数据库系统概论

An Introduction to Database System

第九章 关系查询处理和查询优化

第三篇　系统篇

- 讨论数据库管理系统中查询处理和事务管理的基本概念和基础知识
  - 第9章　关系查询处理和查询优化
  - 第10章　数据库恢复技术
  - 第11章　并发控制

关系查询处理和查询优化

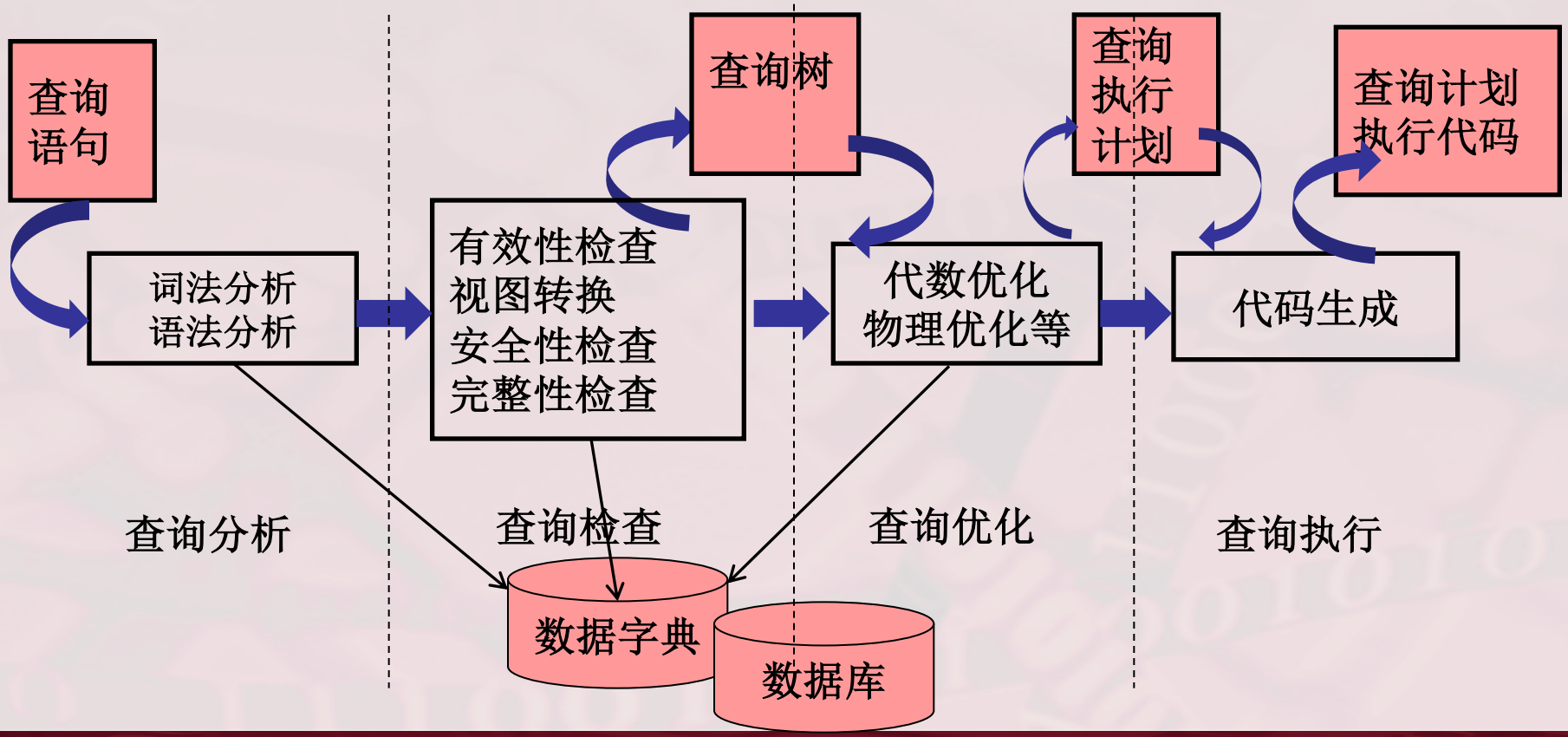
- 本章内容：
  - 关系数据库管理系统的查询处理步骤
  - 查询优化的概念
  - 基本方法和技术
- 查询优化分类：
  - 代数优化：指关系代数表达式的优化
  - 物理优化：指存取路径和底层操作算法的选择

查询处理步骤

- 查询处理的四个阶段：

1. 查询分析
2. 查询检查
3. 查询优化
4. 查询执行

查询处理步骤


查询
语句
词法分析
语法分析
查询树
有效性检查
视图转换
安全性检查
完整性检查
查询
执行
计划
代数优化
物理优化等
查询计划
执行代码
代码生成
查询分析
查询检查
查询优化
查询执行
数据字典
数据库

