

数据库系统之四

--数据库管理系统实现技术



第21讲 数据库查询优化技术

战德臣

哈尔滨工业大学 教授·博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent
Computing for Enterprises & Services,
Harbin Institute of Technology

本讲学习什么？



基本内容

1. 为什么要及什么是查询优化？
2. 查询优化的基本思路
3. 逻辑查询优化
4. 物理查询优化

重点与难点

- 理解查询优化的整体思路是什么？
- 理解并掌握基于关系代数进行逻辑查询优化的方法和原则
- 理解物理查询优化中的代价估算方法

什么是查询优化?

战德臣

哈尔滨工业大学 教授·博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

什么是查询优化?

(1)为什么需要查询优化



关系数据库的执行效率问题

➤一个例子：

$\pi_{Sname} (\sigma_{SC.C\#=Course.C\# \wedge Student.S\#=SC.S\# \wedge Cname='DB'} (Student \times SC \times Course))$

- Student: **10000**个学生记录(每年2500, 四年)
- Course: **1000**门课程记录
- SC: **10000*50**条选课记录(注: 10000学生, 每人选50门课程)
- Student \times SC \times Course

$10000 * 50 * 10000 * 1000$ 条记录 = $5 * 10^{12}$ 条记录

什么是查询优化?

(1)为什么需要查询优化



关系代数操作执行次序对效率的影响

➤如下三条关系代数语句表示同样的检索需求，但哪一个更好呢？

$\pi_{S\#,Sname,Score}(\sigma_{Cname='DB'}((Student \bowtie SC) \bowtie Course))$

10000

500000

500000

1000

500000

$\pi_{S\#,Sname,Score}(Student \bowtie (SC \bowtie (\sigma_{Cname='DB'}(Course))))$

1000

500000

1

2500(约)

10000

2500(约)

关系代数操作次序是否很重要呢?

$\pi_{S\#,Sname,Score}(\pi_{S\#,Sname}(Student) \bowtie (SC \bowtie (\pi_{C\#}(\sigma_{Cname='DB'}(Course)))))$

查询优化

“如何使数据库查询的执行时间最短？”

三个层面进行优化：

- 语义优化**：利用模型的语义及完整性规则，优化查询。
- 语法优化**---**逻辑层优化**：利用语法结构，优化操作执行顺序；
- 执行优化**---**物理层优化**：存取路径和执行算法的选择与执行次序优化；

详细内容，
下文分解

查询优化的总体思路?

战德臣

哈尔滨工业大学 教授·博士生导师

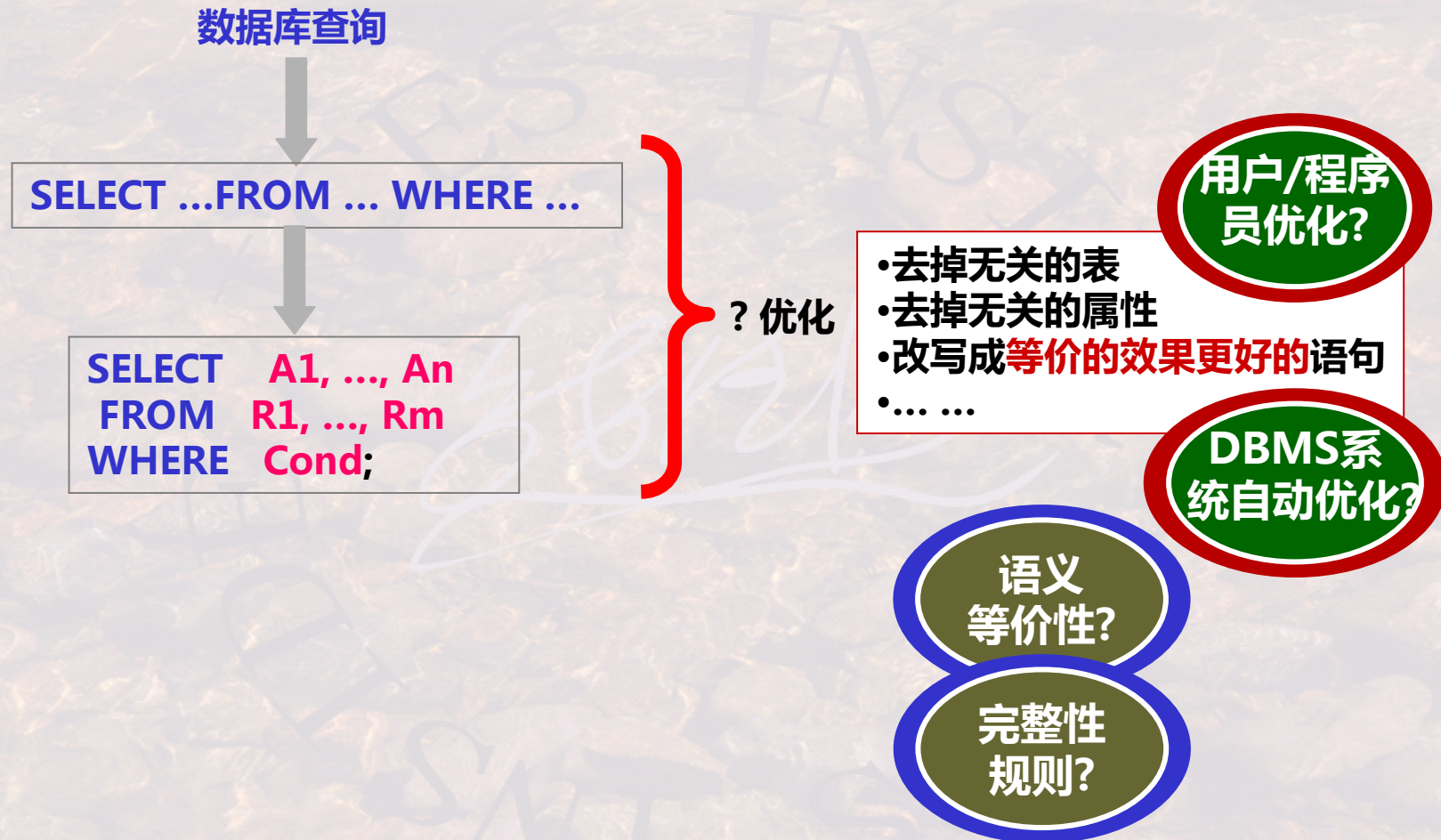
黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

查询优化的总体思路

(1)语义优化---内容等价性



此内容不在本
课程中讨论

查询优化的总体思路

(2)语法优化(逻辑层优化)---语法等价性



查询优化的总体思路

(2)语法优化(逻辑层优化)---语法等价性



逻辑优化/语法优化

基本思想：改变关系代数的操作次序：尽可能早做选择和投影运算

关系代数的五种基本操作中哪两个能够交换次序呢？

次序改变前后两个表达式的等价性问题

关系代数表达式的等价变换定理及其证明(略)

关系代数表达式的优化算法？-逻辑查询计划形成

语法
等价性？

查询优化的总体思路

(3)执行优化(物理层优化)

SELECT A1, ..., An FROM R1, ..., Rm
WHERE Cond;

编译

$\pi_{A1, \dots, An} (\sigma_{Cond} (R1 \times \dots \times Rm))$

语法优化 逻辑层优化

$\pi_{A1, \dots, An} (\sigma_{Cond} (\pi_{a1} (\sigma_{Cond1} (R1)) \times \dots \times \pi_{am} (\sigma_{Condm} (Rm)))$

逻辑查
询计划

执行优化 物理层优化

为每一个关系代数操作**选取**优化的执行层**例行程序**, 形成物理查询计划

物理查询计划
--基于不同算法的
实现程序构造

执行引擎: 依查询计划调用相应的**例行程序**进行处理, 并返回结果

物理优化/执行优化

获取数据库的相关信息
(定期统计)

选取相应的执行层例行程序

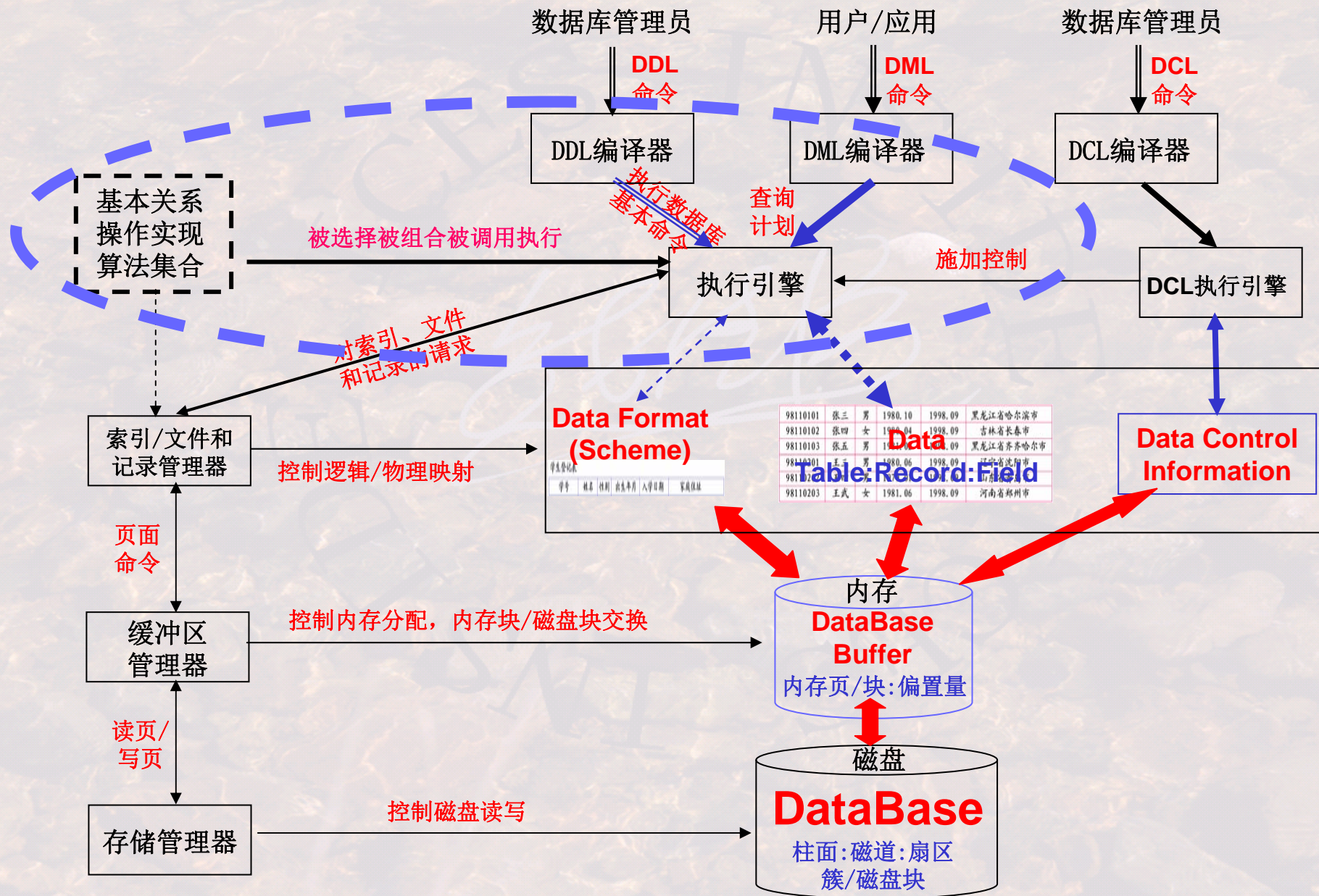
依据相关信息进行代价估算, 并选择代价最少的例行程序及确定相应的参数

形成查询计划: 以基本的例行程序为基本, 确定这些例行程序的执行顺序

实现同一关系操作的不同例行程序库

查询优化的总体思路

(4)在DBMS中的位置



查询优化的总体思路

(5)简要回顾

SELECT A1, ..., An FROM R1, ..., Rm
WHERE Cond;

编译

$\pi_{A1, \dots, An} (\sigma_{Cond}(R1 \times \dots \times Rm))$

逻辑查询计划—
逻辑层优化

物理查询计划—
物理层优化

关系代数
操作次序
优化

代价估算

算法选择与
装配次序

DBMS核
心技术

由执行引擎解释并调用算法(程序)予以执行

逻辑层查询优化策略?

