



University of Science and Technology of
China

软件学院 2015 级高级数 据库技术（金培权 - 数据 库系统实现）

Homework 1

给定下面的关系：图书（图书号，书名，作者，单价，库存量），读者（读者号，姓名，工作单位，地址），借阅（图书号，读者号，借期，还期，备注） 注：还期为 **NULL** 表示该书未还。

使用关系代数表达式实现下列 **1 – 3** 小题

检索读者 **Rose** 的工作单位和地址




$$\Pi_{\text{工作单位, 地址}} (\sigma_{\text{姓名} = \text{Rose}'} (\text{读者}))$$

检索读者 **Rose** 所借阅读书（包括已还和未还图书）的图书名和借期

$$\Pi_{\text{书名, 借期}} (\sigma_{\text{姓名} = \text{Rose}'} (\text{图书} \bowtie \text{借阅} \bowtie \text{读者}))$$

Homework 1

检索未借阅图书的读者姓名

 (读者)
姓名姓名 -  读者借阅) 

用 **SQL** 语言完成 4 — 8 小题:

查询语句结果可以计算如下:

1. 取 **FROM** 子句中列出的各个关系的元组的所有可能的组合
2. 将不符合 **WHERE** 子句中给出的条件的元组去掉
3. 如果有 **GROUP BY** 子句, 则将剩下的元组按 **GROUP BY** 子句中给出的属性值分组
4. 如果有 **HAVING** 子句, 则按照 **HAVING** 子句中给出的条件检查每一组, 去掉不符合条件的组
5. 按照 **SELECT** 子句的说明, 对于指定的属性和属性上的聚集 (例如一组中的和) 计算出结果元组
6. 按照 **ORDER BY** 子句中的属性列的值对结果元组进行排序

Homework 1

检索 **Ullman** 所写的书的书名和单价

```
SELECT 书名, 单价  
FROM 图书  
WHERE 作者 = 'Ullman'
```

检索读者“李林”借阅未还的图书的图书号和书名

```
SELECT 图书号, 书名  
FROM 图书, 读者, 借阅  
WHERE 图书.图书号 = 借阅.图  
书号 AND 读者.读者号 = 借阅.  
读者号 AND 读者.姓名 = “李林”  
  
AND 借阅.还期 = NULL ;
```

Homework 1

检索借阅图书数目超过 **3** 本的读者姓名

```
SELECT 姓名  
FROM 读者, 借阅  
WHERE 借阅.读者号 = 读者.读  
者号  
GROUP BY 读者号  
HAVING COUNT(*) > 3;
```

Homework 1

检索没有借阅读者“李林”所借的任何一本书的读者姓名和读者号

```
SELECT 姓名, 读者号
FROM 读者, 借阅
WHERE 借阅.读者号 = 读者.读者号 AND
      借阅.图书号 NOT IN
      (SELECT 图书号
       FROM 借阅, 读者
       WHERE 借阅.读者号 = 读者.读者号
       AND 读者.姓名 = '李林');
```

Homework 1

检索书名中包含“**Oracle**”的图书书名及图书号。

```
SELECT 图书号, 书名  
FROM 图书  
WHERE 书名 LIKE '%Oracle%';
```


Homework 1

现有如下关系模式： $R(A, B, C, D, E, F, G)$ ， R 上存在的函数依赖有： $AB \rightarrow E$ ， $A \rightarrow B$ ， $B \rightarrow C$ ， $C \rightarrow D$ 。
该关系模式满足第几范式吗？为什么？

满足 **1NF** 范式。因为每一个属性值都只含有一个值，所以满足 **1NF**。由于 R 的候选码为 (A, F, G) ，而 B 、 C 、 D 局部依赖于 A ，所以不满足 **2NF**。

如果将关系模式 R 分解为： $R_1(A, B, E)$ ， $R_2(B, C, D)$ ， $R_3(A, F, G)$ ，该数据库模式最高满足第几范式？

最高满足 **2NF** 范式。因为对于模式 R_2 ， $B \rightarrow C$ ， $C \rightarrow D$ ，存在传递依赖，所以不满足 **3NF**。

