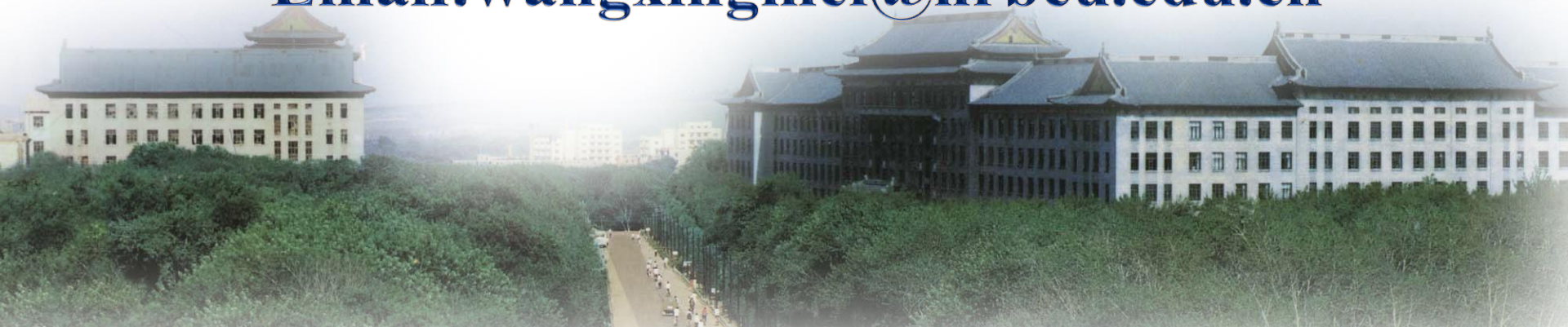
数据库原理

王兴梅
计算机科学与技术学院
Email:wangxingmei@hrbeu.edu.cn



第九章 关系系统及其查询优化

9.1 关系数据库系统的查询处理

9.2 关系数据库系统的查询优化

9.3 代数优化

9.4 物理优化

9.1 关系数据库系统的查询处理

- 9.1.1 查询处理步骤
- 9.1.2 实现查询操作的算法示例

9.1.1 查询处理步骤

- 关系数据库管理系统查询处理阶段：
  - 1. 查询分析
  - 2. 查询检查
  - 3. 查询优化
  - 4. 查询执行

1.查询分析

- 查询分析的任务：对查询语句进行扫描、词法分析和语法分析
  - 词法分析：从查询语句中识别出正确的语言符号
  - 语法分析：进行语法检查

2.查询检查

- 根据数据字典中有关的模式定义检查语句中的数据库对象，如关系名、属性名是否存在和有效。
- 合法权检查
- 完整性检查
- 视图转换

3.查询优化

- 查询优化：选择一个高效执行的查询处理策略
- 查询优化分类：
  - 代数优化/逻辑优化：指关系代数表达式的优化
  - 物理优化：指存取路径和底层操作算法的选择
- 查询优化的选择依据：
  - 基于规则
  - 基于代价
  - 基于语义

4.查询执行

- 依据优化器得到的执行策略生成查询执行计划
- 代码生成器（code generator）生成执行查询计划的代码
- 两种执行方法
  - 自顶向下
  - 自底向上

9.1关系数据库系统的查询处理

- 9.1.1查询处理步骤
- 9.1.2 实现查询操作的算法示例

9.1.2 实现查询操作的算法示例

- 1.选择操作的实现
- 2.连接操作的实现

1.选择操作的实现

- 选择操作典型实现方法：
- （1）全表扫描方法
  - 对查询的基本表顺序扫描，逐一检查每个元组是否满足选择条件，把符合条件的元组作为结果输出
  - 适合小规模表
- （2）索引扫描方法
  - 通过索引先找到满足条件的元组主码，再通过元组指针直接在基本表中找到元组
  - 适合选择条件中的属性上有索引（B+树索引）

2.连接操作的实现

- 连接操作是查询处理中最耗时的操作之一
- 本节只讨论等值连接（或自然连接）最常见的实现算法。
  - 嵌套循环算法
  - 排序-合并算法
  - 索引连接算法
  - Hash join 算法

第九章 关系系统及其查询优化

9.1 关系数据库系统的查询处理

9.2 关系数据库系统的查询优化

9.3 代数优化

9.4 物理优化

9.2 关系数据库系统的查询优化

- 9.2.1 查询优化概论
- 9.2.2 一个实例

9.2.1 查询优化概论

- 查询优化的必要性
  - 查询优化极大地影响了关系数据库DBMS的性能。
- 查询优化的可能性
  - 关系数据语言的级别很高，使DBMS可以从关系表达式中分析查询语义。

由DBMS进行查询优化的好处

- 用户不必考虑如何最好地表达查询以获得较好的效率
- 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

由DBMS进行查询优化的好处

(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相适应的执行计划。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，而程序员一般只能考虑有限的几种可能性。

