作业 1

1.

1）$\prod_{SC1.S\#,SC1.Grade,SC2.Grade}(\sigma_{SC1.C\#=001\wedge SC2.C\#=002\wedge SC1.S\#=SC2.S\#}(\rho_{SC1}(SC)\times\rho_{SC2}(SC)))$

2）$\prod_{S\#,SNAME,AGE}(\sigma_{C\#=001}(S\bowtie SC))$

3）$\prod_{SNAME,AGE}(S)-\prod_{SNAME,AGE}(\sigma_{C\#=002}(S\bowtie SC))$

4）$\prod_{SNAME}(\sigma_{TEACHER='G'\wedge GRADE\geq 90}(C\bowtie SC\bowtie S))$

5）$\prod_{SNAME}(\prod_{S\#,C\#}(SC)\div\prod_{C_{no}}(C)\bowtie S)$

2.

1) $\prod_{J\#}(\sigma_{SCITY='北京'\wedge SNAME='S1'\wedge COLOR='蓝色'}(S\bowtie SPJ\bowtie P)$

2) $\prod_{J\#,JNAME}(\sigma_{SCITY=JCITY}(S\bowtie SPJ\bowtie J))$

3) $\prod_{P\#}(P)-\prod_{P\#}(\sigma_{JCITY='长春'}(J\bowtie SPJ))$

4) $\prod_{J\#,JNAME}(\sigma_{P\#='P2'}(SPJ\bowtie J))$

5) $\prod_{S\#,SNAME}(\sigma_{J\#='J5'\wedge COLOR='绿色'}(SPJ\bowtie S\bowtie P))$

3.判断$\prod_F(S)-\prod_K(R)$是否为空，若不为空则说明违反。

作业 2

1.

1）SELECT COUNT(*) FROM Employee WHERE D#=1;

2）SELECT D#,COUNT(*) FROM Employee GROUP BY D#;

3）SELECT NAME FROM Employee WHERE D# IN(SELECT D# FROM Department WHERE DNAME=’技术部’) AND SALARY>10000;

4）SELECT D#,AVG(SALARY) AS avgSalary FROM Employee GROUP BY D#;

5）SELECT COUNT(*) FROM Employee WHERE D# IN(SELECT D# FROM Department WHERE DNAME=’技术部’) AND Employee.NAME LIKE '张%';

2.

1）SELECT Sno# FROM Borrow GROUP BY Sno# HAVING COUNT(*)>5;

2）SELECT Sname,Sage FROM Student WHERE Sno# in(
SELECT Sno# FROM Borrow WHERE B# in(
SELECT B# FROM Book WHERE Publisher=’人民教育出版社’)) ORDER BY Sage DESC;

3）SELECT Sno# FROM Borrow GROUP BY Sno# HAVING MIN(Time) > 90;

4）SELECT Title,COUNT(*) FROM Book WHERE Title LIKE ‘%Big/%Date%;

5）SELECT Title# FROM Book NATURAL JOIN Borrow NATURAL JOIN Student WHERE Sdept=’CS’ GROUP BY Title# HAVING COUNT(DISTINCT Sno#) > 5;

3.

1）SELECT * FROM S WHERE D#=’物理系’ ORDER BY S#;

2）SELECT S#, SNAME FROM S WHERE SNAME LIKE ’王%’;