一个例子

```
SELECT  Sname
FROM  Student
WHERE Sno = “20100017”
```

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
- 这与编译器的原理是一样的

- 词法分析：
  - 保留字：SELECT， FROM， WHERE
  - 变量：Sname, Student, Sno
  - 常量：“20100017”
  - 运算符： =
- 可以发现：拼写错误，命名错误，非法符号等

❖语法分析

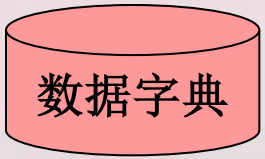
- 按照SQL语言的句法解释查询语句
- 例如：下面的SQL就是语法错误的
- SELECT Sname
  FROM Student
  WHEN Sno=“20100017”

2. 查询检查

- 查询检查的任务
  - 有效性检查
  - 视图转换
  - 安全性检查
  - 完整性初步检查

2. 查询检查

- 有效性检查：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
- 根据数据字典中有关的模式定义信息进行检查



2. 查询检查

- 视图转换
- 如果查询是对视图的操作，则要用视图消解法把对视图的操作转换成对基本表的操作

```
SELECT Sno
FROM Info_Student
WHERE Sname="张三"
```

```
SELECT Sno
FROM Student
WHERE Sname="张三"
      AND  Sdept="Information"
```

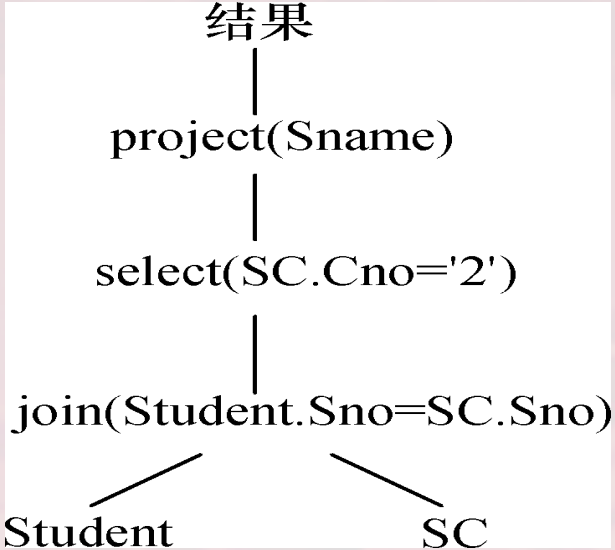
```
Create View Info_student
As  Select * From Student
Where Sdept="Information"
```

2. 查询检查

- 安全性检查
  - 根据数据字典中的用户权限对用户的存取权限进行检查
- 完整性检查
  - 根据数据字典中存储的完整性约束定义，对句子进行检查
  - 例如，**where Sno=20100017** 就是错误的，因为**Sno**是字符类型**Char(8)**

2. 查询检查

- 语法分析树
  - 检查通过后，把SQL查询语句转换成内部表示，一般用语法树来表示



3. 查询优化

- 查询优化是关系数据库成功的关键因素之一。早年的关系数据库由于性能不高备受诟病，正是由于查询优化技术的进步，才使得关系数据库走向产品。
- 查询优化：选择一个高效执行的查询处理策略

3. 查询优化

- 查询优化分类
  - 代数优化/逻辑优化：指针对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(rule based)
  - 基于代价(cost based)
  - 基于语义(semantic based)

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

9.2.1 查询优化概述

- 查询优化的优点
  - 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
  - 减轻了用户对于系统底层选择存取路径的负担
  - 用户就可以关注查询的正确表达上，而无需考虑查询的执行效率如何

查询优化概述

- 系统优化后的程序通常可以比用户程序做得更好

（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，这在实际中往往是不可行的。

查询优化概述（续）

（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。

查询优化概述（续）

- 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)
      - 内存空间的开销
    - 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

查询优化概述（续）

- 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)