(4)优化器中包括了很多复杂的优化技术

查询优化目标

- 查询优化的总目标

  选择有效策略，求得给定关系表达式的值

- 实际系统的查询优化步骤

  1. 将查询转换成某种内部表示，通常是语法树

  2. 根据一定的等价变换规则把语法树转换成标准（优化）形式

实际系统的查询优化步骤

3. 选择低层的操作算法

对于语法树中的每一个操作

- 计算各种执行算法的执行代价
- 选择代价小的执行算法

4. 生成查询计划(查询执行方案)

- 查询计划是由一系列内部操作组成的。

代价模型

- 集中式数据库
  - 单用户系统

    总代价 = I/O代价 + CPU代价
  - 多用户系统

    总代价 = I/O代价 + CPU代价 + 内存代价
- 分布式数据库

总代价 = I/O代价 + CPU代价[+ 内存代价] + 通信代价

9.2.2 一个实例

[例] 求选修了2号课程的学生姓名

```
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2：一个内存块装元组:10个Student, 或100个SC,

内存中一次可以存放：5块Student元组,1块SC元组和若干块连接结果元组。

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。

执行策略1(考虑I/O时间)

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

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)) ×(10000/100)

=100+20×100=2100块

读数据时间=2100/20=105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

②б

读数据时间 = 50000秒

③П

总时间 =105＋50000＋50000秒 = 100105秒

= 27.8小时

执行策略2(考虑I/O时间)

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

