# 第9章 关系查询处理和查询优化

- 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查询优化概述

9.2.2一个实例

## 9.2.1 查询优化概述

- •关系系统的查询优化
  - 是关系数据库管理系统实现的关键技术又是关系系统的优点所在
  - •减轻了用户选择存取路径的负担

- 查询优化的优点
  - 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。
    - (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,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,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。

- 关系数据库管理系统通过某种代价模型计算出各种查询执 行策略的执行代价,然后选取代价最小的执行方案
  - 集中式数据库
    - ●执行开销主要包括
      - ➤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
    - ●I/O代价是最主要的
  - 分布式数据库
    - ●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
  - 选择有效的策略
  - 求得给定关系表达式的值
  - 使得查询代价最小(实际上是较小)

# 9.2 关系数据库系统的查询优化

9.2.1查询优化概述

9.2.2一个实例

#### 9.2.2 一个实例

- •一个关系查询可以对应不同的执行方案, 其效率可能相差非常大。
- [例9.3] 求选修了2号课程的学生姓名。

用SQL表达:

SELECT Student.Sname

FROM Student, SC

WHERE Student.Sno=SC.Sno AND SC.Cno='2'

- •假定学生-课程数据库中有1000个学生记录,10000个选课记录
- 选修2号课程的选课记录为50个

### •可以用多种等价的关系代数表达式来完成这一查询

- $Q_1 = \pi_{Sname}(\sigma_{Student.Sno=SC.Sno \land SC.Cno='2'}(Student \times SC))$
- $Q_2 = \pi_{Sname}(\sigma_{SC.Cno='2'}(Student \bowtie SC))$
- $Q_3 = \pi_{Sname}(Student \bowtie \sigma_{SC.Cno='2'}(SC))$

# 1.第一种情况

•  $Q_1 = \pi_{\text{Sname}}(\sigma_{\text{Student.Sno} = \text{SC.Sno} \land \text{SC.Cno} = '2'}(\text{Student} \times \text{SC}))$ 

#### (1) 计算广义笛卡尔积

#### • 算法:

- 在内存中尽可能多地装入某个表(如Student表)的若干块,留出一块存放另一个表(如SC表)的元组。
- 把SC中的每个元组和Student中每个元组连接,连接后的元组 装满一块后就写到中间文件上
- •从SC中读入一块和内存中的Student元组连接,直到SC表处理完。
- 再读入若干块Student元组,读入一块SC元组
- 重复上述处理过程,直到把Student表处理完

•设一个块能装10个Student元组或100个SC元组,在内存中存放5块Student元组和1块SC元组,则读取总块数:

$$\frac{1000}{10} + \frac{1000}{10 \times 5} \times \frac{10000}{100} = 100 + 20 \times 100 = 2100$$

- 读Student表100块, 读SC表20遍, 每遍100块, 则总计要读取2100数据块。
- •连接后的元组数为10<sup>3</sup>×10<sup>4</sup>=10<sup>7</sup>。设每块能装10个元组,则 写出10<sup>6</sup>块。

#### (2) 作选择操作

- 依次读入连接后的元组,按照选择条件选取满足要求的记录
- •假定内存处理时间忽略。读取中间文件花费的时间(同写中间文件一样)需读入10<sup>6</sup>块。
- 若满足条件的元组假设仅50个,均可放在内存。

- (3) 作投影操作
  - 把第(2)步的结果在Sname上作投影输出,得到最终结果
- •第一种情况下执行查询的总读写数据块=2100+106+106

### 2.第二种情况

$$Q_2 = \pi_{\text{Sname}}(\sigma_{\text{Sc.Cno}='2'} \text{ (Student} \bowtie \text{SC)})$$

- (1) 计算自然连接
  - ●执行自然连接,读取Student和SC表的策略不变,总的读取块数仍为2100块
  - ●自然连接的结果比第一种情况大大减少,为104个元组
  - ●写出数据块= 10³ 块

- (2) 读取中间文件块,执行选择运算,读取的数据块= 10<sup>3</sup> 块
- (3) 把第2步结果投影输出。
- •第二种情况下执行查询的总读写数据块=2100+103+103
- 其执行代价大约是第一种情况的488分之一

#### 3.第三种情况

 $Q_3 = \pi_{Sname}(Student \bowtie \sigma_{SC.Cno='2'}(SC))$ 

- (1) 先对SC表作选择运算,只需读一遍SC表,存取100块,因为满足条件的元组仅50个,不必使用中间文件。
- (2) 读取Student表,把读入的Student元组和内存中的SC元组作连接。也只需读一遍Student表共100块。
  - (3) 把连接结果投影输出

- 第三种情况总的读写数据块=100+100
- 其执行代价大约是第一种情况的万分之一,是第二种情况的20分之一

- •假如SC表的Cno字段上有索引
  - 第一步就不必读取所有的SC元组而只需读取Cno='2'的那些元组(50个)
  - 存取的索引块和SC中满足条件的数据块大约总共3~4块
- 若Student表在Sno上也有索引
  - 不必读取所有的Student元组
  - 因为满足条件的SC记录仅50个,涉及最多50个Student记录
  - 读取Student表的块数也可大大减少

·把代数表达式Q1变换为Q2、Q3

$$Q_1 = \pi_{\text{Sname}}(\sigma_{\text{Student.Sno}=\text{SC.Sno} \land \text{Sc.Cno}='2'}(\text{Student} \times \text{SC}))$$



 $Q_2 = \pi_{\text{Sname}}(\sigma_{\text{Sc.Cno}='2'} (\text{Student} \bowtie \text{SC}))$ 

 $Q_3 = \pi_{Sname}(Student \bowtie \sigma_{SC.Cno='2'}(SC))$ 

• 有选择和连接操作时,先做选择操作,这样参加连接的元组就可以大大减少,这是代数优化

### 实例: 小结

- 在Q<sub>3</sub>中
  - SC表的选择操作算法有全表扫描或索引扫描, 经过初步估算, 索引扫描方法较优。
  - •对于Student和SC表的连接,利用Student表上的索引,采用索引连接代价也较小,这就是物理优化。