一个实例

[例9.3] 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

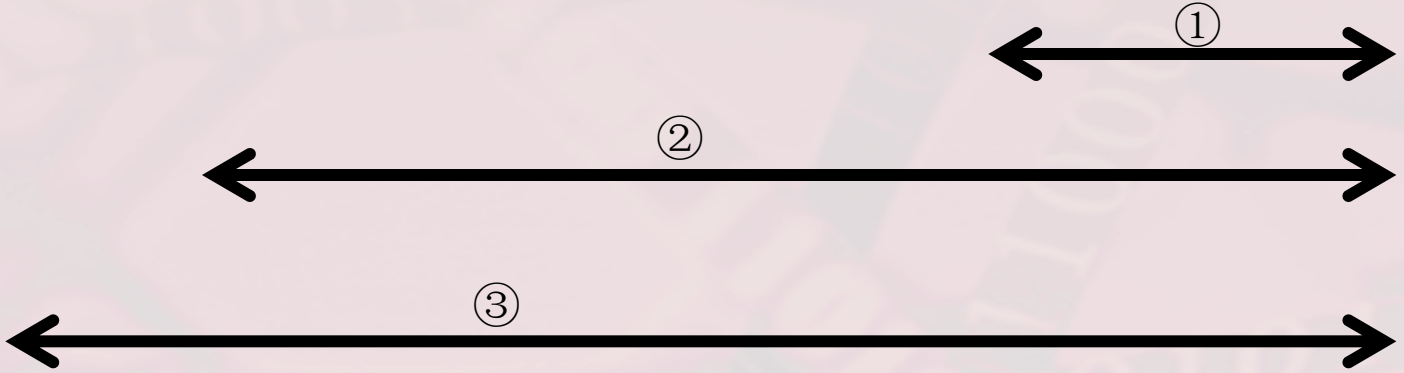
一个实例（续）

- 可以用多种等价的关系代数表达式来完成这一查询
  - $Q_1=\pi_{Sname}(\sigma_{Student.Sno=SC.Sno \wedge SC.Cno='2'}(Student \times SC))$
  - $Q_2=\pi_{Sname}(\sigma_{SC.Cno='2'}(Student \bowtie SC))$
  - $Q_3=\pi_{Sname}(Student \bowtie \sigma_{SC.Cno='2'}(SC))$

一个实例（续）

1.第一种情况

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



一个实例（续）

（1） 计算广义笛卡尔积

- 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

一个实例（续）

- 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，则读取总块数为

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

  - 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$10^3\times10^4=10^7$。设每块能装10个元组，则写出10^6块。

一个实例（续）

（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10^6块。
- 若满足条件的元组假设仅50个，均可放在内存。

一个实例（续）

（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
$=2100+10^6+10^6$

一个实例（续）

2.第二种情况

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

（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10^4个元组
- 写出数据块= 10^3 块

一个实例（续）

2.第二种情况（续）

（2）读取中间文件块，执行选择运算，读取的数据块=

10^3 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+ 10^3 +10^3
- 其执行代价大约是第一种情况的488分之一

一个实例（续）

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

一个实例（续）

3.第三种情况（续）

- 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

一个实例（续）

- 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

一个实例（续）

- 把代数表达式Q1变换为Q2、 Q3

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

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

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

- 先做选择再做连接操作，这样参与连接的元组就可以大大减少，这是代数优化

一个实例（续）

- 在Q_3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。

9.3 代 数 优 化

观察1：一个代数表达式（查询），可以有多个结果相同但是形式不同的表达式

观察2：这些表达式的执行效率是不一样的，有时候还差别巨大！

代数优化策略是通过对关系代数表达式的等价变换来提高查询效率，所谓关系代数表达式的等价是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。

查询树的启发式优化

❖典型的启发式规则

（1）选择运算应尽可能先做

在优化策略中这是最重要、最基本的一条。

（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。

查询树的启发式优化（续）

（3） 把投影同其前或其后的双目运算结合起来，没有必要为了去掉某些字段而扫描一遍关系。

（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，连接特别是等值连接运算要比同样关系上的笛卡尔积省很多时间。

查询树的启发式优化（续）

（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是合算的。
- 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

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，还存在多个存取方案，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，求得更好的查询计划

物理优化（续）

- 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。

物理优化（续）

- 物理优化方法（续）
  - 两者结合的优化方法：
    - 常常先使用启发式规则，选取若干较优的候选方案，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

9.6 小 结

- 查询处理是关系数据库管理系统的核心，查询优化技术是查询处理的关键技术
- 本章主要内容
  - 查询处理过程
  - 查询优化
    - 代数优化
    - 物理优化

查询分析
查询检查
查询优化
查询执行

9.6 小 结

- 查询处理是关系数据库管理系统的核心，查询优化技术是查询处理的关键技术
- 本章主要内容
  - 查询处理过程
  - 查询优化
    - 代数优化
    - 物理优化

9.6 小 结

- 查询处理是关系数据库管理系统的核心，查询优化技术是查询处理的关键技术
- 本章主要内容
  - 查询处理过程
  - 查询优化
    - 代数优化
    - 物理优化

基于规则的存取路径优化
基于代价的优化

小结（续）

- 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关系数据库管理系统上
  - 应该找出关系数据库管理系统的优化规律，以写出适合关系数据库管理系统自动优化的SQL语句
- 对于关系数据库管理系统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