① ⋈

读取总块数= 2100块

读数据时间=2100/20=105秒

中间结果大小=10000 （减少1000倍）

写中间结果时间=10000/10/20=50秒

②б

读数据时间=50秒

③Π

总时间＝105＋50＋50秒＝205秒=3.4分

执行策略3(考虑I/O时间)

3. Q3＝ Π_{Sname}(Student ⋈ б$_{SC.Cno='2'}$ (SC))

①б

读SC表总块数= 10000/100=100块

读数据时间=100/20=5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＝5＋5秒＝10秒

执行策略4(考虑I/O时间)

4. Q4＝ $\Pi_{S\,name}$(Student⋈ $б_{SC.Cno='2'}$ (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= 50/100<1块

读数据时间

中间结果大小=50条 不必写入外存

执行策略4(考虑I/O时间)

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

查询优化的一般准则

- 选择运算应尽可能先做
  - 目的：减小中间关系
- 在执行连接操作前对关系适当进行预处理
  - 按连接属性排序
  - 在连接属性上建立索引
- 投影运算和选择运算同时做
  - 目的：避免重复扫描关系
- 将投影运算与其前面或后面的双目运算结合
  - 目的：减少扫描关系的遍数

查询优化的一般准则

- 例：$\sigma_{Student.Sno=SC.Sno}(Student \times SC)$

$$\Downarrow$$

$$Student \bowtie \sigma_{SC.Cno='2'} SC$$

- 提取公共子表达式

第九章 关系系统及其查询优化

9.1 关系数据库系统的查询处理

9.2 关系数据库系统的查询优化

9.3 代数优化

9.4 物理优化

9.3 代数优化

9.3.1 关系代数等价变换规则