(暨关系代数操作优化)

战德臣

哈尔滨工业大学 教授·博士生导师
黑龙江省教学名师
教育部大学计算机课程教学指导委员会委员

Research Center on Intelligent
Computing for Enterprises & Services,
Harbin Institute of Technology

逻辑层查询优化(暨关系代数操作优化)

(1)逻辑查询优化—总体思路



逻辑优化/语法优化

基本思想：改变关系代数的操作次序：尽可能早做选择和投影运算

关系代数的五种基本操作中哪两个能够交换次序呢？

次序改变前后两个表达式的等价性问题

关系代数表达式的等价变换定理及其证明(略)

关系代数表达式的优化算法？—逻辑查询计划形成

语法等价性，即形式上等价性

[示例]考虑一图书馆的关系数据库。

BOOKS (TITLE , AUTHOR , PNAME , LC_NO)

注：PNAME为出版社名，LC_NO为图书馆图书编目号

PUBLISHERS (PNAME , PADDR , PCITY)

注：PADDR为出版社地址，PCITY为出版社所在地

BORROWERS (NAME , ADDR , CITY , CARD_NO)

注：NAME为读者名，ADDR为读者所在地址，CITY为读者所在城市，CARD_NO为图书证号。

LOANS (CARD_NO , LC_NO , DATE)

注：DATE为借出日期

➤为方便用户使用，定义了视图**XLOANS**:

$$\text{XLOANS} = \Pi_S(\sigma_F(\text{LOANS} \times \text{BORROWERS} \times \text{BOOKS}))$$
$$S = \text{TITLE , AUTHOR , PNAME , LC_NO , NAME , ADDR , CITY , CARD_NO , DATE}$$
$$F = (\text{BORROWERS.CARD_NO} = \text{LOANS.CARD_NO}) \wedge (\text{BOOKS.LC_NO} = \text{LOANS.LC_NO})$$

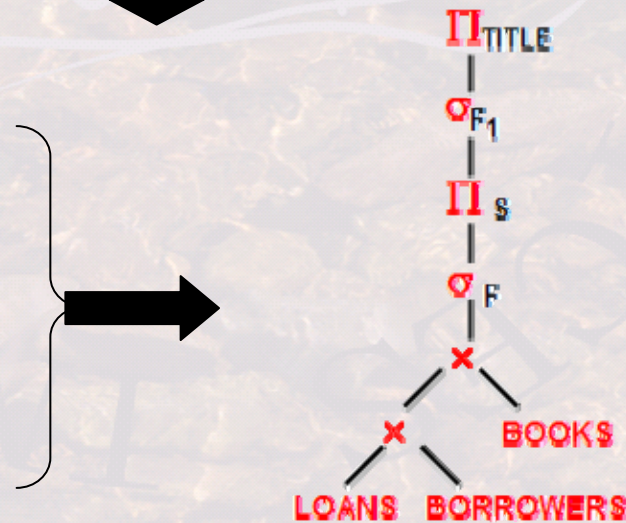
逻辑层查询优化(暨关系代数操作优化)

(3)用语法树表达关系代数表达式

查询：“查出1978年1月1日前被借出的所有书的书名”

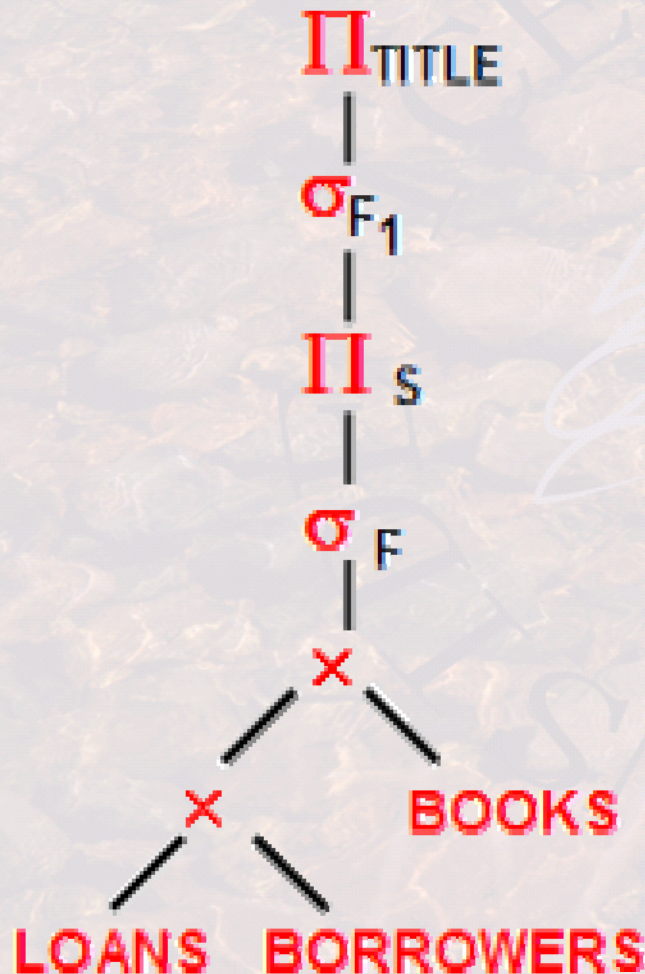
“SELECT Title FROM XLOANS WHERE Date <= 1/1/78”

$\Pi_{\text{TITLE}}(\sigma_{\text{DATE} \leq \text{"1/1/78"}}(\text{XLOANS}))$



由树叶向树根反映了操作先后次序

其中： $F = F_2 \wedge F_3$ ； $F_1 = \text{"DATE} \leq 1/1/78\text{"}$ ；
 $F_2 = \text{"BORROWERS.CARD_NO} = \text{LOANS.CARD_NO"}$ ；
 $F_3 = \text{"BOOKS.LC_NO} = \text{LOANS.LC_NO"}$ 。



逻辑优化的策略

(1) **尽可能地早做选择和投影**：可使中间结果变小，节省几个数量级的执行时间。

(2) **把选择与投影串接起来**：一元运算序列可一起执行，只需对整个关系扫描一遍。

(3) **把投影与其前或后的二元运算结合起来**：在第一次用关系时去掉一些无关属性，可以避免多次扫描整个关系。

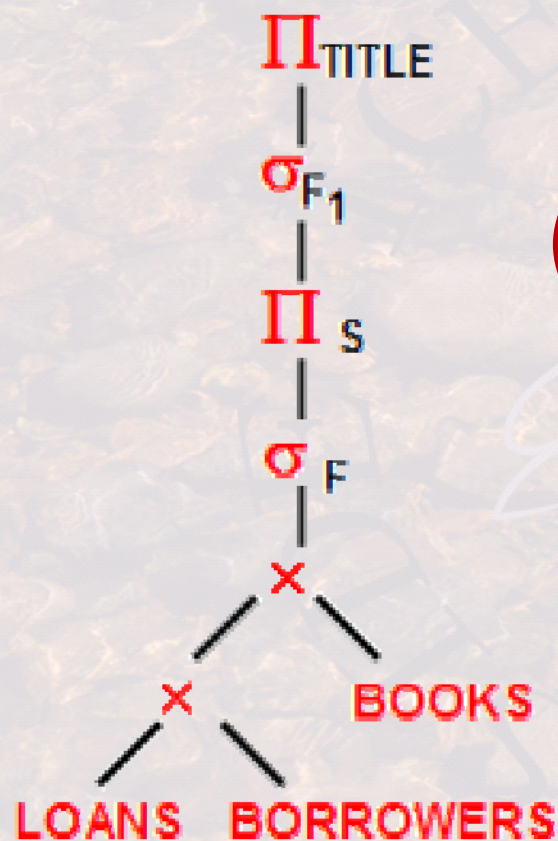
(4) **把某些选择与其前的笛卡尔积合并成一个连接**：当 $R \times S$ 前有选择运算且其中有条件是 R 、 S 属性间比较的运算时，可将其转化为连接运算可节省时间。