Homework 1

请将关系模式 **R** 无损连接并且保持函数依赖地分解到 **3NF**，要求给出具体步骤。

先将 **R** 保持函数依赖地分解到 **3NF**。

1. 求出 **R**<**U**,**F**> 的最小函数依赖集（仍记为**F**）
2. 把所有不在**F**中出现的属性组成一个关系模式**R'**，并在**U**中去掉这些属性(剩余属性仍记为**U**)
 1. 求 **R** 上函数依赖集 **F** 的最小 **FD** 集合：
F = { **AB**→**E**, **A**→**B**, **B**→**C**, **C**→**D** } ; **U** = {**A**,**B**,**C**,**D**,**E**}
 2. 所有不在 **F** 中出现的属性组成 **R'**(**F**,**G**)

Homework 1

3. 若**F**中存在 $\mathbf{X} \rightarrow \mathbf{A}$ ，且 $\mathbf{XA}=\mathbf{U}$ ，则输出 $\mathbf{R(U)}$ 和 $\mathbf{R'}$ ，算法结束，否则
4. 对**F**按相同的左部分组，将所有 $\mathbf{X} \rightarrow \mathbf{A1}, \mathbf{X} \rightarrow \mathbf{A2}, \dots, \mathbf{X} \rightarrow \mathbf{Ak}$ 形式的**FD**分为一组，并将每组涉及的所有属性作为一个关系模式输出。若某个关系模式 \mathbf{Ri} 的属性集是另一个关系模式的属性集的子集，则在结果中去掉 \mathbf{Ri} 。设最后得到关系模式 $\mathbf{R1}, \mathbf{R2}, \dots, \mathbf{Rk}$ ，则 $\mathbf{p}=\{\mathbf{R1}, \mathbf{R2}, \dots, \mathbf{Rk}, \mathbf{R'}\}$ 一个保持函数依赖的分解，并且满足**3NF**

3. 对 **F** 按相同的左部分组，并去除子集，得到

:

$\mathbf{p} = \{\mathbf{R1} \ (\mathbf{A,B,E}) ; \ \mathbf{R2} \ (\mathbf{B,C}) ; \ \mathbf{R3(C,D)} ;$
 $\mathbf{R'(F,G)} \}$

Homework 1

无损连接且保持函数依赖地分解到 **3NF**

1. 首先用算法1求出**R**的保持函数依赖的**3NF**分解，设为 $q=\{R_1, R_2, \dots, R_k\}$
2. 设**X**是**R**的码，求出 $p=q \cup \{R(X)\}$
3. 若**X**是 q 中某个**R_i**的子集，则在 p 中去掉**R(X)**
4. 得到的 p 就是最终结果

Homework 1

4. 由于R的主码是 (A,F,G) , 所以:

$p = q \cup \{R(X)\} = \{R1(A,B,E), R2(B,C), R3(C,D), R'(F,G), R4(A,F,G)\}$

5. 而 R' 是 R4 的子集, 所以从 p 中去掉 R' (F,G)

6. $p = \{R1(A,B,E), R2(B,C), R3(C,D), R4(A,F,G)\}$ 为最终结果

Homework 1

Megatron 777 磁盘具有以下特性：

- (1) 有 **10** 个盘面，每个盘面有 **100000** 个磁道；
- (2) 磁道平均有 **1000** 个扇区，每个扇区为 **1024** 字节；
- (3) 每个磁道的 **20%** 用于间隙；
- (4) 磁盘旋转为 **10000** 转 /min ；

回答下列有关 **Megatron 777** 的问题：
ms

磁盘的容量是多少？

$$\text{磁盘容量} = 10 * 100000 * 1000 * 1024 \text{B} = 10^9 \text{KB}$$

Homework 1

如果磁道是在直径 **3.5** 英寸的圆面上，那么一个磁道的扇区中的平均位密度是多少？

我们选取中间磁道来计算平均位密度，中间磁道的直径为 **3.5inch/2**，该磁道的周长为 **$(3.5\pi/2)\text{inch}$** ，扇区所占的周长是 **$80\%\times(3.5\pi/2)\text{inch}$** 。同时，每个磁道的容量是 **$1000\times1024\times8$ bits** 所以一个磁道的扇区中的平均位密度是
 $(1000\times1024\times8) \text{ bits}/(80\%\times3.5\pi/2)\text{inch} = 1861733.6 \text{ bpi}$

