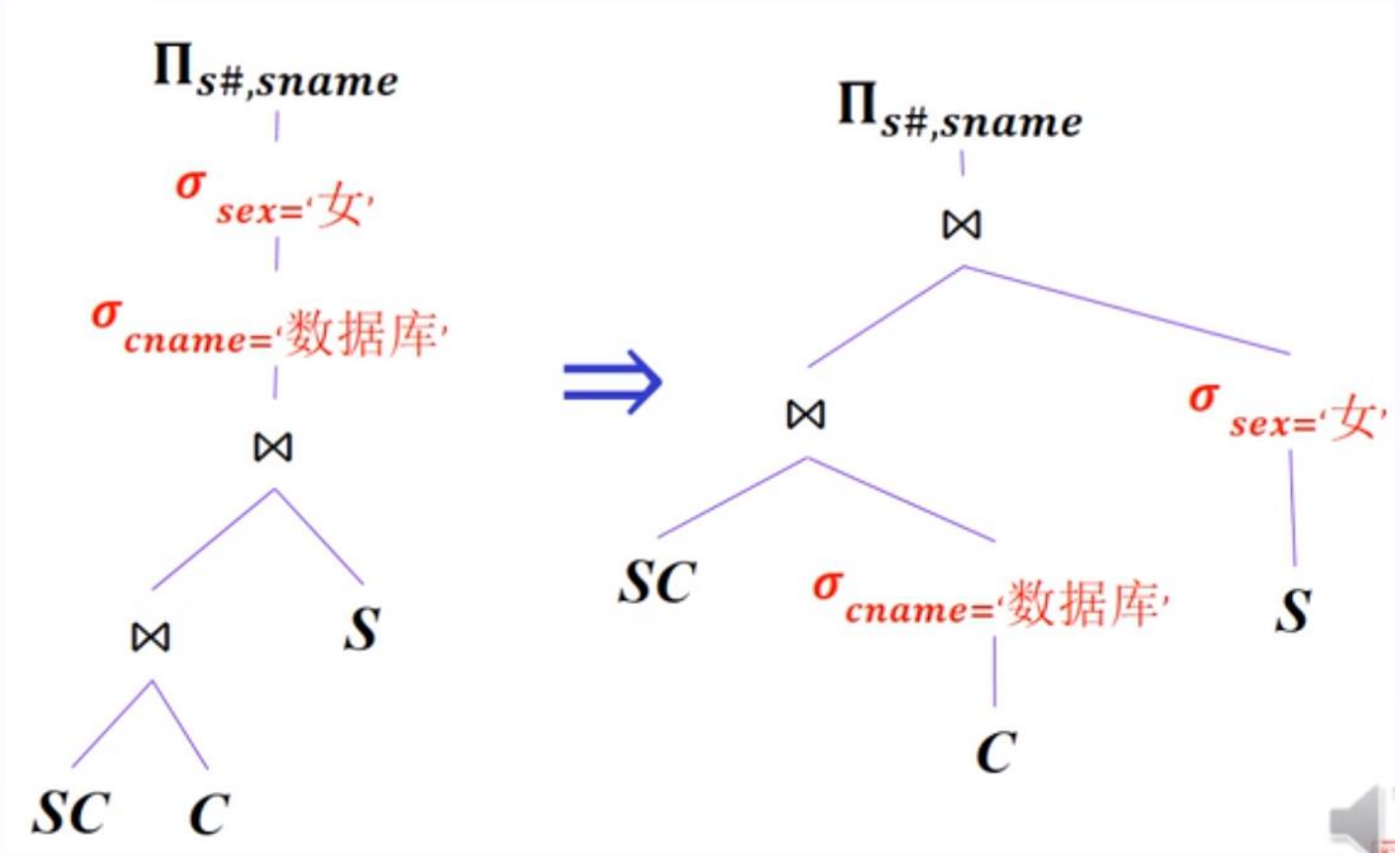
优化的一般步骤-例 (2)