(5) **执行连接运算前对关系做适当预处理**：文件排序、建立临时索引等，可使两关系公共值高效联接。

(6) **找出表达式里的公共子表达式**：若公共子表达式结果不大，则预先计算，以后可读入此结果，节时多，尤当视图情况下有用。

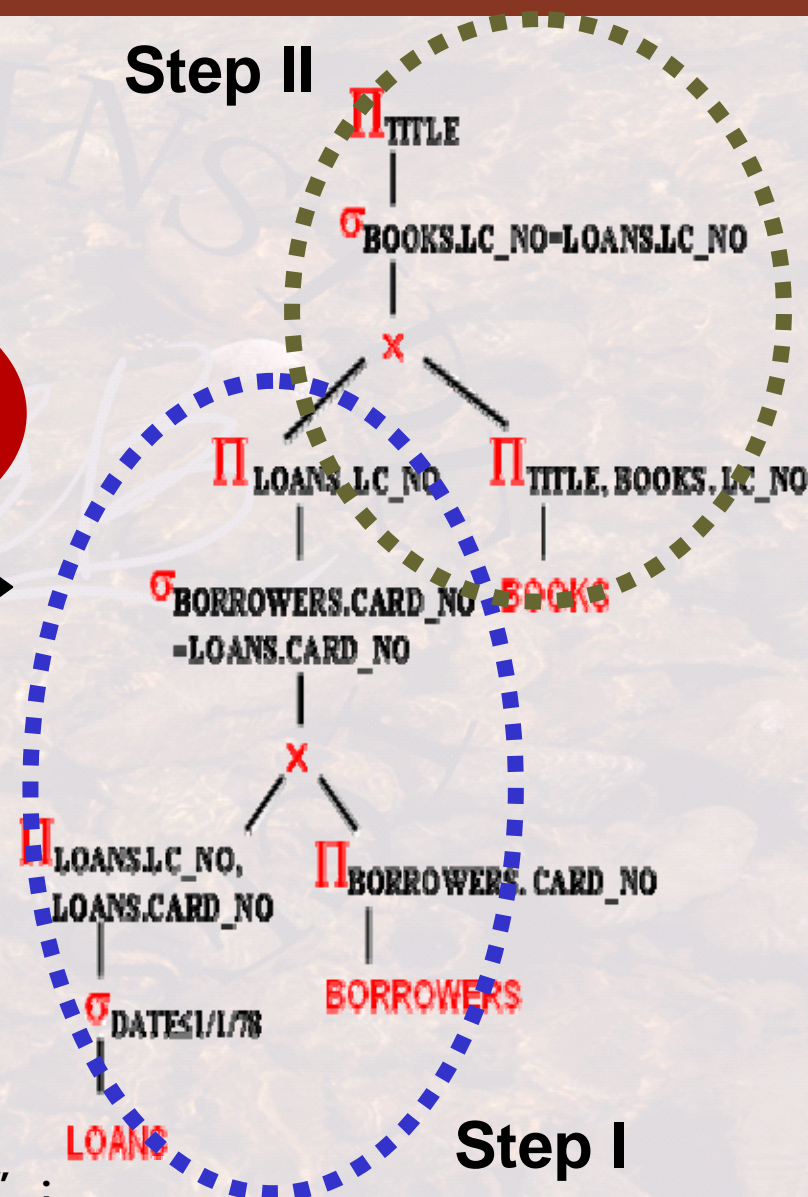
逻辑层查询优化(暨关系代数操作优化)

(5)示例的优化结果示意



怎么实现
呢？另文
分解

Step II



其中： $F = F_2 \wedge F_3$ ； $F_1 = \text{"DATE} \leq 1/1/78\text{"}$ ；
 $F_2 = \text{"BORROWERS.CARD_NO} = \text{LOANS.CARD_NO"}$ ；
 $F_3 = \text{"BOOKS.LC_NO} = \text{LOANS.LC_NO"}$ 。

关系代数操作次序交换的等价性

战德臣

哈尔滨工业大学 教授·博士生导师

黑龙江省教学名师

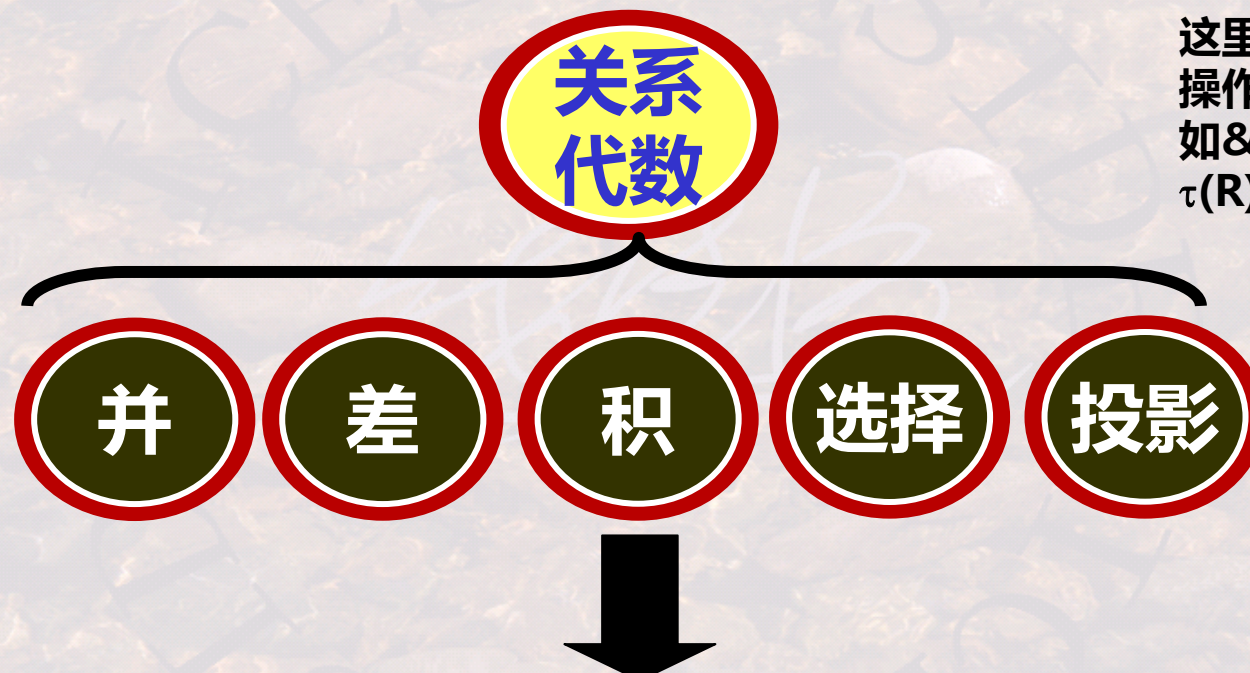
教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

关系代数操作次序交换的等价性

(1)拟解决的问题？

“哪些关系代数操作能够交换次序”



这里仅考虑基本操作，其他操作如 $\bowtie(R)$, $\gamma(R)$, $\tau(R)$ 等暂不考虑

“两两组合交换，验证组合前后是否等价？”

- ✓等价--可交换
- ✓不等价--不可交换

关系代数
交换定理

[定义] 设 E_1, E_2 是两个关系操作表达式。若 E_1, E_2 表示相同的映射，即当 E_1, E_2 的同名变量代入相同关系后产生相同的结果(映像集合)，则说 E_1, E_2 是等价的，记为 $E_1 \equiv E_2$ 。

注：关系可被看成 (1) k 元组集合(属性有先后次序)。(2)一组属性（名）到值的映像集合(元组中的属性没有先后次序)。只有把关系看成是从一组属性名到值的映像的集合，等价性才成立。

家庭

丈夫	妻子	子女
李基	王方	李键
张鹏	刘玉	张睿
张鹏	刘玉	张峰

家庭

丈夫	子女	妻子
李基	李键	王方
张鹏	张峰	刘玉
张鹏	张睿	刘玉

关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换？

定理L1：连接与连接，积与积的交换律

设 E_1, E_2 是关系代数表达式， F 是 E_1, E_2 中属性的附加限制条件，则有：

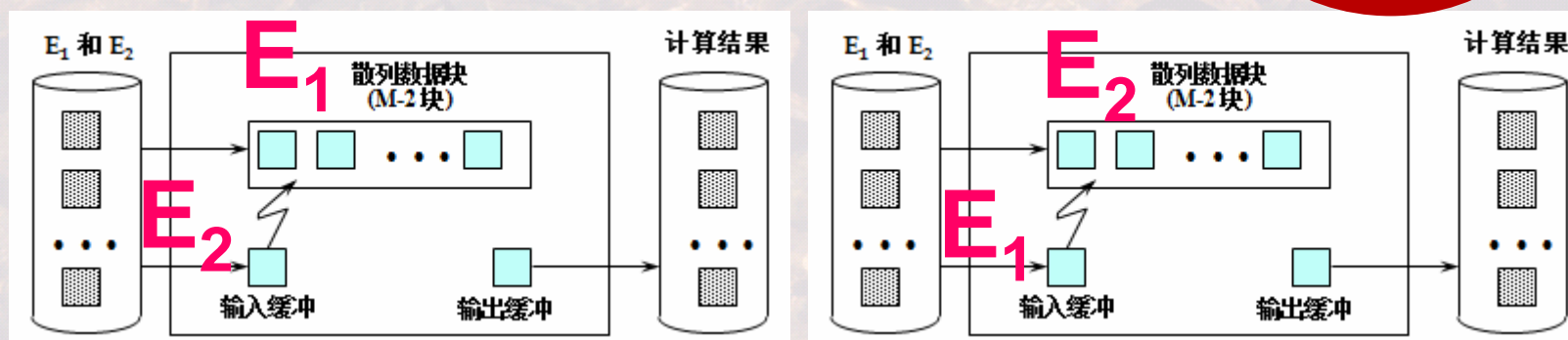
$$(1) E_1 \bowtie_F E_2 \equiv E_2 \bowtie_F E_1$$

$$(2) E_1 \bowtie E_2 \equiv E_2 \bowtie E_1$$

$$(3) E_1 \times E_2 \equiv E_2 \times E_1$$

注：并运算、交运算也有这种交换律

这个定理似乎
没有什么用
哟？真的吗



通常，我们选择结果集合小的表达式，先装入内存。

关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换？

定理L2：连接与连接、积和积的结合律

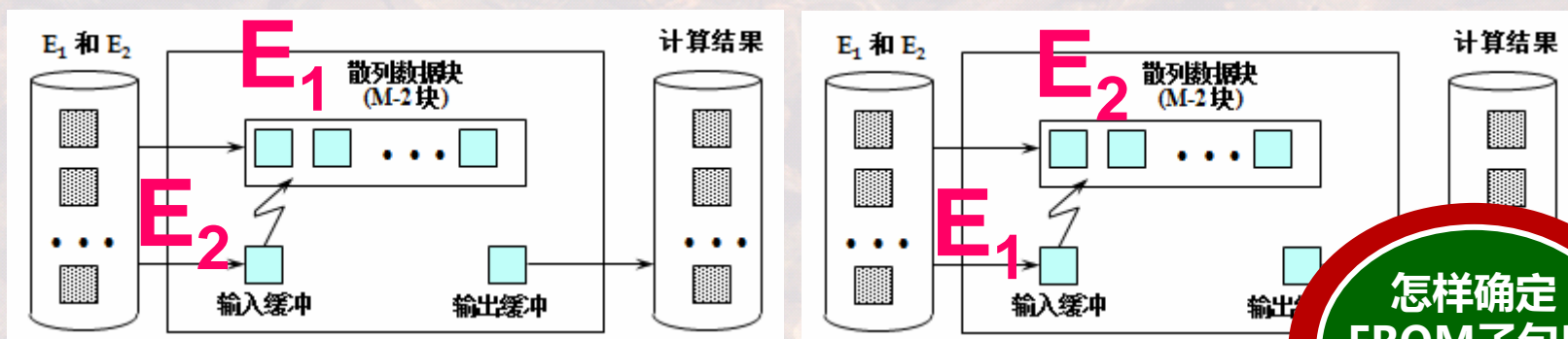
若 E_1, E_2, E_3 是关系代数表达式， F_1, F_2 是条件，则有：

$$(1) (E_1 \bowtie_{F_1} E_2) \bowtie_{F_2} E_3 \equiv E_1 \bowtie_{F_1} (E_2 \bowtie_{F_2} E_3)$$

$$(2) (E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3)$$

$$(3) (E_1 \times E_2) \times E_3 \equiv E_1 \times (E_2 \times E_3)$$

注：并运算、交运算也有这种结合律



SELECT * FROM R1, R2, R3, R4 WHERE ...