最大寻道时间是多少？

最大寻道时间 **$1 + 0.0002 \times 99\,999 = 21\text{ms}$**

Homework 1

最大旋转等待时间是多少？

最大旋转等待时间： $60 \times 1000\text{ms} / 10\,000 = 6\text{ms}$

如果一个块是 **65536** 字节（即 **64** 扇区），一个块的传输时间是多少？

如果一个块是 **65536** 字节（即 **64** 扇区），则磁头必须越过 **64** 个扇区以及扇区之间的 **63** 个间隙。

需要的时间为：

$$\begin{aligned} & 64 \text{ (扇区 + 间隙)} - 1 \text{ (间隙)} \\ &= 64 * \left(\frac{6}{1000} \right) - \left(\frac{6}{1000} \right) * 0.2 \\ &= 0.3828\text{ms} \end{aligned}$$

Homework 1

平均寻道时间是多少？

$$1 + 0.0002 * 99999 / 3 = 7.67\text{ms}$$

平均旋转等待时间是多少？

$$\text{平均旋转等待时间为: } 6\text{ms} / 2 = 3\text{ms}$$

Homework2

假设一条记录有如下顺序的字段：一个长度为 **23** 的字符串，一个 **2** 字节整数，一个 **SQL** 日期，一个 **SQL** 时间（无小数点）。

字段可在任何字节处开始？

一个 **SQL** 日期是 **10** 个字节的字符串，一个 **SQL** 时间是 **8** 个字节的字符串。

因为是任何字节处开始的，所以记录长度需要 $23+2+10+8=43$ 字节。

Homework2

字段必须在 **8** 的倍数的字节处开始？

如果考虑记录首部的话，假设首部有多字段长度 **K1**，**K2**，...**Kn**考虑对齐，每个字段都要如此处理：

K1的长度变成应该为 $\left\lceil \frac{K1}{8} \right\rceil * 8$ (取上整)。

因为必须是**8**的倍数，而长度为**23**的字符串需要分配**24**个字节，**2**字节的整数需要分配**8**字节，**SQL**日期需要分配**16**个字节，**SQL**时间需要分配**8**个字节。
所以：**24+8+16+8=56**字节

Homework2

字段必须在 **4** 的倍数的字节处开始？

因为必须是 **4** 的倍数，而长度为 **23** 的字符串需要分配 **24** 个字节，**2** 字节的整数需要分配 **4** 个字节，**SQL** 日期需要分配 **12** 个字节，**SQL** 时间需要分配 **8** 个字节。所以： **$24+4+12+8=48$** 字节。

Homework2

假设我们有 **4096** 字节块，块中存储 **200** 字节长的记录。块首部由一个偏移量表组成，它使用 **2** 字节长指针指向块内记录。通常，每天向每块插入两条记录，删除一条记录。删除记录必须使用一个“删除标记”代替它的指针，因为可能会有悬挂指针指向它。更明确地说，假设任何一天删除记录总发生在插入之前。如果刚开始时块是空的，多少天之后，不再有插入记录的空间？

第一天，只做插入操作，插入两条记录，同时使用 **2** 个指针指向记录，总计增加了 $2 \times (2 + 200) = 404$ 字节。

Homework2

之后的每一天都先删除一条记录再增加两条记录，净增 $404 - 200 = 204$ 字节。由于 $(4096 - 404) / 204 = 18 \cdots 20$ ，即在 $1 + 18 = 19$ 天之后，块中剩余空间为 **20** 字节。

在第 **20** 天，先删除一条记录，余下 $200 + 20 = 220$ 字节空间，这时候只能够再插入一条记录（**202** 字节）。

Homework2

一个病人记录包含以下定长字段：病人的出生日期，社会保险号码，病人 **ID**，每一个字段都是 9 字节长。它还有下列变长字段：姓名，住址和病史。如果记录内一个指针需要 8 字节，记录长度是一个 2 字节整数，不包含变长字段空间，这条记录需要多少字节？你可以假设不需要对字节进

记录长度	出生日期	保险号码	病人 ID	住址指针	病史指针	姓名	住址	病史
------	------	------	-------	------	------	----	----	----