9.3.2 查询树的启发式优化

9.3.1 关系代数等价变换规则

- 关系代数表达式等价
  - 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
  - 上面的优化策略大部分都涉及到代数表达式的变换

常用的等价变换规则

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

$E1 \times E2 \equiv E2 \times E1$

$E1 \bowtie E2 \equiv E2 \bowtie E1$

$E1 \bowtie_F E2 \equiv E2 \bowtie_F E1$

关系代数等价变换规则

2. 连接、笛卡尔积的结合律

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

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

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

F F F F

关系代数等价变换规则

3. 投影的串接定律

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

假设：

1)E是关系代数表达式

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

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

关系代数等价变换规则

4. 选择的串接定律

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

- 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。

关系代数等价变换规则

5. 选择与投影的交换律

(1)假设: 选择条件F只涉及属性A1，…，An

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

(2)假设: F中有不属于A1, …,An的属性B1,…,Bm

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

关系代数等价变换规则

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：

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

关系代数等价变换规则

(3) 假设： $F=F1 \wedge F2$，

F1只涉及E1中的属性，

F2涉及E1和E2两者的属性

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

它使部分选择在笛卡尔积前先做

关系代数等价变换规则

7. 选择与并的交换

假设：E=E1∪E2，E1，E2有相同的属性名

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

8. 选择与差运算的交换

假设：E1与E2有相同的属性名

$б_F(E1-E2) \equiv б_F(E1) - б_F(E2)$

关系代数等价变换规则

9. 投影与笛卡尔积的交换

假设：E1和E2是两个关系表达式，

A1，…，An是E1的属性，

B1，…，Bm是E2的属性

$$\pi_{A1,A2,\cdots,An,B1,B2,\cdots,Bm}(E1\times E2)\equiv \pi_{A1,A2,\cdots,An}(E1)\times \pi_{B1,B2,\cdots,Bm}(E2)$$

关系代数等价变换规则

I0. 投影与并的交换

假设：E1和E2 有相同的属性名

$$\pi_{A1,A2,\cdots,An}(E1 \cup E2) \equiv$$

$$\pi_{A1,A2,\cdots,An}(E1) \cup \pi_{A1,A2,\cdots,An}(E2)$$

小结

1-2: 连接、笛卡尔积的交换律、结合律

3: 合并或分解投影运算

4: 合并或分解选择运算