通常，我们选择结果集合小的表达式，先计算。

怎样确定
FROM子句后
的乘积次序呢？

关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换?

定理L3：投影串接律

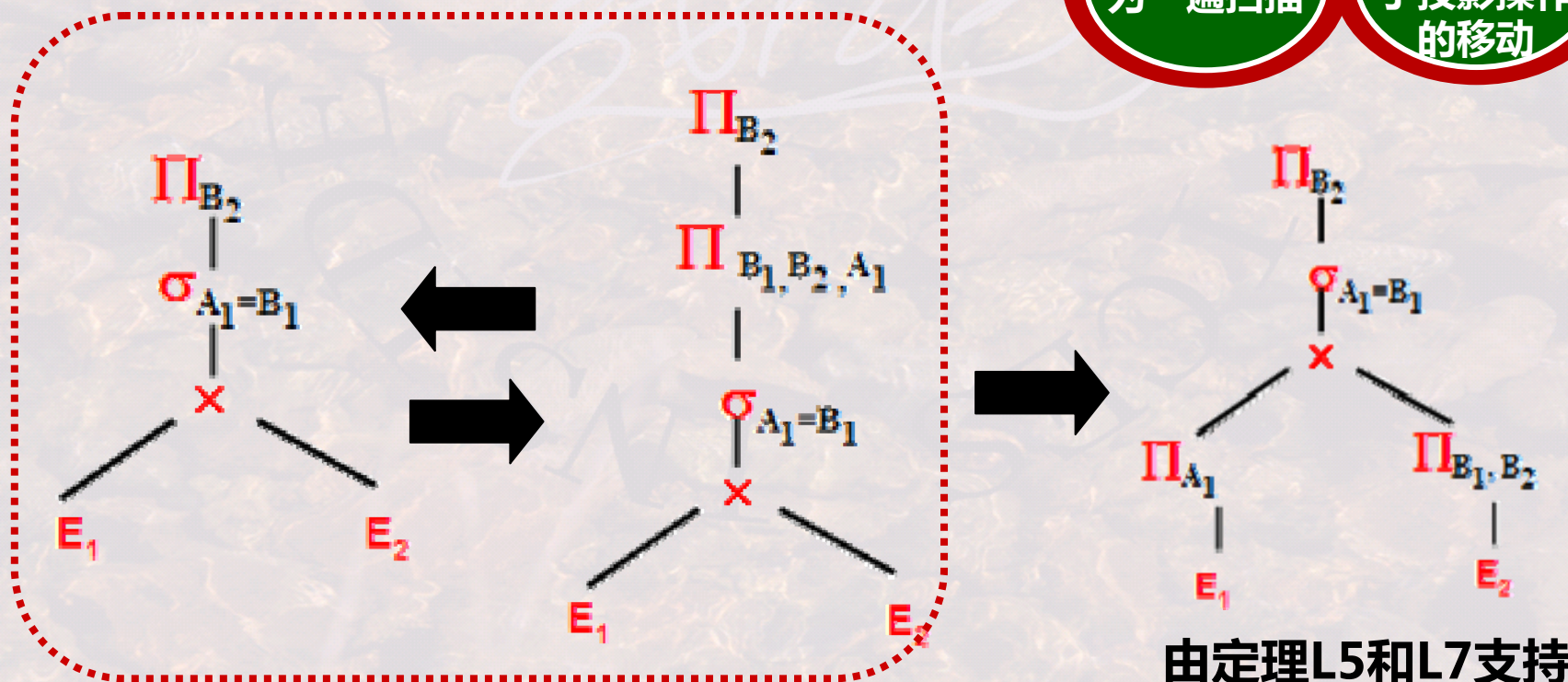
设属性集 $\{A_1, \dots, A_n\} \subseteq \{B_1, \dots, B_m\}$, E 是表达式, 则有:

$$\pi_{A_1, \dots, A_n} (\pi_{B_1, \dots, B_m} (E)) \equiv \pi_{A_1, \dots, A_n} (E)$$

此定理可双向使用。

两遍扫描变为一遍扫描

属性扩展便于投影操作的移动



关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换？

定理L4：选择串接律

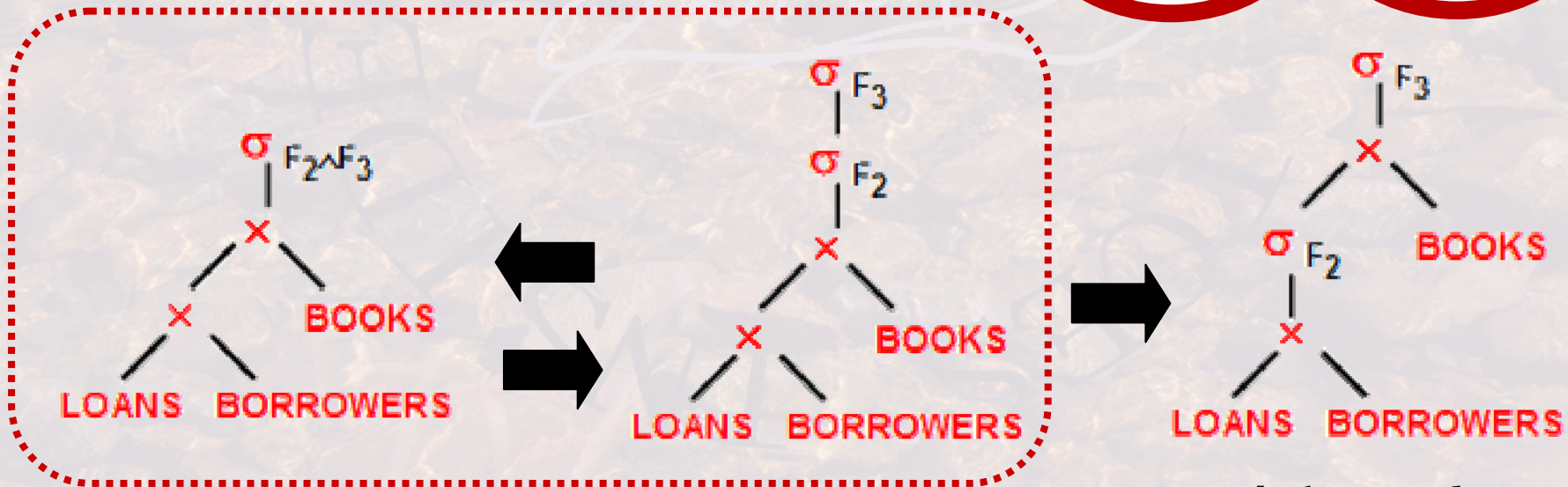
若E是关系代数表达式， F_1, F_2 是条件，则有：

$$\sigma_{F_1}(\sigma_{F_2}(E)) \equiv \sigma_{F_1 \wedge F_2}(E)$$

此定理可双向使用。

两遍扫描变为一遍扫描

分解复杂操作便于选择操作的移动



$F_2 = \text{"BORROWERS.CARD_NO = LOANS.CARD_NO"} ;$
 $F_3 = \text{"BOOKS.LC_NO = LOANS.LC_NO"} .$

由定理L6实现

关系代数操作次序交换的等价性

(3) 哪些关系操作次序可以交换？

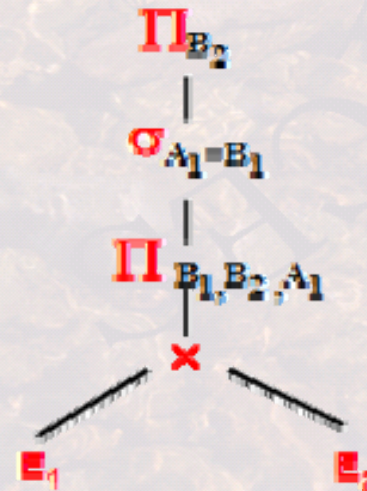
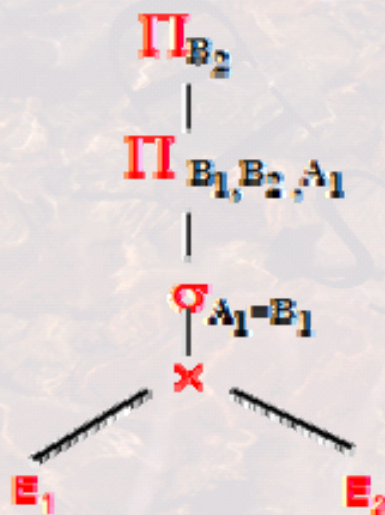
定理L5：选择和投影交换律

设条件F只涉及属性 $\{A_1, \dots, A_n\}$ ，E是关系表达式，则有：

$$\pi_{A_1, \dots, A_n}(\sigma_F(E)) \equiv \sigma_F(\pi_{A_1, \dots, A_n}(E))$$

更一般地，若F还涉及不属于 $\{A_1, \dots, A_n\}$ 的属性 $\{B_1, \dots, B_m\}$ ，则：

$$\pi_{A_1, \dots, A_n}(\sigma_F(E)) \equiv \pi_{A_1, \dots, A_n}(\sigma_F(\pi_{A_1, \dots, A_n, B_1, \dots, B_m}(E)))$$



“尽可能地”

关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换？

定理L6：选择和积的交换律 设 E_1, E_2 是关系代数表达式

(1)若条件 F 只涉及 E_1 中的属性，则有：

$$\sigma_F(E_1 \times E_2) \equiv \sigma_F(E_1) \times E_2$$

(2)若 $F = F_1 \wedge F_2$ ， F_1, F_2 分别只涉及 E_1, E_2 中属性，则有：

$$\sigma_F(E_1 \times E_2) \equiv \sigma_{F_1}(E_1) \times \sigma_{F_2}(E_2)$$

(3)若 $F = F_1 \wedge F_2$ ， F_1 只涉及 E_1 中属性，而 F_2 涉及 E_1, E_2 中属性，则有：

$$\sigma_F(E_1 \times E_2) \equiv \sigma_{F_2}(\sigma_{F_1}(E_1) \times E_2)$$



“这是根本”