Homework2

定长字段有 **3** 个，每个有 **9** 个字节长，所以需要 **$3 \times 9 = 27$** 字节。

而记录的首部需要写入记录的长度和指向所有除第一个以外的变长字段起始处的指针。而记录长度 **2** 字节，指向“住址”的指针 **8** 字节，指向“病史”的指针 **8** 字节。所以一共需要 **$27 + 2 + 8 + 8 = 45$** 字节。

Homework3

5.4.8 习题

习题 5.4.1 下面是 4 个关系 W、X、Y、Z 的关键统计值：

$W(a, b)$	$X(b, c)$	$Y(c, d)$	$Z(d, e)$
$T(W) = 400$	$T(X) = 300$	$T(Y) = 200$	$T(Z) = 100$
$V(W, a) = 50$	$V(X, b) = 60$	$V(Y, c) = 50$	$V(Z, d) = 10$
$V(W, b) = 40$	$V(X, c) = 100$	$V(Y, d) = 20$	$V(Z, e) = 50$

估计下列表达式结果关系的大小：

- | | | |
|---------------------------------------|---------------------------------------|------------------------------|
| a) $W \bowtie X \bowtie Y \bowtie Z$ | b) $\sigma_{a=10}(W)$ | c) $\sigma_{c=20}(Y)$ |
| d) $\sigma_{c=20}(Y) \bowtie Z$ | e) $W \times Y$ | f) $\sigma_{d>10}(Z)$ |
| g) $\sigma_{a=1 \text{ AND } b=2}(W)$ | h) $\sigma_{a=1 \text{ AND } b>2}(W)$ | i) $X \bowtie_{X.c < Y.c} Y$ |

$$W \bowtie X \bowtie Y \bowtie Z$$

$$T(W) * T(X) * T(Y) * T(Z)$$

$$\begin{aligned} & \frac{\max\{V(W, b), V(X, b)\} * \max\{V(X, c), V(Y, c)\} * \max\{V(Y, d), V(Z, d)\}}{400 * 300 * 200 * 100} \\ &= \frac{60 * 100 * 20}{400 * 300 * 200 * 100} \\ &= 20000 \end{aligned}$$

Homework3

$$\sigma_{a=10}(W)$$

$$T(W)/V(W,a) = 400/50 = 8$$

$$\sigma_{c=20}(Y)$$

$$T(Y)/V(Y,c) = 200/50 = 4$$

$$\sigma_{c=20}(Y) \propto Z$$

根据前一问的结果，可以知道前半部分的结果为4，而 $V(Z, d)=10>4$ ，所以结果为

$$\frac{T(\sigma_{c=20}(Y)) * T(Z)}{V(Z, d)} = \frac{4 * 100}{10} = 40$$

Homework3

$$W * Y$$

$$T(W) * T(Y) = 400 * 200 = 80000$$

$$\sigma_{d>10}(Z)$$

$$T(Z)/3 = 100/3 = 33.3$$

$$\sigma_{a=1 \text{ and } b=2}(W)$$

$$T(W)/[V(W,a) * V(W,b)] = 400/(50 * 40) = 0.2$$

Homework3

$$\sigma_{a=1 \text{ and } b > 2}(W)$$

$$T(W)/[3 * V(W,a)] = 400 / (3 * 50) = 2.67$$

$$X^{\infty}_{X.c < Y.c} Y$$

$$T(X) * T(Y) / 3 = 300 * 200 / 3 = 20000$$

Homework4

如果 **R** 和 **S** 都是非聚集的，似乎嵌套循环连将需要大约 $T(R)T(S)/M$ 次磁盘 I/O 时间。

你怎样做才能明显好于这个代价？

假设 $S(R)=S(S)$, 每次迭代时读取 **R** 的元组塞满 **M-1** 块的 **chunk**, 此时迭代次数为 $T(R)*S(R)/(M-1)$, 那么总的磁盘 I/O 时间为 $T(R)+T(R)*T(S)*S(R)/(M-1)$ 。近似为: $T(R)*T(S)*S(R)/M$. 例如: 1 个 **block** 中能存放 10 个元组, 即 $S(R)=1/10*block$, 那么效率提高 10 倍。