（2）分解选择运算



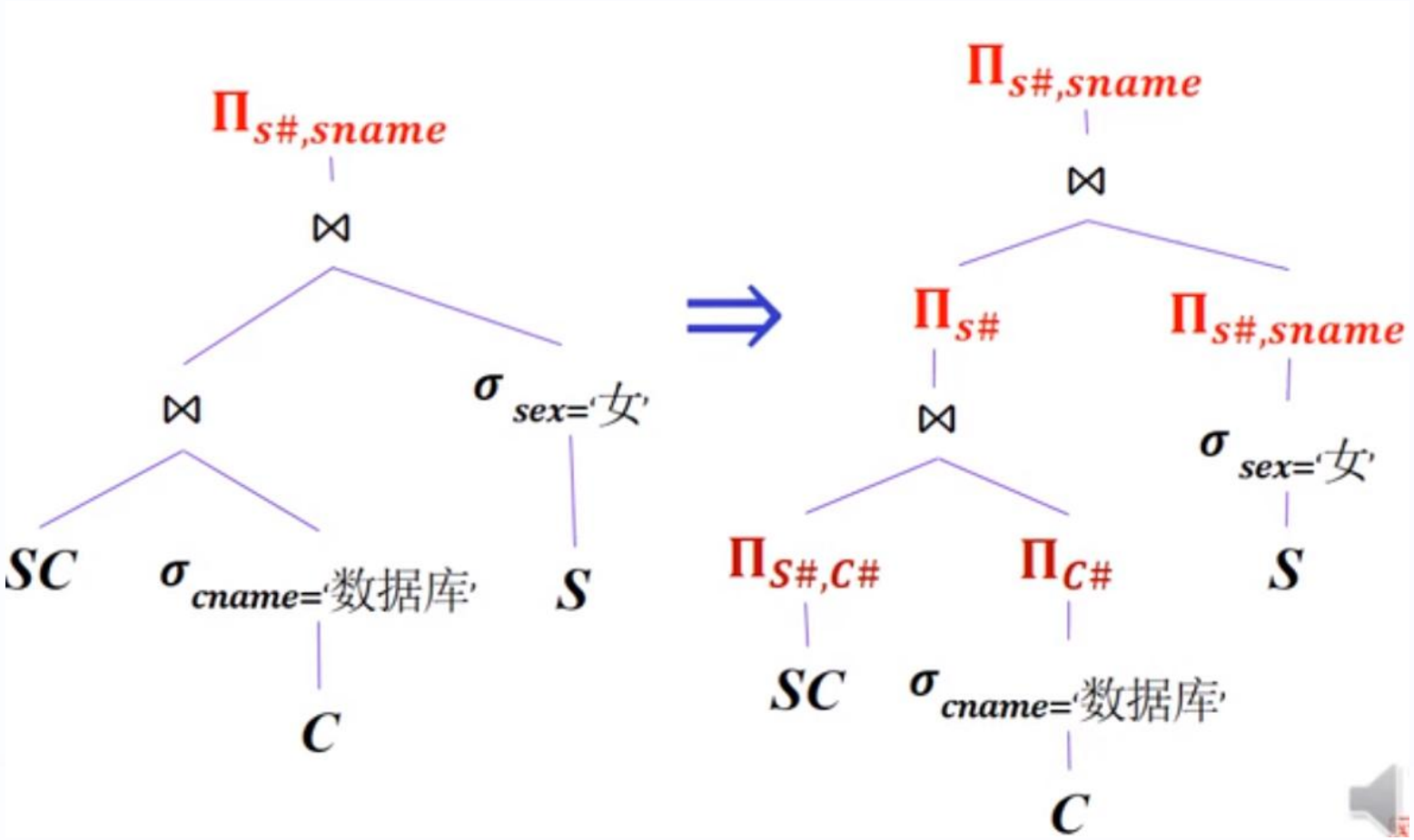
优化的一般步骤-例 (2)