“尽可能地”
早做选择

$F_2 = \text{"BORROWERS.CARD_NO = LOANS.CARD_NO"}$;
 $F_3 = \text{"BOOKS.LC_NO = LOANS.LC_NO"}$ 。

关系代数操作次序交换的等价性

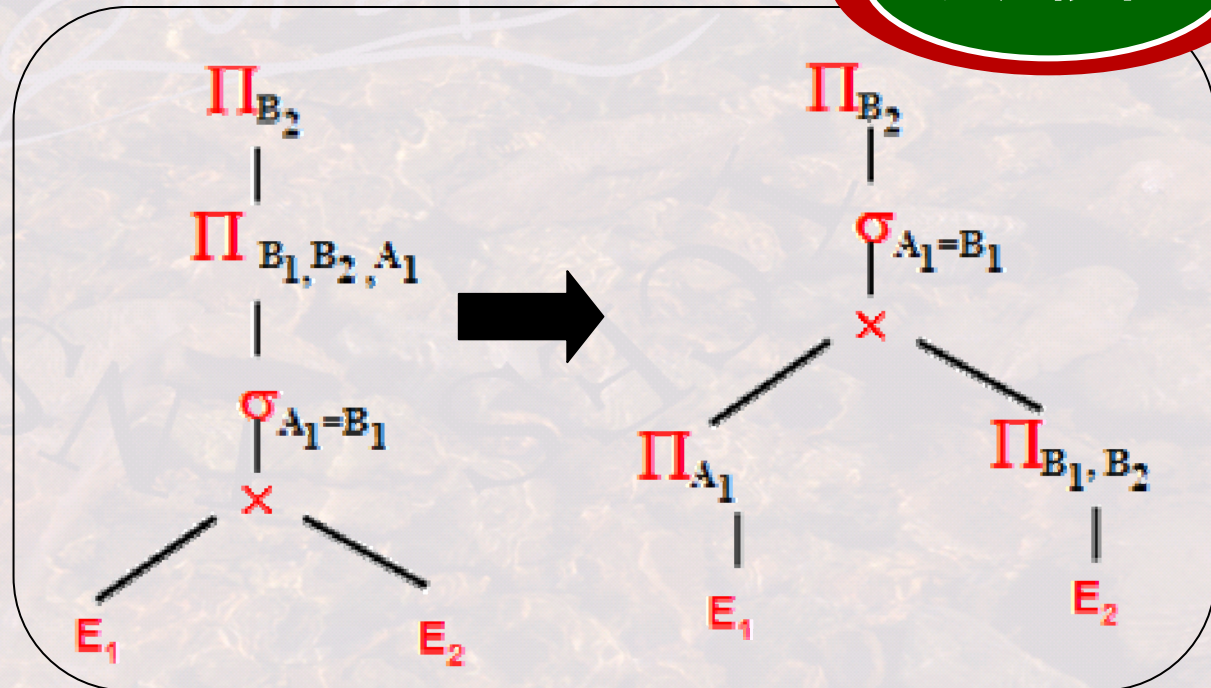
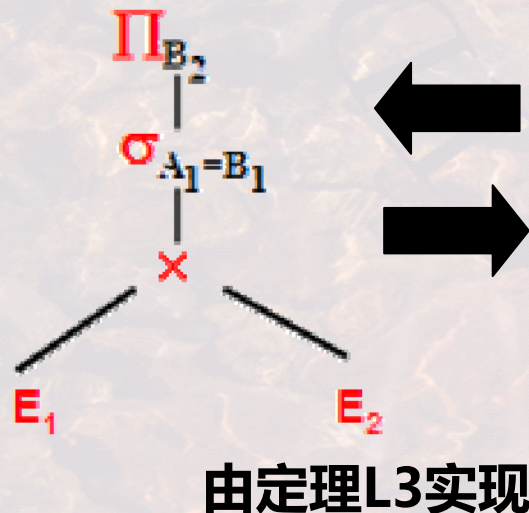
(3)哪些关系操作次序可以交换？

定理L7：投影和积的交换律

设 E_1, E_2 为两关系代数表达式， A_1, \dots, A_n 是出现在 E_1 或 E_2 中的一些属性，其中 B_1, \dots, B_m 出现在 E_1 中，剩余的属性 C_1, \dots, C_k 出现在 E_2 中，则有：

$$\pi_{A_1, \dots, A_n}(E_1 \times E_2) \equiv \pi_{B_1, \dots, B_m}(E_1) \times \pi_{C_1, \dots, C_k}(E_2)$$

“这是根本”



关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换？

定理L8：选择和并的交换律

设关系代数表达式 $E = E_1 \cup E_2$ ， F 是条件，则有：

$$\sigma_F(E_1 \cup E_2) \equiv \sigma_F(E_1) \cup \sigma_F(E_2)$$

注意：此定理要求 E_1 ， E_2 是并相容的

定理L9：选择和差的交换律

设关系代数表达式 $E = E_1 - E_2$ ， F 是条件，则有：

$$\sigma_F(E_1 - E_2) \equiv \sigma_F(E_1) - \sigma_F(E_2)$$

降低中间结果

关系代数操作次序交换的等价性

(3)哪些关系操作次序可以交换？



L10：投影和并的交换律

设关系代数表达式 $E = E_1 \cup E_2$ ， A_1, \dots, A_n 是 E 中的一些属性，则有：

$$\pi_{A_1, \dots, A_n}(E_1 \cup E_2) \equiv \pi_{A_1, \dots, A_n}(E_1) \cup \pi_{A_1, \dots, A_n}(E_2)$$

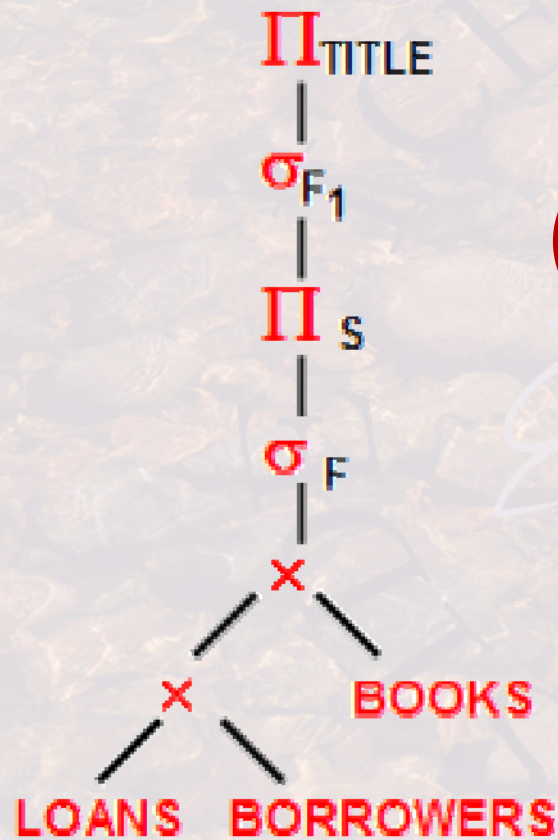
问：投影和集差运算有交换律吗？

$$\pi_{A_1, \dots, A_n}(E_1 - E_2) \equiv \pi_{A_1, \dots, A_n}(E_1) - \pi_{A_1, \dots, A_n}(E_2)$$

这个是不成立的，为什么呢？

关系代数操作次序交换的等价性

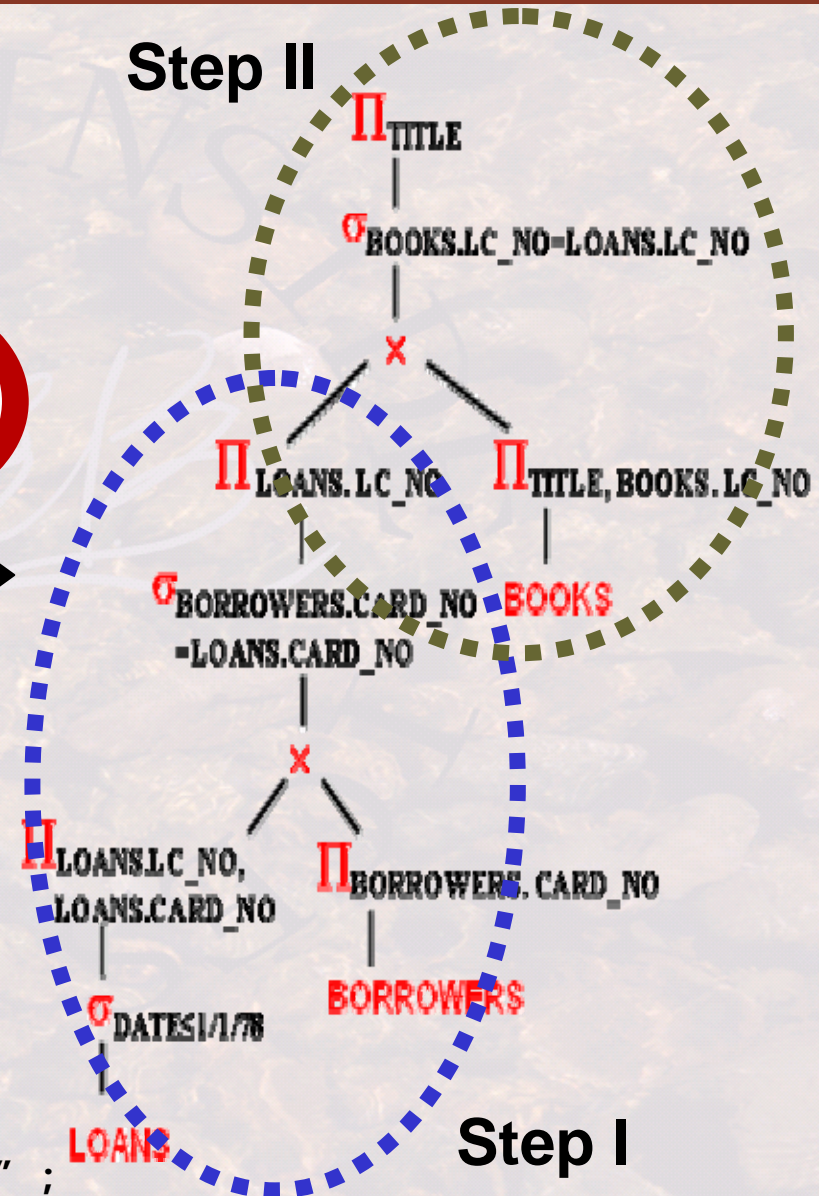
(4)怎样运用定理呢？



怎样运用
定理呢？
另文分解



Step II



其中： $F = F_2 \wedge F_3$ ； $F_1 = \text{"DATE} \leq 1/1/78\text{"}$ ；
 $F_2 = \text{"BORROWERS.CARD_NO} = \text{LOANS.CARD_NO"}$ ；
 $F_3 = \text{"BOOKS.LC_NO} = \text{LOANS.LC_NO"}$ 。

Step I

基于关系代数的查询优化算法及示例

战德臣

哈尔滨工业大学 教授·博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

Algorithm : 关系代数表达式的优化算法

Input : 一个关系代数表达式的语法树

Output : 计算该表达式的程序

Method :

(S1)依据定理L4, 把形如 $\sigma_{F1 \wedge F2 \wedge \dots \wedge F_n}(E)$ 的选择表达式变成串接形式

$\sigma_{F1}(\sigma_{F2}(\dots(\sigma_{F_n}(E))))$.