5-8: 选择运算与其他运算交换

5，9，10: 投影运算与其他运算交换

9.3 代数优化

9.3.1 关系代数等价变换规则

9.3.2 查询树的启发式优化

关系代数表达式的优化算法

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把形如$б_{F1 \wedge F2 \wedge \cdots \wedge Fn}(E)$变换为

$б_{F1}(б_{F2}(\cdots (б_{Fn}(E))\cdots))$

关系代数表达式的优化算法

（2）通过交换选择运算，将其尽可能移到叶端

对每一个选择，利用规则4-8尽可能把它移到树的叶端。

（3）通过交换投影运算，将其尽可能移到叶端

对每一个投影利用规则3，9，l0，5中的一般形式尽可能把它移向树的叶端。

关系代数表达式的优化算法

（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，但这样做效率更高。

关系代数表达式的优化算法

（5）对内结点分组

- 把上述得到的语法树的内节点分组。
- 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。

关系代数表达式的优化算法

（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。

代数优化的一般步骤

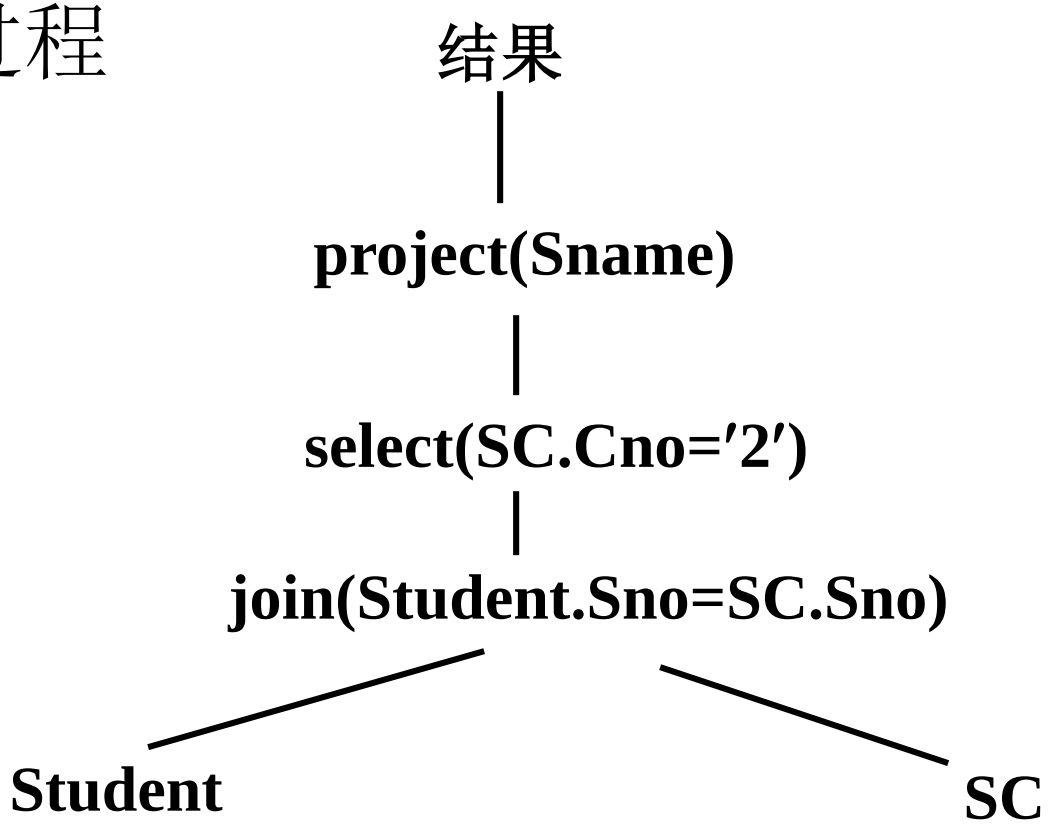
（1）把查询转换成某种内部表示

例：求选修了2号课程的学生姓名

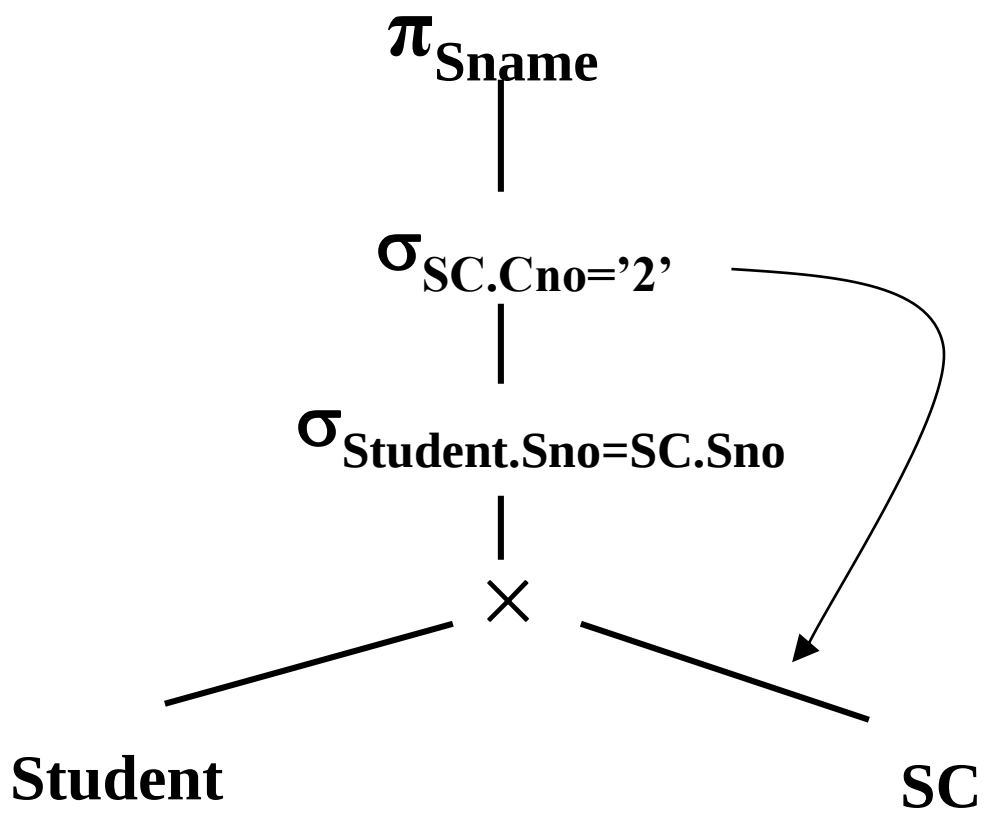
```
SELECT  Student.Sname
FROM    Student, SC
WHERE  Student.Sno=SC.Sno
AND     SC.Cno='2';
```

（1）把查询转换成某种内部表示

查询过程



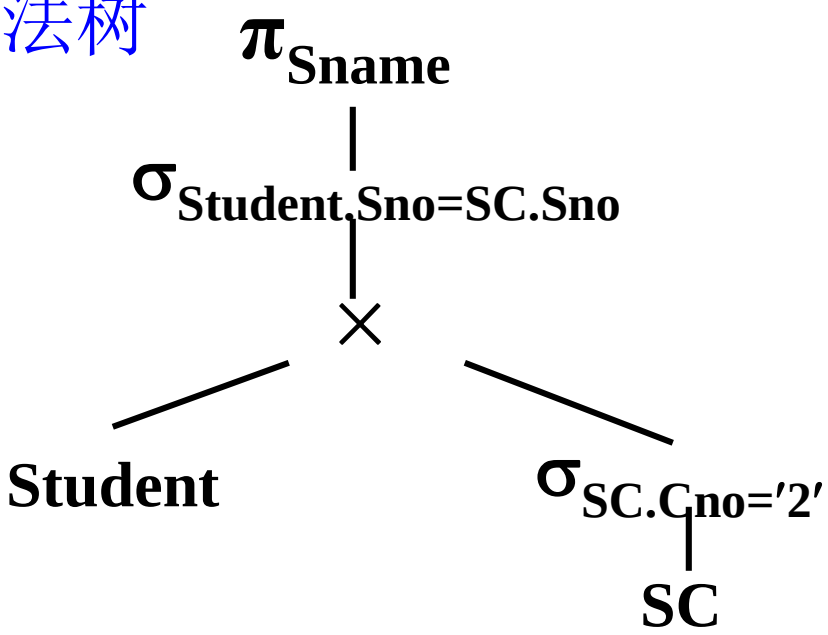
关系代数语法树



（2）代数优化

利用优化算法把语法树转换成标准（优化）形式