（3）把选择运算尽可能移到树的叶子端



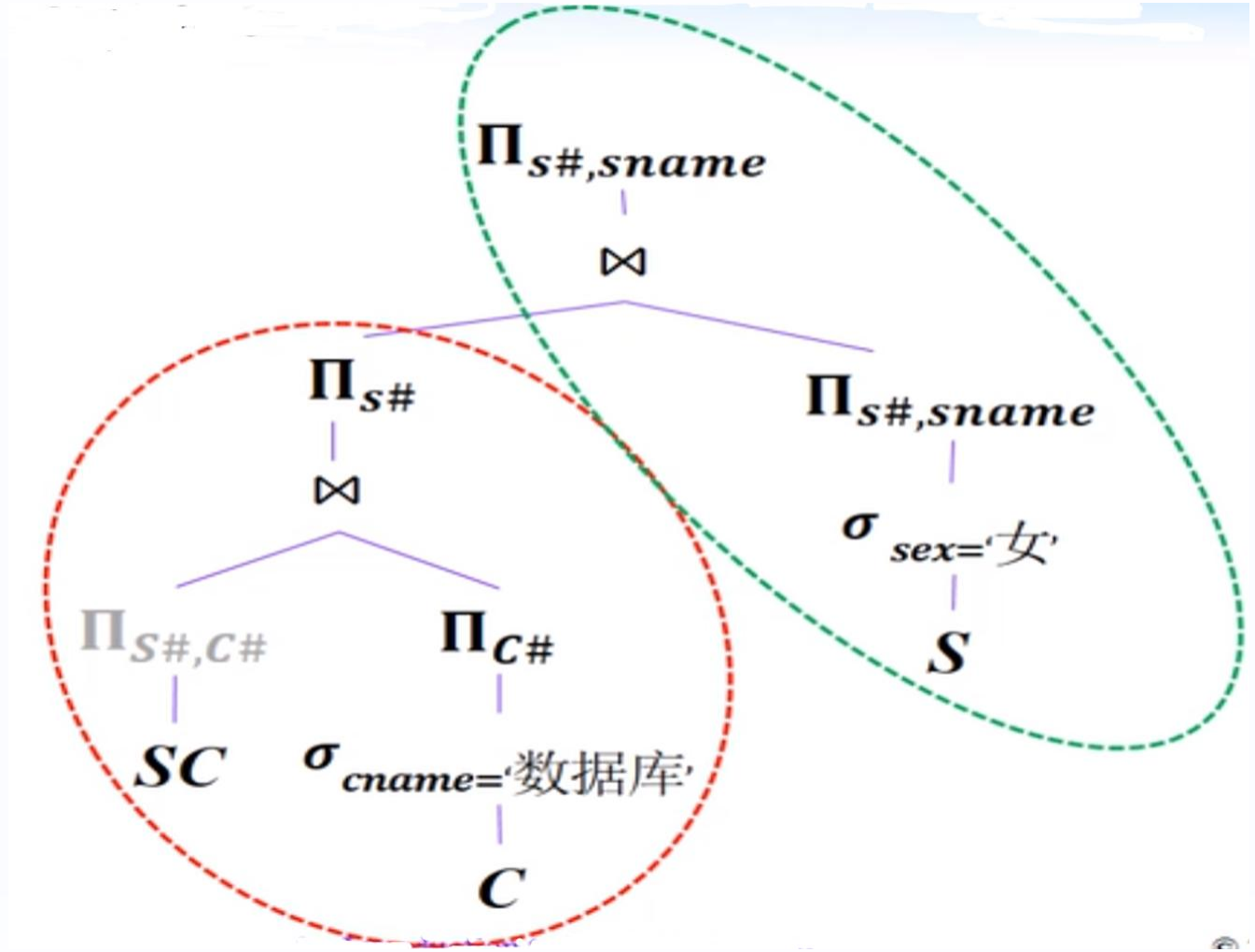
优化的一般步骤-例 (2)

（4）把投影运算尽可能移到树的叶子端



优化的一般步骤-例 (2)

（5）内节点分组



第 9 章　查询处理和查询优化

9.4 物理优化(1)

- 代数优化改变查询语句中操作的次序和组合，不涉及底层的存取路径
- 对于一个查询语句有许多存取方案，它们的执行效率不同， 仅仅进行代数优化是不够的
- 物理优化就是要选择高效合理的操作算法或存取路径，求得优化的查询计划