(S2)对每个选择, 依据定理L4至L9, 尽可能把它移至树的底部。

(S3)对每个投影, 依据定理L3, L7, L10和L5, 尽可能把它移至树的底部。如果一个投影是对某表达式所有属性进行的, 则去掉之。

(S4)依据定理L4至L5把串接的选择和投影组合为单个选择、单个投影, 或者一选择后跟一个投影。

(S5)对修改后的语法树, 将其内结点按以下方式分组:

每个二元运算结点(积、并、差、连接等)和其所有的一元运算直接祖先结点放在一组; 对于其后代结点, 若后代结点是一串一元运算且以树叶为终点, 则将这些一元运算结点放在该组中; 若该二元运算结点是笛卡儿积, 且其后代结点不能和它组合成等连接, 则不能将后代结点归入该组。

(S6)产生一个程序: 它以每组结点为一步, 但后代组先执行。

基于关系代数的查询优化算法及示例

(2)示例

查询：“查出1978年1月1日前被借出的所有书的书名”

“SELECT Title FROM XLOANS WHERE Date ≤ 1/1/78”

$\Pi_{\text{TITLE}}(\sigma_{\text{DATE} \leq \text{"1/1/78"}}(\text{XLOANS}))$

语法树



**CREATE VIEW XLOANS as SELECT
* FROM LOANS, BORROWERS,
BOOKS WHERE
BORROWERS.CARD_NO =
LOANS.CARD_NO AND
BOOKS.LC_NO = LOANS.LC_NO**



由树叶向树
根反映了操
作先后次序

其中： $F = F_2 \wedge F_3$ ； $F_1 = \text{"DATE} \leq 1/1/78\text{"}$ ；
 $F_2 = \text{"BORROWERS.CARD_NO = LOANS.CARD_NO"}$ ；
 $F_3 = \text{"BOOKS.LC_NO = LOANS.LC_NO"}$ 。

$S = \text{TITLE, AUTHOR, PNAME, LC_NO, NAME, ADDR, CITY, CARD_NO, DATE}$

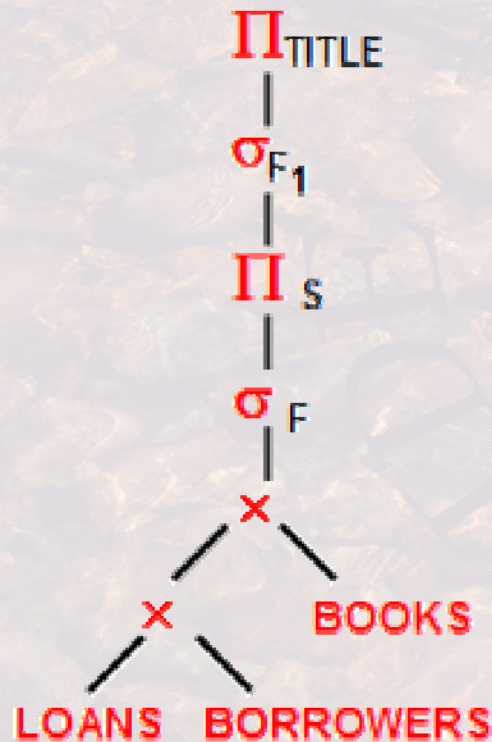
基于关系代数的查询优化算法及示例

(2)示例

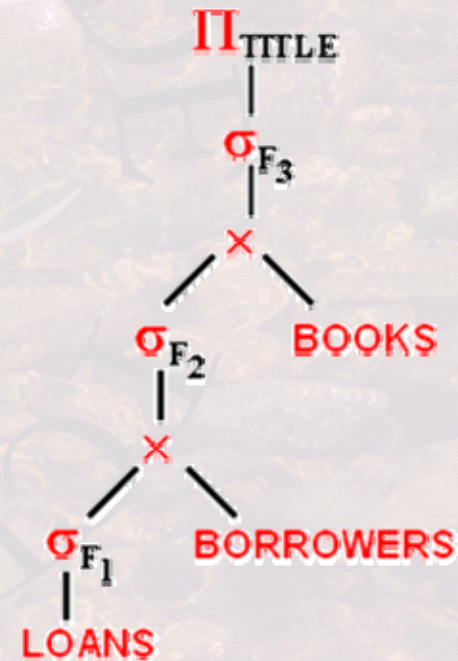
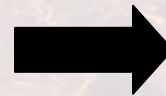
(1)依据定理L4，把形如 $\sigma_{F_1 \wedge F_2 \wedge \dots \wedge F_n}(E)$ 的选择表达式变成串接形式

$\sigma_{F_1}(\sigma_{F_2}(\dots(\sigma_{F_n}(E))))$.

(2)对每个选择，依据定理L4至L9，尽可能把它移至树的底部。



- $\sigma_{F_2 \wedge F_3} = \sigma_{F_2}(\sigma_{F_3})$
- σ_{F_1} 通过交换移动到底部
- σ_{F_2} 通过交换移动到底部
- $\Pi_{TITLE}(\Pi_S) = \Pi_{TITLE}$



其中: $F = F_2 \wedge F_3$; $F_1 = \text{"DATE} \leq 1/1/78\text{"}$;
 $F_2 = \text{"BORROWERS.CARD_NO} = \text{LOANS.CARD_NO"}$;
 $F_3 = \text{"BOOKS.LC_NO} = \text{LOANS.LC_NO"}$.

基于关系代数的查询优化算法及示例

(2)示例

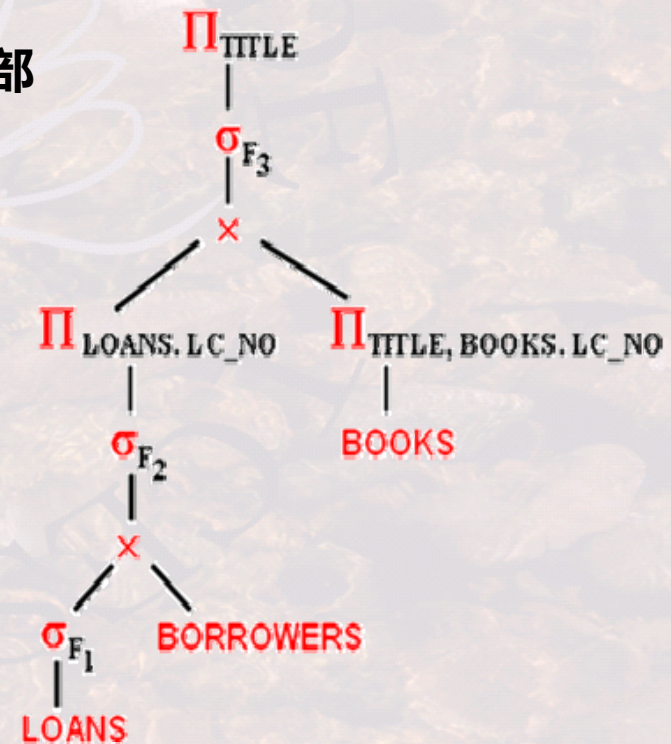
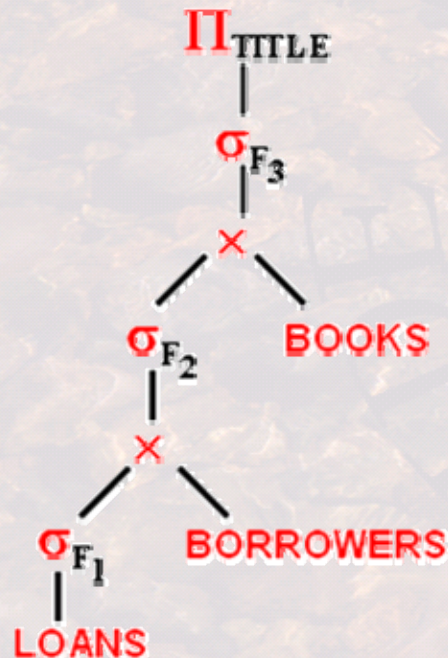
(S3)对每个投影，依据定理L3，L7，L10和L5，尽可能把它移至树的底部。
如果一个投影是对某表达式所有属性进行的，则去掉之。

• $\Pi_{\text{TITLE}} = \Pi_{\text{TITLE}} (\Pi_{\text{TITLE}, \text{BOOKS.LC_NO}, \text{LOANS.LC_NO}})$

使其包含 F_3 涉及的属性

• $\Pi_{\text{TITLE}, \text{BOOKS.LC_NO}}$ 移动到底部

• $\Pi_{\text{LOANS.LC_NO}}$ 移动到底部

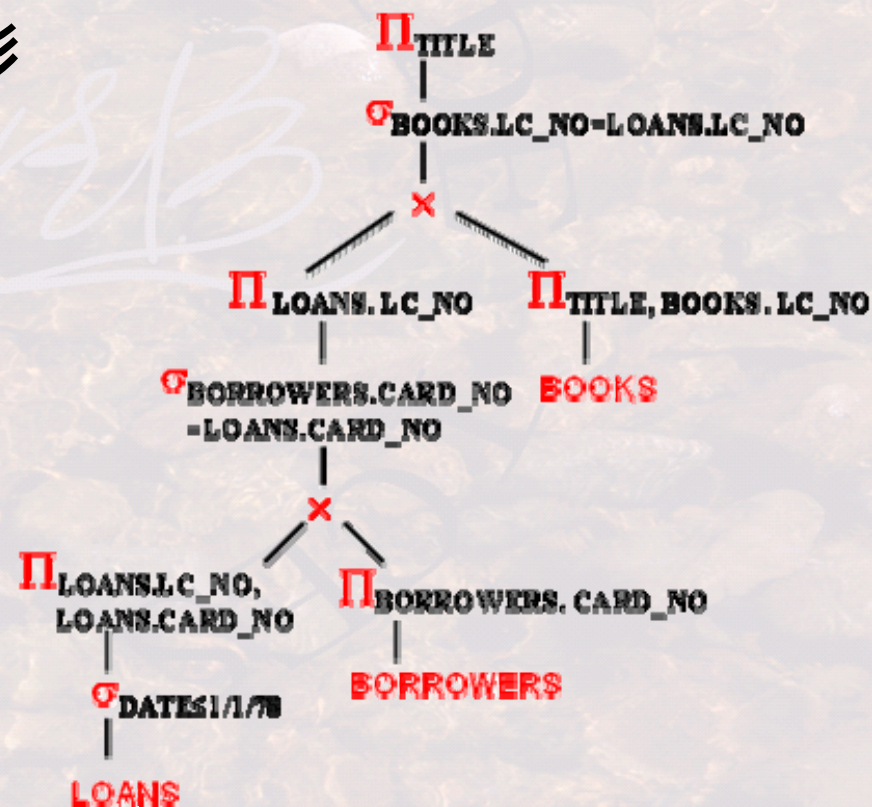
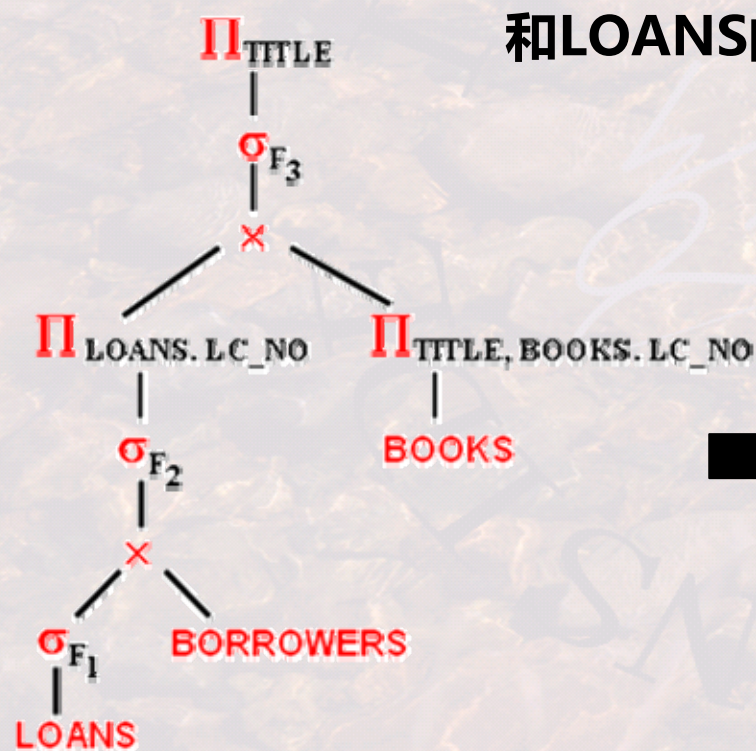