优化后的语法树



第九章 关系系统及其查询优化

9.1 关系数据库系统的查询处理

9.2 关系数据库系统的查询优化

9.3 代数优化

9.4 物理优化

9.4 物理优化

- 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

- 9.4.1 基于启发式规则的存取路径选择优化
- 9.4.2 基于代价估算的优化

物理优化

- 物理优化方法
  - 基于规则的启发式优化
    - 启发式规则是指那些在大多数情况下都适用，但不是在每种情况下都是适用的规则。
  - 基于代价估算的优化
    - 优化器估算不同执行策略的代价，并选出具有最小代价的执行计划。

（3）物理优化：选择低层的存取路径

- 优化器查找数据字典获得当前数据库状态信息
  - ➓选择字段上是否有索引
  - ➓连接的两个表是否有序
  - ➓连接字段上是否有索引

ॐ然后根据一定的优化规则选择存取路径

如代数优化例子中若SC表上建有Cno的索引，则应该利用这个索引，而不必顺序扫描SC表。

（4）生成查询计划，选择代价最小的

--在作连接运算时，若两个表(设为R1，R2)均无序，连接属性上也没有索引，则可以有下面几种查询计划：

- ⑩ 对两个表作排序预处理
- ⑩ 对R1在连接属性上建索引
- ⑩ 对R2在连接属性上建索引
- ⑩ 在R1，R2的连接属性上均建索引

--对不同的查询计划计算代价，选择代价最小的一个。

--在计算代价时主要考虑磁盘读写的I/O数，内存CPU处理时间在粗略计算时可不考虑。

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

本章总结

- 主要内容：

  关系系统，查询优化

- 重要知识点：

  关系代数等价变换规则，关系代数表达式的查询优化问题，优化策略

- 本章题型：

  选择，填空，应用题

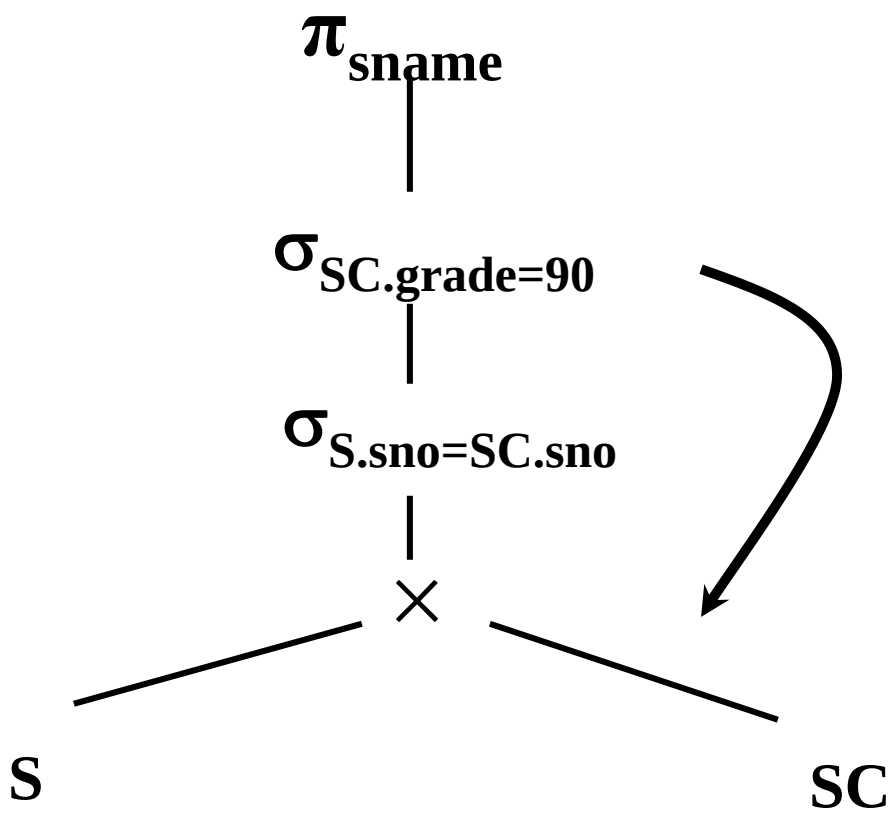
本章习题

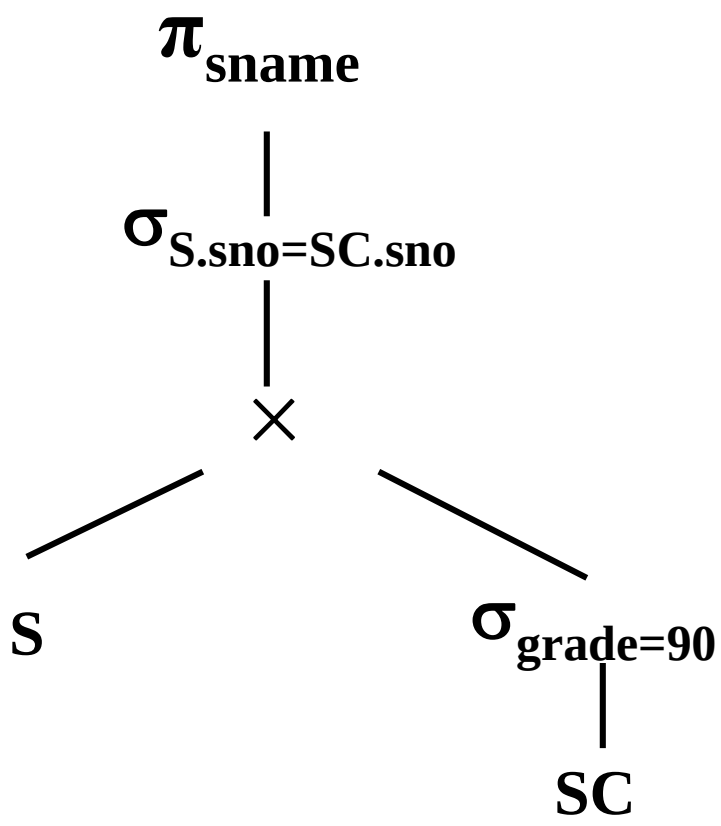