Homework4

如果 **R** 和 **S** 中只有一个是非聚集的，你应该怎样执行嵌套循环连接？考虑两种情况：较大的关系是非聚集的和较小的事非聚集的

假定 **R** 为较小关系，**S** 为较大关系

(1) **S** 是非聚集的：

方案 1：

For each loop:

Read $M - 1$ blocks of **R**

Read all of **S** (using 1 block) + join

代价为： $B(R) + B(R) * T(S) / (M - 1)$

Homework4

方案 2 :

Read $M-1$ blocks $((M-1) \times 1/S(S)$ tuples) of S

Read all of R (using 1 block) + join

代价为: $T(S) + B(R) * T(S) * S(S) / (M-1)$

选择代价最小的方案

(2) R 是非聚集的:

方案 1 :

For each loop:

Read $M-1$ blocks $((M-1) \times 1/S \quad (R)$ tuples) of R

Read all of S (using 1 block) + join

代价为: $T(R) + T(R) * B(S) * S(R) / (M-1)$

Homework4

方案 2 :

For each loop:

Read M-1 blocks of S

Read all of R(using 1 block)+ join

代价为: $B(S) + T(R) * B(S) / (M-1)$

选择代价最小的方案

比较 2 种情况下的最优代价

Homework4

假设这节中所描述算法的第二趟不需要所有的 **M** 个缓冲区，因为子表数小于 **M**。我们怎样通过使用额外的缓冲区来节省磁盘 **I/O**？

原本我们需要将第一趟中得到的有序子表都写回磁盘，现在由于子表数小于 **M**，可以将部分子表不写回，直接存储在内存缓冲区中，从而减少第二趟中的读子表操作，对于这样每块我们节省了 **2 次 IO**。

Test 1

假设某磁盘块参数如下：容量 **36.7GB**, 传输速率 **45MB/S** , 旋转一圈时间 **4ms** , 平均寻道时间 **5ms** , 最小寻道时间 **0.65ms** , 一个磁道大小 **180KB** 。如果磁盘块大小 **4KB**, 请回答下面问题:

(1) 随机读取 **1000** 个磁盘块需要多少时间 (**ms**) ?

(2) 假定 (1) 中的 **1000** 个磁盘块在单个磁道上连续存储, 并且所有磁盘块存储在相邻的磁道上, 此时读取这 **1000** 个磁盘块需要多少时间?

随机读取**1000**个块

$$t_{random} = 1000 \left(t_{\text{平均寻道}} + \frac{t_{\text{旋转时间}}}{2} + t_{\text{传输时间}} \right)$$

Test 1

由磁盘传输速率45MB/S，即45KB/ms（若1MB=1024KB,则46.08KB/ms也可以。或者用旋转时间计算：180KB/4ms=45KB/ms也OK），因此

$$t_{\text{传输时间}} = \frac{4KB}{45KB/ms} = 0.09ms$$

或者

$$\frac{4KB}{46.08KB/ms} = 0.09ms$$

因此，

$$t_{\text{random}} = 1000(5+2+0.09)ms = 7090ms$$

Test 1

顺序读取 **1000** 个块

$$t_{\text{sequential}} = t_{\text{平均寻道}} + \frac{t_{\text{旋转时间}}}{2} + 1000 * t_{\text{传输时间}} + k * t_{\text{最小寻道}}$$

其中k是寻道次数。因为磁盘块大小为4KB，单个磁道180KB,因此1000个磁盘块需要分配 $\left\lceil \frac{4*1000}{180} \right\rceil = 23$ 个磁道，因此除了第一次寻道外还需执行22次相邻磁道的寻找操作，即k=22，因此：

$$t_{\text{sequential}} = 5 + 2 + 1000 * 0.09 + 22 * 0.65 \text{ms} = 111.3 \text{ms}$$

Test

B+-Tree 设定如下:

N: 记录数

n: **B+-Tree** 的阶, 即节点所能容纳的键数

R: 读取一个磁盘块的旋转延迟

S: 读取一个磁盘块的寻道时间

T: 读取一个磁盘块的传输时间

m: 在内存的 **m** 条记录查找一条记录的时间

假设所有磁盘块都不在内存中

现在考虑压缩 **B+-Tree**, 假设每个节点的键值压

缩 **1** 倍, 即同样空间可压缩存储 **2n** 个键值和

2n+1 个指针。额外代价是解压缩, 设每个压缩键

值的内存解压时间为 **c**, 请问在一棵满的 **n** 阶压

缩 **B+-Tree** 中查找给定记录地址的时间多少?