其中: $F = F_2 \wedge F_3$; $F_1 = \text{"DATE} \leq 1/1/78\text{"}$;
 $F_2 = \text{"BORROWERS.CARD_NO} = \text{LOANS.CARD_NO"}$;
 $F_3 = \text{"BOOKS.LC_NO} = \text{LOANS.LC_NO"}$.

基于关系代数的查询优化算法及示例

(2)示例

•类似处理BORROWERS 和LOANS的投影



其中: $F = F_2 \wedge F_3$; $F_1 = \text{"DATE} \leq 1/1/78\text{"}$;
 $F_2 = \text{"BORROWERS.CARD_NO} = \text{LOANS.CARD_NO\text{"}$;
 $F_3 = \text{"BOOKS.LC_NO} = \text{LOANS.LC_NO\text{"}$.

(S5)按以下方式分组：

每个二元运算结点

(积、并、差、连接等)

和其所有一元运算直接

祖先结点放在一组；对

于其后代结点，若后代

结点是一串一元运算且

以树叶为终点，则将这

些一元运算结点放在该

组中；若该二元运算结

点是笛卡儿积，且其后

代结点不能和它组合成

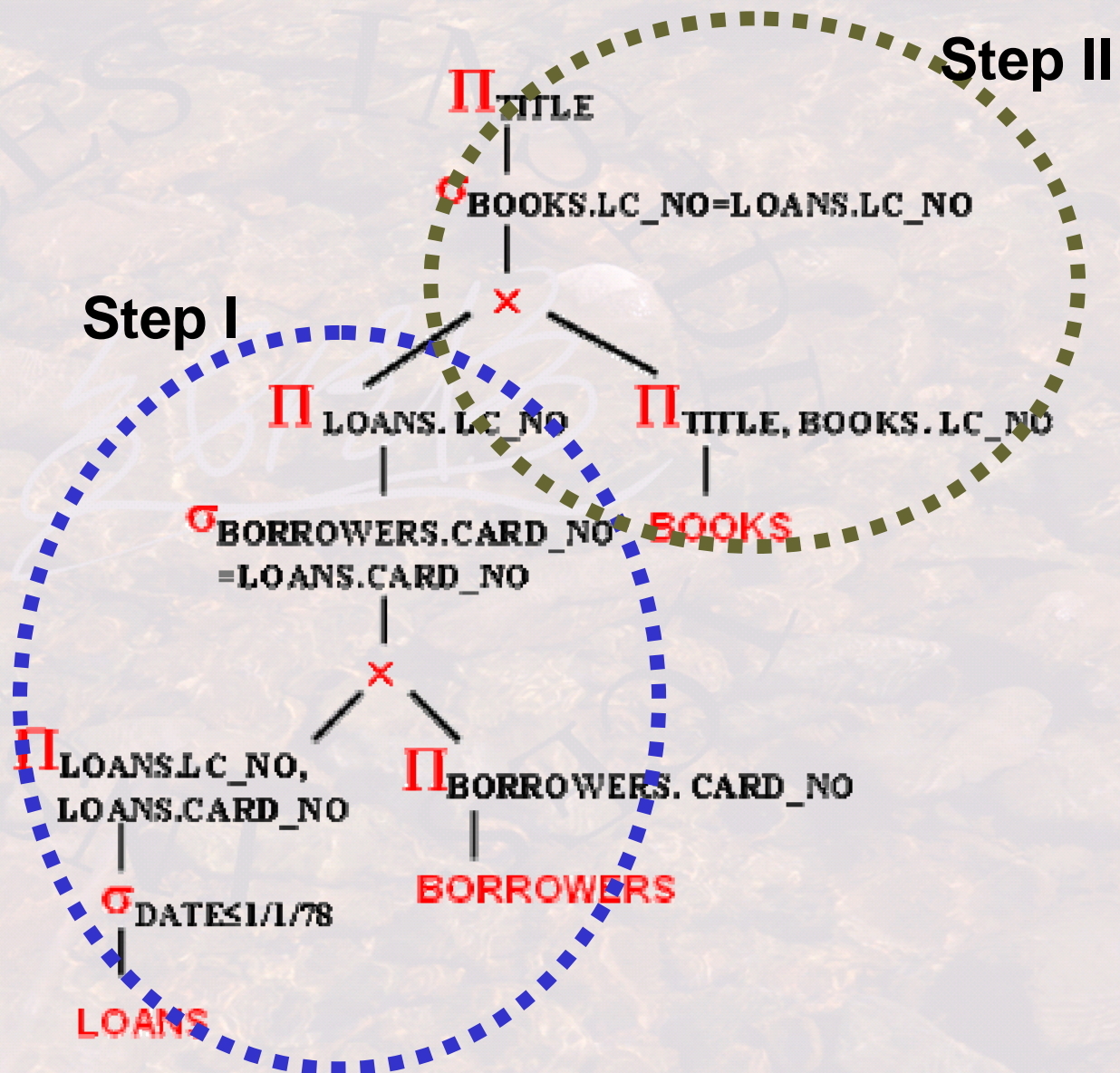
等连接，则不能将后代

结点归入该组。

(S6)产生一个程序：它

以每组结点为一步，但

后代组先执行。



基于关系代数的查询优化算法及示例

(3)复杂SQL语句的处理问题

“SELECT A, SUM(B) FROM R1, ..., Rm WHERE Condition1
GROUP BY A HAVING Condition 3 ORDER BY A”;

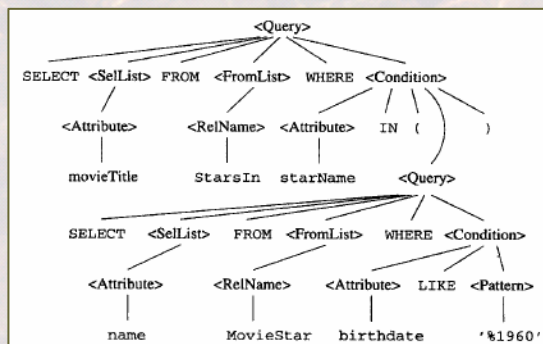


教材中关系代数的
分组聚集运算符



$\tau_A(\Pi_{A,X}(A \underset{g}{\bowtie} \text{SUM}(B)(\gamma_A(\sigma_{\text{condition1}}(R_1 \times \dots \times R_m))))))$

```
SELECT movieTitle
FROM StarsIn
WHERE starName IN (
  SELECT name
  FROM MovieStar
  WHERE birthdate LIKE '%1960'
);
```



分组、聚集和排序运算有交换律吗？

嵌套查询如何处理呢？

同学自
学研究

物理层查询优化？

(暨物理实现算法选择与次序优化)

战德臣

哈尔滨工业大学 教授·博士生导师
黑龙江省教学名师
教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

为什么要物理查询优化？

一个例子： $\sigma_{Cname = \text{“数据结构”}}$ (Course) 的执行方案

方案1：表空间扫描方法

直接对Course表进行扫描，从第一条检索到最后一条，将满足条件的记录找出

方案2：利用Course上的Cname排序索引的方法

利用排序索引可以进行诸如二分查找等快速检索，找到相应的索引项，依据指针将满足条件的记录找出

➤当条件更复杂时，可选择的方案还会更多

究竟用哪一个算法的程序来执行？为什么如此选择？

(1)物理查询优化—总体思路

物理查询运算符

□获取关系元组的操作

- ✓ TableScan(R) ---表空间扫描算法
- ✓ SortTableScan(R)---表空间扫描排序算法
- ✓ IndexScan(R)---索引扫描算法
- ✓ SortIndexScan(R)---索引扫描排序算法

物理查询运算符通常是关系代数操作符的一个特定实现

□关系操作的各种实现算法

- ✓ $\sigma_F(R)$, $\pi_\alpha(R)$, $\delta(R)$, $\gamma(R)$, $\tau(R)$, 集合上的操作 : \cup_S , \cap_S , $-_S$, 包上的操作 : \cup_B , \cap_B , $-_B$, 积, 连接 : PRODUCT, JOIN
- ✓ 一趟算法、两趟算法 ; 基于索引算法、基于散列算法、基于排序算法 ;
- 迭代器构造--流水化、物化 ;

物理层查询优化(暨物理实现算法选择与次序优化)

(1)物理查询优化—总体思路

SELECT A1, ..., An FROM R1, ..., Rm
WHERE Cond;

编译

$\pi_{A1, \dots, An} (\sigma_{Cond} (R1 \times \dots \times Rm))$

语法优化 逻辑层优化

$\pi_{A1, \dots, An} (\sigma_{Cond} (\pi_{a1} (\sigma_{Cond1} (R1)) \times \dots \times \pi_{am} (\sigma_{Condm} (Rm)))$

逻辑查
询计划

执行优化 物理层优化

为每一个关系代数操作**选取**优化的执行层**例行程序**, 形成物理查询计划

物理查询计划
--基于不同算法的
实现程序构造

执行引擎: 依查询计划调用相应的**例行程序**进行处理, 并返回结果

物理优化/执行优化

获取数据库的相关信息
(定期统计)

选取相应的执行层例行程序

依据相关信息进行代价估算, 并选择代价最少的例行程序及确定相应的参数

形成查询计划: 以基本的例行程序为基本, 确定这些例行程序的执行顺序

实现同一关系操作的不同例行程序库

物理层查询优化(暨物理实现算法选择与次序优化)

(2)如何衡量一个物理查询计划的好与坏?