以下各题均对学生-课程数据库进行查询，并且用关系代数表达式表示该查询并画出初始语法树，并将其进行优化。

1.查询选修课成绩为90的学生的姓名。

$\Pi_{sname}(\sigma_{grade=90}(S \bowtie SC))$

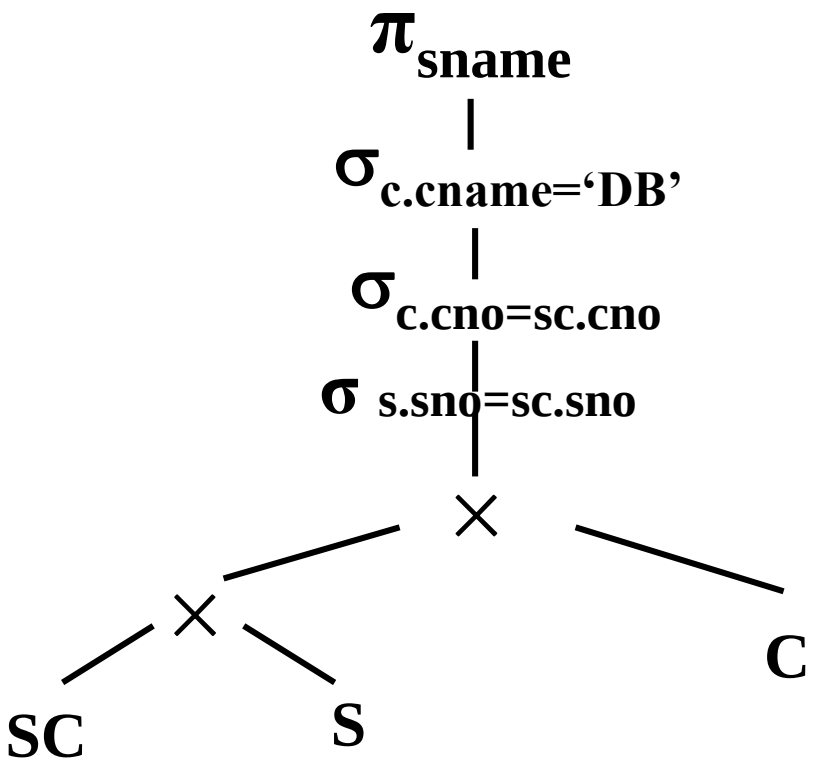
$\Pi_{sname}(\sigma_{grade=90}(\sigma_{s.sno=sc.sno}(S \times SC)))$

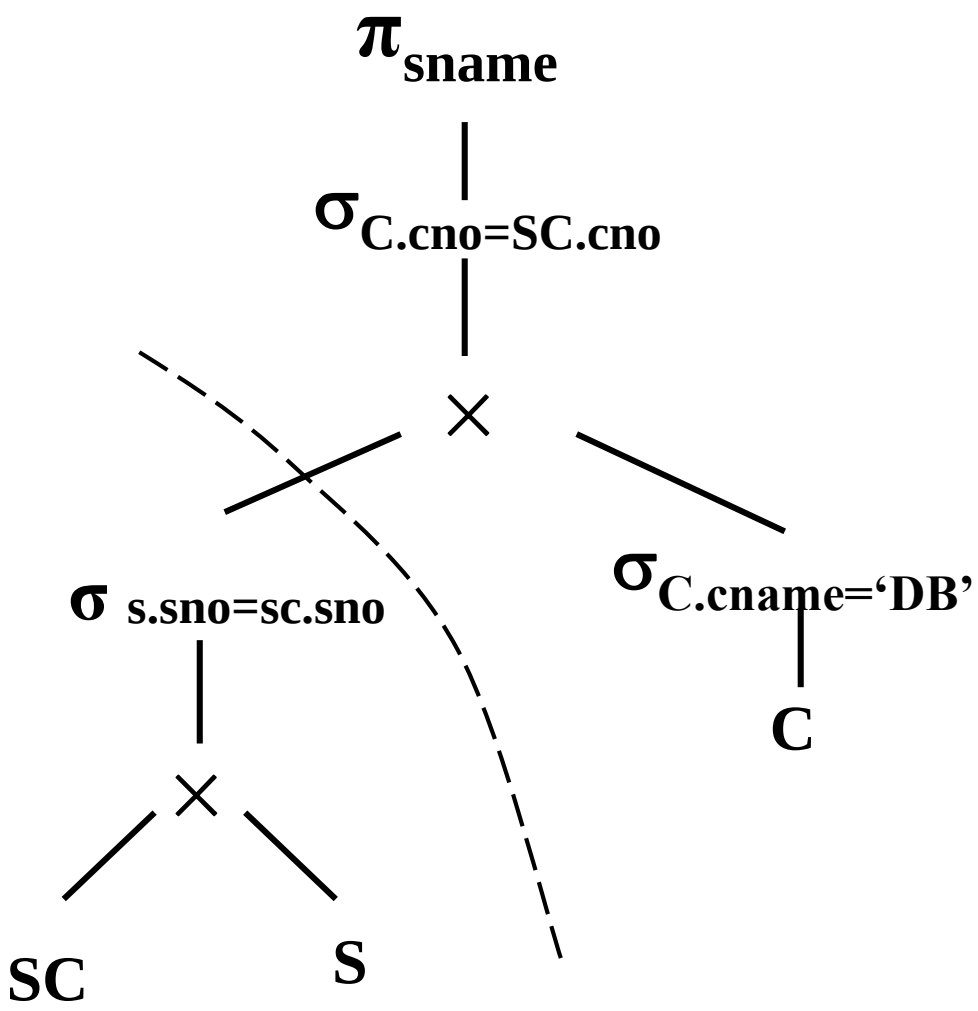

π sname
σ SC.grade=90
σ S.sno=SC.sno
×
S
SC


πsname
σS.sno=SC.sno
×
S
σgrade=90
SC

2.查询选修课程名为DB的学生姓名。

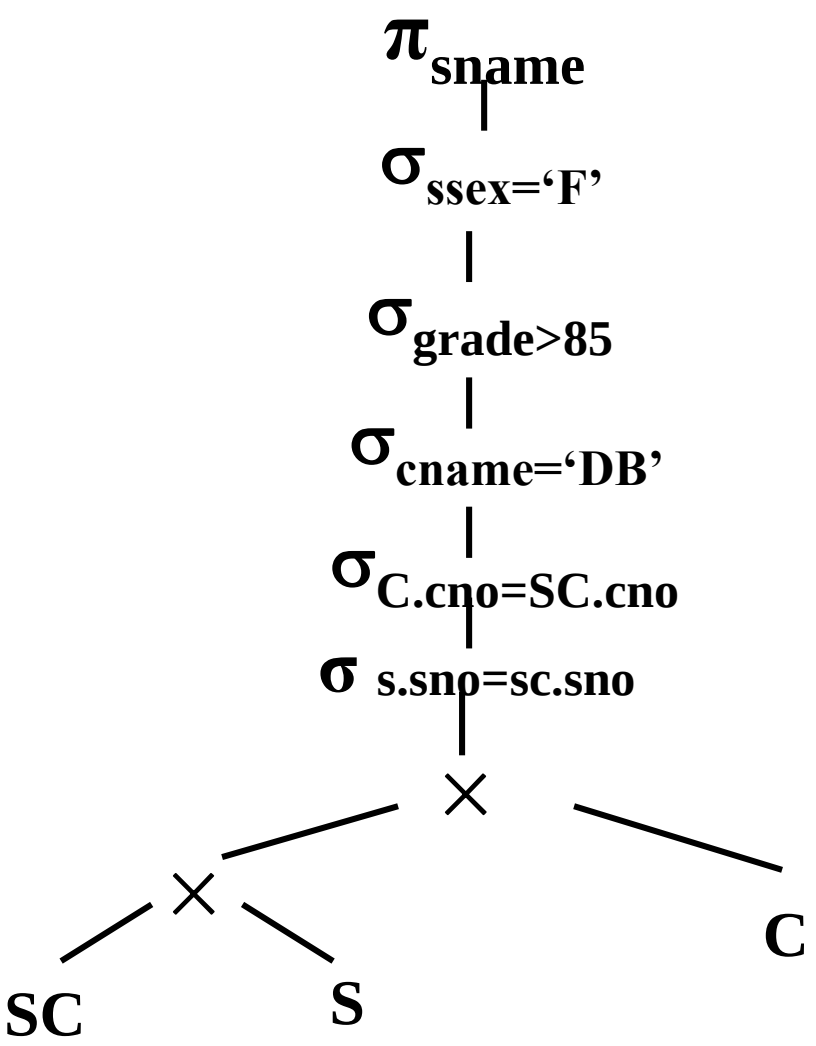
$\Pi_{sname}(\sigma_{c.cname='DB'}(\sigma_{s.sno=sc.sno \wedge sc.cno=c.cno}(S \times SC \times C)))$

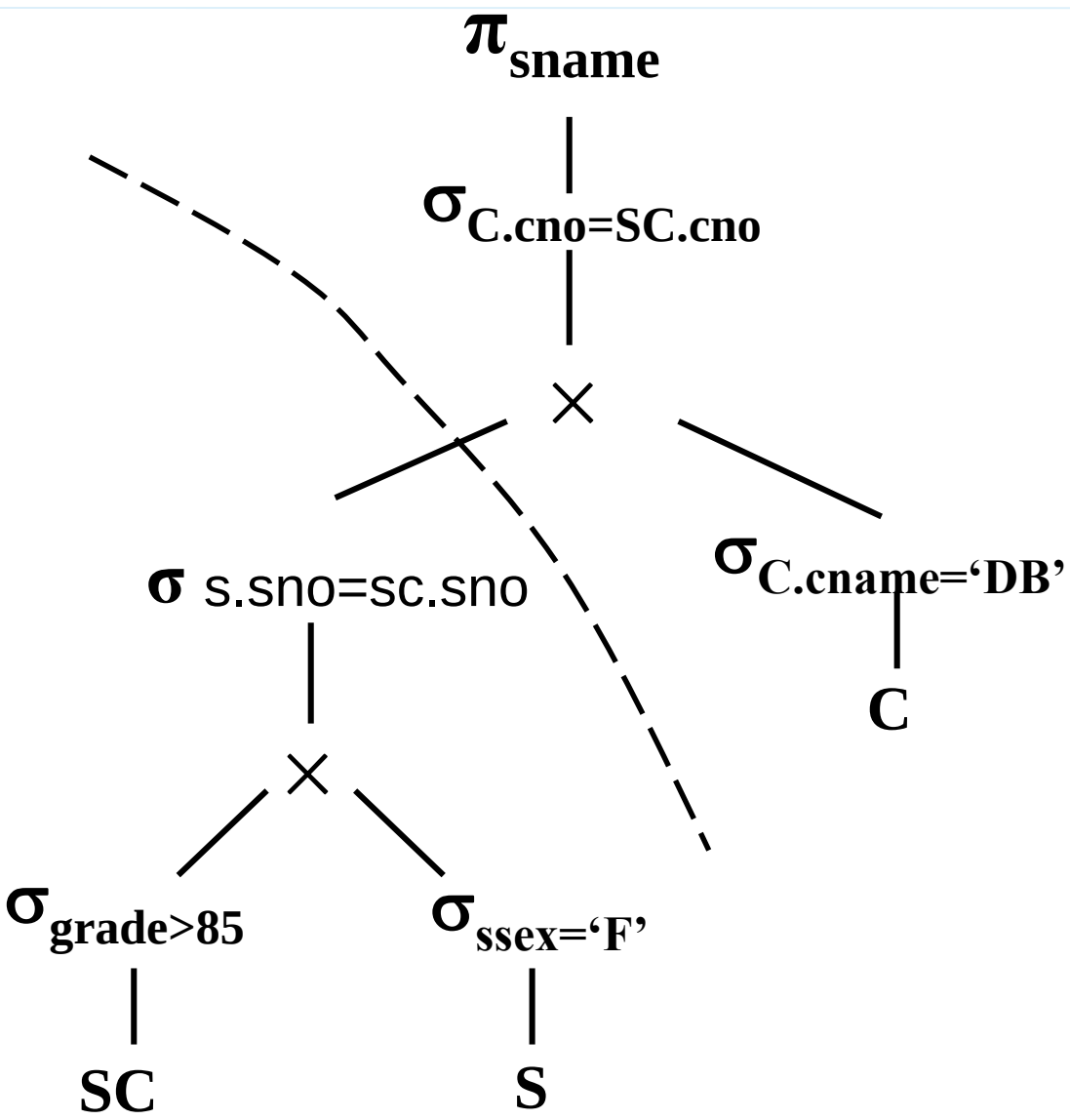



π sname
σ C.cno=SC.cno
×
σ s.sno=sc.sno
σ C.cname='DB'
×
C
SC
S

3.查询选修课程名为DB的成绩在85分以上的女学生姓名。

$\Pi_{sname}(\sigma_{cname='DB' \wedge ssex='F' \wedge grade>85}(\sigma_{s.sno=sc.sno \wedge sc.cno=c.cno}(S \times SC \times C)))$


π sname
σ ssex='F'
σ grade>85
σ cname='DB'
σ C.cno=SC.cno
σ s.sno=sc.sno
×
×
SC
S
C


πsname
σC.cno=SC.cno
×
σ s.sno=sc.sno
σC.cname='DB'
C
×
σgrade>85
σssex='F'
SC
S