(**n+1** 可近似表示为 **n**)

Test

1. 传统的B+-Tree树的高度为 $\log_n N$
2. 读块的时间 $= R + S + T$
3. 每块有 n 条记录，内存查找时间为 n
4. 所以传统 B+-Tree查找时间= $\log_n N * (R + S + T + n)$
5. 考虑压缩B+-Tree，树的高度为 $\log_{2n} N$
6. 增加了解压时间 $2cn$ ，内存查找时间变为 $2n$
7. 因此压缩B+-Tree的查找时间为
 $\log_{2n} N * (R + S + T + 2cn + 2n)$

Final Exam

考试形式不同于往年，往年为开卷考试，今年为闭卷考试，总共 10 个判断题，及 4 个大题。

10 个判断， 30 分，个人感觉不容易，考点比较细，都是一些概念的判断，考的基本都是前三章的内容，请大家仔细复习。（eg:ER图是自下向顶设计的；关系模型中候选码一定要存在；一个只有 2 个属性的关系模式一定满足第三范式吗）

第一个大题考的是 PPT 第七章的内容，查询优化，老师上课讲的 PPT 上不全，只要把书上 P153 页 5.4.3 内容掌握即可完美答题。题目难度：一颗星

Final Exam

第二个大题考的是 **PPT** 第十章的内容，并发调度，考察冲突可串性及优先图的画法，掌握 **PPT** 即可。题目难度：一颗星

第三个大题考的是模糊查询结合 **B+** 树索引查询（问题是否能用 **B+** 树索引查询应用于模糊查询能否提高查询效率），金老师上课基本没提到模糊查询，因此需要自己结合 **SQL** 中模糊查询的原理和 **B+** 树索引查询原理解答。题目难度：四颗星

第四个大题考的是 **PPT** 第八章的内容，查询执行，主要考察优化的归并排序算法。题目难度：五颗星（题目见附录）

附第五题

我们想将关系 R 按某个字段排序。已知 R 的下列信息:

- R 包含 100000 个元组, 即 $T(R) = 100000$.
- 一个磁盘块大小为 4000 bytes.
- R 的元组大小为 400 bytes, 即 $S(R) = 400$.
- 关系 R 在磁盘上是非连续 (contiguous) 存放的 •
- 排序字段的大小为 32 bytes.
- 记录指针的大小为 8 bytes.

(普通硬盘读写一个块的时间都是 $30t$, 固态硬盘读取一个块的时间是 t , 写出一个块的时间为 $50t$ 。)

考虑下面改进的归并排序算法。原来的两阶段归并排序的第一阶段是将排序后的整个元组写到 chunk 中, 现在我们仅将排序后的 $\langle \text{sorting key}, \text{recordPointer} \rangle$ 写出。第一阶段, 我们在内存中将记录按 $\langle \text{sorting key}, \text{recordPointer} \rangle$ 排序, 当 $\langle \text{sorting key}, \text{recordPointer} \rangle$ 记录填满内存时将其写到 chunk 中。第二阶段, 读入各个 chunk 中的 $\langle \text{sorting key}, \text{recordPointer} \rangle$ 并在内存中归并。通过记录指针 (recordPointer) 我们可以读取记录的其它部分 (从 R 的存储块中), 并将排好序的记录写回磁盘。

附第五题

回答下面的问题：

（1）普通硬盘使用两阶段优化归并排序需要多少次磁盘 I/O？（包括最后将排序文件写回磁盘的代价，用 t 表示）这个改进算法的 I/O 代价优于原来的归并排序算法吗？为什么？

（2）SSD 使用两阶段优化归并排序需要多少次磁盘 I/O？（包括最后将排序文件写回磁盘的代价，用 t 表示）这个改进算法的 I/O 代价优于原来的归并排序算法吗？为什么？

放映结束
感谢各位的批评指导！
谢 谢！

让我们共同进步