DBMS如何衡量物理查询计划的优劣呢？

衡量I/O访问次数

衡量CPU的占用时间

内存使用代价(与缓冲区数目与大小的匹配)

中间结果存储代价

计算量(如搜索记录、合并记录、排序记录、字段值的计算等)

网络通信量

... ..

➤ **依据什么信息来计算这些方案的上述各种指标呢？**

依据数据库的一些统计信息---存放在数据字典或系统目录中的

- T_R 或 $T(R)$: 关系R的元组数目 ;
- B_R 或 $B(R)$: 关系R的磁盘块数目 ;
- I_R 或 $I(R)$: 关系R的每个元组的字节数 ;
- f_R 或 $f(R)$: R的块因子 , 即一块能够存储的R的元组数目
- $V(A, R)$: R中属性A出现不同值的数目 , 即 $\Pi_A(R)$ 的数目.
- $SC(A, R)$: R中属性A的选择基数 , 满足A上等值条件的平均记录数
- b : 每个磁盘块的字节数 ;
-



➤ DBMS依据上述统计信息对DB操作的各种物理查询计划进行评估 , 以确定最优的计划予以执行。

➤ 上述信息如何获得呢 ?

物理层查询优化(暨物理实现算法选择与次序优化)

(4)如何收集这些信息?



- 当一个表装入内存和创建索引的时候，统计信息不是被自动收集的，必须由DBA使用特定的命令来完成信息统计，这些命令就是收集统计信息并把其存入系统目录中的实用程序
- 随着表的更新操作，统计信息可能会过时，过时的统计信息会使DBMS确定方案时决策错误，因此要求DBA定期的对有频繁更新操作的Table进行统计
- DBA要熟悉统计信息收集命令的使用，并定期执行
- IBM DB2使用Runstats收集统计信息

```
RUNSTATS ON TABLE username.tablename
[ WITH DISTRIBUTION [ AND DETAILED ]
{ INDEXES ALL | INDEX indexname} ] ;
```
- 例如，收集SCT数据库的Student表的统计信息

```
Runstats on table SCT.student ;
```


物理层查询优化(暨物理实现算法选择与次序优化)

(4)如何收集这些信息?



- Oracle使用Analyze命令收集统计信息并将其放入系统表中

ANALYZE { INDEX | TABLE | CLUSTER }

{ indexname | tablename | clustername }

COMPUTE STATISTICS

{ FOR TABLE | FOR ALL [INDEXED] COLUMNS [SIZE n] } ;

- 例如，收集SCT数据库的Student表的统计信息

Analyze table **student** compute statistics for table ;

有了统计信息，
如何进行**代价估**
算呢？下回分解

$R_1 \text{ Join } R_2 \text{ Join } \dots \text{ Join } R_n$
的连接顺序确定问题？
同学自学相关内容

物理查询计划的形成：

- 理想: 寻找最优的查询计划；
- 现实: 避免最差的查询计划

代价估算

战德臣

哈尔滨工业大学 教授·博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

代价估算

已知表达式E中的各个关系的统计信息

- T_R 或 $T(R)$: 关系R的元组数目 ;
- B_R 或 $B(R)$: 关系R的磁盘块数目 ;
- I_R 或 $I(R)$: 关系R的每个元组的字节数 ;
- f_R 或 $f(R)$: R的块因子, 即一块能够存储的R的元组数目
- $V(R, A)$: R中属性A出现不同值的数目, 即 $\Pi_A(R)$ 的数目.
- $SC(R, A)$: R中属性A的选择基数, 满足A上等值条件的平均记录数

“给定一个表达式E, 如何计算E的元组数目 $T(E)$ 以及属性A上不同值的数目 $V(E, A)$?”

- 在E实际获得之前计算 $T(A), V(E, A)$ 等是很难的事情 ;
- 因而, 要“估算”, 代价估算

“估算一个投影 $\pi_L(R)$ 的大小”

- 简单: $T(\pi_L(R)) = T(R)$
- 投影运算只是对列有所取舍, 并未对行有所变化, 如并未消除重复
- 投影运算并未减少行数, 但可能有效地减少了存储结果关系的块数
- 例如: 磁盘块大小=1024 Byte

R的元组长度=120 Byte, 8元组/块, $T(R)=10,000$, 则

$$B(R) = 10000/8 = 1250;$$

$\pi_L(R)$ 的元组长度=20 Byte, 50元组/块, 则

$$B(\pi_L(R)) = 10000/50 = 200;$$

“估算选择运算 $S = \sigma_{A=c}(R)$ 的大小”

- $T(S)$ 介于 0 to $T(R) - V(R, A) + 1$ 之间
 - 最多：A属性不同值的元组都只存在一个，剩余的都是 $A=c$ 的元组
- 估计: $T(S) = T(R) / V(R, A)$
 - A属性不同值的元组数假设是平均分布的
- 当不知道 $V(R, A)$ 时，估计： $T(S) = T(R)/10$.

“估算选择运算 $S = \sigma_{A < c}(R)$ 的大小”

- $T(S)$ 介于 0 to $T(R)$ 之间
 - 最多：所有元组都满足条件
- 估计: $T(S) = T(R)/2$
 - 直觉，应有一半的元组，
- 实际应用的估计： $T(S) = T(R)/3$

“估算选择运算 $S = \sigma_{A=10 \text{ AND } B < 20}(R)$ 的大小”

➤估计 : $T(S) = T(R)/(V(R, A)*3)$

--- $\sigma_{A=10 \text{ AND } B < 20}(R) = \sigma_{B < 20}(\sigma_{A=10}(R))$

--- $A=10$, 得出 $T(S) = T(R)/V(R, A)$;

--- $B < 20$, 得出 $T(S) = T(S)/3$

“估算选择运算 $S = \sigma_{C1 \text{ or } C2}(R)$ 的大小”

➤估计 : $T(S) = n(1 - (1 - m_1/n)(1 - m_2/n))$

--- R 有 n 个元组 , 其中有 m_1 个满足 $C1$, 有 m_2 个满足 $C2$

--- $(1 - m_1/n)$ 是不满足 $C1$ 的那些元组 , $(1 - m_2/n)$ 是不满足 $C2$ 的那些元组

--- 两数之积是不在 S 中的那部分 R 的元组 , 1 减去这个积就是属于 S 的那部分元组出现的概率。

“估算选择运算 $S = \sigma_{A=10 \text{ OR } B<20}(R)$ 的大小”

➤估计 : $T(S) = n(1 - (1 - m_1/n)(1 - m_2/n))$

--- $n = T(R) = 10000$, $V(R, A) = 50$,

$m_1 = T(R)/V(R, A) = 10000/50 = 200$;

$m_2 = T(R)/3 = 10000/3 \approx 3333$

(有 m_1 个满足 C_1 , 有 m_2 个满足 C_2 ,

$(1 - m_1/n)(1 - m_2/n)$ 不满足这个条件的元组的概率

$1 - (1 - m_1/n)(1 - m_2/n)$ 满足这个条件的元组的概率)

---简单估计 : $T(S) = T(R)/3 = 10000/3 \approx 3333$

---复杂估计 :

$T(S) = 10000 * (1 - (1 - 200/10000)(1 - 3333/10000)) \approx 3466$

“估算连接运算 $S = R(X,Y) \text{ Natural Join } S(Y,Z)$ 的大小”

➤估计： $T(S) = T(R)T(S) / \max(V(R,Y), V(S,Y))$

---假定 $V(R,Y) \geq V(S,Y)$, R中元组r和S中元组有相同Y值的概率 = $1/V(R,Y)$

---假定 $V(R,Y) < V(S,Y)$, R中元组r和S中元组有相同Y值的概率 = $1/V(S,Y)$

---则，在Y上相等的概率 = $1/\max(V(R,Y), V(S,Y))$

➤例： $T(R)=10000, T(S)=50000, V(R, Y) = 500, V(S, Y)=1000$

估计： $T(S)=10000*50000/1000=500000$ 。

➤例： $T(R)=10000, T(S)=50000, V(R, Y) = 2000, V(S, Y)=1000$

估计： $T(S)=10000*50000/2000=250000$ 。

代价估计

- $T(R)$ -- R 的元组个数, $V(R, A)$ — R 中属性 A 上出现的不同值的数目
- 判断满足单一条件元组出现的概率
 - ✓ 出现等于某一个值的概率 = $1 / V(R, A)$, 或者简单的概率 = $1/10$
 - ✓ 出现不等于某一个值的概率 = $1 - 1/V(R, A)$, 或者简单的概率 = $1-1/10$
 - ✓ 出现小于或不等于某一个值的概率直觉上 = $1/2$, 实际处理概率= $1/3$
- 判断满足多个条件的元组出现的概率
 - ✓ 如果是“与”, 则将满足两个条件的概率相乘
 - ✓ 如果是“或”, 则= $(1-(1-m_1/n)(1-m_2/n))$, 不出现满足条件1的元组的概率 $(1-m_1/n)$, 不出现满足条件2的元组的概率 $(1-m_2/n)$, 二者相乘是不同时出现的概率, 则 $1- (1-m_1/n)(1-m_2/n)$ 即为去掉不同时出现的概率, 即为或条件的概率。
- 复杂的表达式可以依上原则进行估算, 确定估算公式。

回顾本讲学了什么？

战德臣

哈尔滨工业大学 教授·博士生导师

黑龙江省教学名师

教育部大学计算机课程教学指导委员会委员

Research Center on **I**ntelligent
Computing for **E**nterprises & **S**ervices,
Harbin **I**nstitute of **T**echnology

回顾本讲学习了什么?

SELECT A1, ..., An FROM R1, ..., Rm
WHERE Cond;

编译

$\pi_{A1, \dots, An} (\sigma_{Cond} (R1 \times \dots \times Rm))$

语法优化 逻辑层优化

$\pi_{A1, \dots, An} (\sigma_{Cond} (\pi_{a1} (\sigma_{Cond1} (R1)) \times \dots \times \pi_{am} (\sigma_{Condm} (Rm)))$

逻辑查询计划

执行优化 物理层优化

为每一个关系代数操作**选取**优化的执行层**例行程序**, 形成物理查询计划

物理查询计划
--基于不同算法的实现程序构造

执行引擎: 依查询计划调用相应的**例行程序**进行处理, 并返回结果

逻辑层优化

关系代数
操作次序
优化

物理层优化

代价估算

算法选择与
装配次序

DBMS核
心技术