9.4 物理优化(2)

- 选择的方法：

基于规则的启发式优化

基于代价估算的优化

两者结合的优化方法：常常先用启发式规则，选取若干较优的候选取方案，减少代价估算的工作量；然后分别计算这些候选方案的执行代价，较快地选出最终的优化方案。

9.4.1　基于启发式规则的存取路径选择优化

- 一、　选择操作的启发式规则
- 二、　连接操作的启发式规则

基于启发式规则的存取路径选择优化(续)

一、选择操作的启发式规则：

- 对于小关系，使用全表顺序扫描，即使选择列上有索引
- 对于大关系，启发式规则有：
  - ✓ 对于选择条件是主码＝值的查询
    - 查询结果最多是一个元组，可以选择主码索引
    - 一般的RDBMS会自动建立主码索引。
- 对于选择条件是非主属性＝值的查询，并且选择列上有索引
  - ✓ 要估算查询结果的元组数目
    - 如果比例较小(<10%)可以使用索引扫描方法
    - 否则还是使用全表顺序扫描

基于启发式规则的存取路径选择优化(续)

- 一、选择操作的启发式规则：
  - 对于用AND连接的合取选择条件
    - ✓如果有涉及这些属性的组合索引
      - 有组合索引？优先采用组合索引扫描方法
      - 只有部分索引？索引+扫描筛选后的结果
    - ✓如果某些属性上有一般的索引
      - 则可以用［例1-C4］中介绍的索引扫描方法
      - 否则使用全表顺序扫描。
  - 对于用OR连接的析取选择条件，一般使用全表顺序扫描

基于启发式规则的存取路径选择优化(续)

二、 连接操作的启发式规则：

1. 如果2个表都已经按照连接属性排序

   选用排序-合并方法

2. 如果一个表在连接属性上有索引

   选用索引连接方法

3. 如果上面2个规则都不适用，其中一个表较小

   选用Hash join方法

基于启发式规则的存取路径选择优化(续)

二、 连接操作的启发式规则：

4. 可以选用嵌套循环方法，并选择其中较小的表，确切地讲是占用的块数(b)较少的表，作为外表(外循环的表) 。

R: Br块; S: Bs块

内存缓冲区块数为K:

外表：K-2块； 内表：1块；中间结果集:1块

若R为外表，read的总块数 $= Br + \frac{Br}{K-2} \cdot Bs$

若S为外表，read的总块数 $= Bs + \frac{Bs}{K-2} \cdot Br$

∴ 选块数较少的表做外表，能减少磁盘I/O的总代价。

思考：

$K \geq min(Br, Bs) + 2$

分配策略和代价公式?

9.4.2 基于代价的优化

- 启发式规则优化是定性的选择，适合解释执行的系统
  - 解释执行的系统，优化开销包含在查询总开销之中
- 编译执行的系统中查询优化和查询执行是分开的
  - 可以采用精细复杂一些的基于代价的优化方法
    - 根据数据字典中的统计信息
    - 计算各种操作算法的执行代价，从中选择最小的

9.5 小结

- 查询处理是RDBMS的核心，查询优化技术是查询处理的关键技术
- 本章讲解的优化方法
  - 启发式代数优化
  - 基于规则的存取路径优化
  - 基于代价的优化
- 本章的目的：希望读者掌握查询优化方法的概念和技术、比较复杂的查询，尤其是涉及连接和嵌套的查询
  - 不要把优化的任务全部放在RDBMS上，应该找出RDBMS的优化规律，以写出适合RDBMS自动优化的SQL语句
- 对于RDBMS不能优化的查询需要重写查询语句，进行手工调整以优化性能

作业

作业：P290　2、3、5

课后练习：

设有学生关系S(Sno,Sname,Sage,Ssex)

课程关系C(Cno,Cname,Tname)

学习关系SC(Sno,Cno,grade)

查询学习刘红老师课程的所有女同学的学号和姓名。要求画出语法树并优化。

下课了。。

休息一会儿。。。