3）SELECT VIEW SumC AS

SELECT S# SNAME, count(*) AS Count FROM S, SC

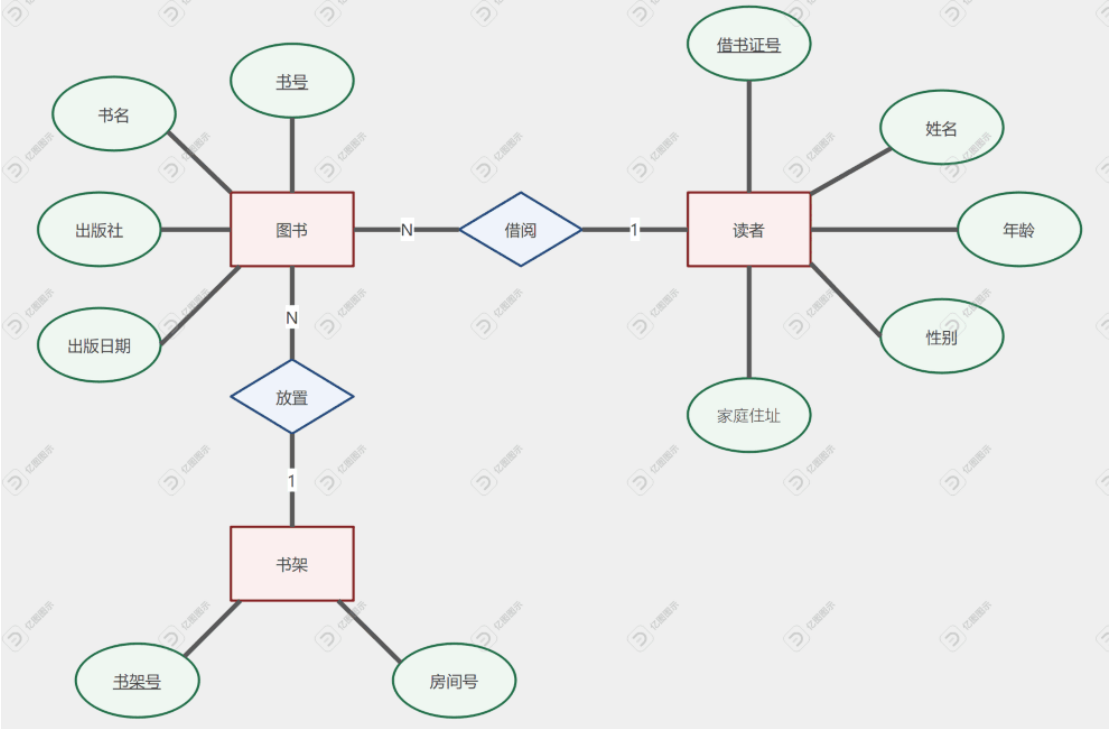
WHERE S.S#=SC.S# GROUP BY S#;

4 ） SELECT SNAME FROM S NATURE JOIN SC WHERE SC.C#=’1002’ INTERSECT SELECT SNAME FROM S NATURAL JOIN SC WHERE SC.C#=’1003’;

作业 3

1.

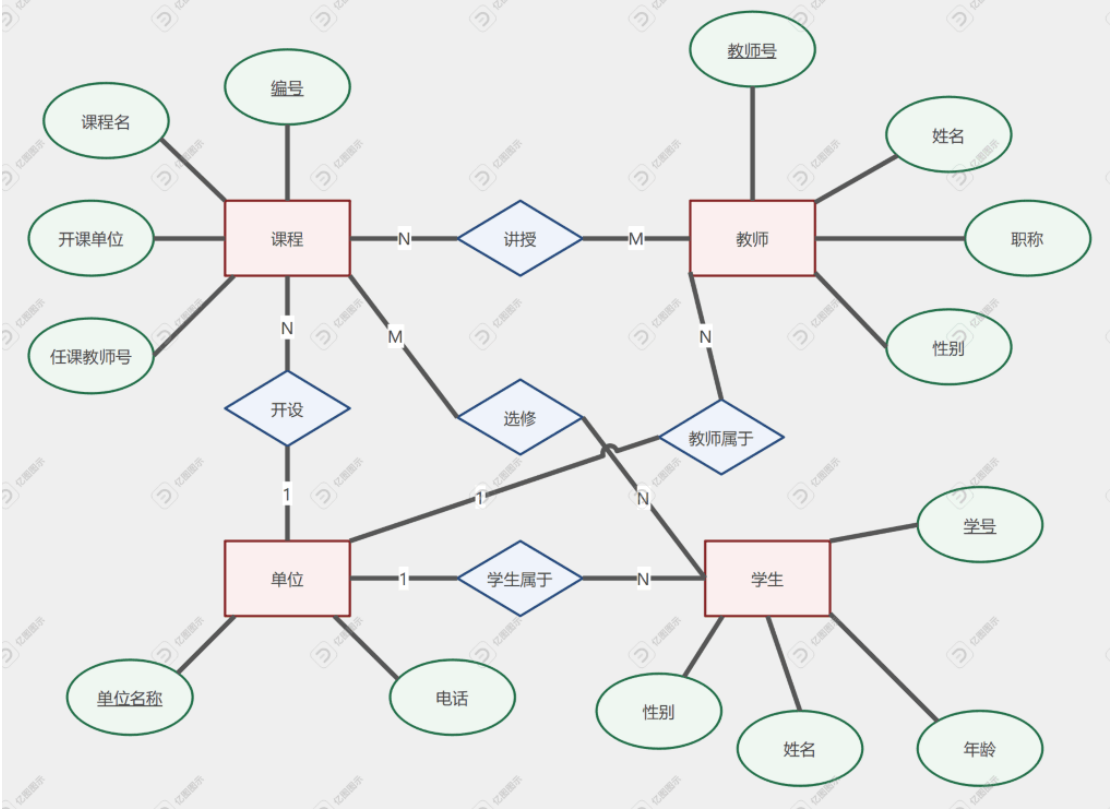
1)



2) 图书（书号、书名、出版日期、出版社）
读者（借书证号、姓名、年龄、性别、家庭住址）
书架（书架号、房间号）
借阅（书号、借书证号）
放置（书架号、书号）

2.

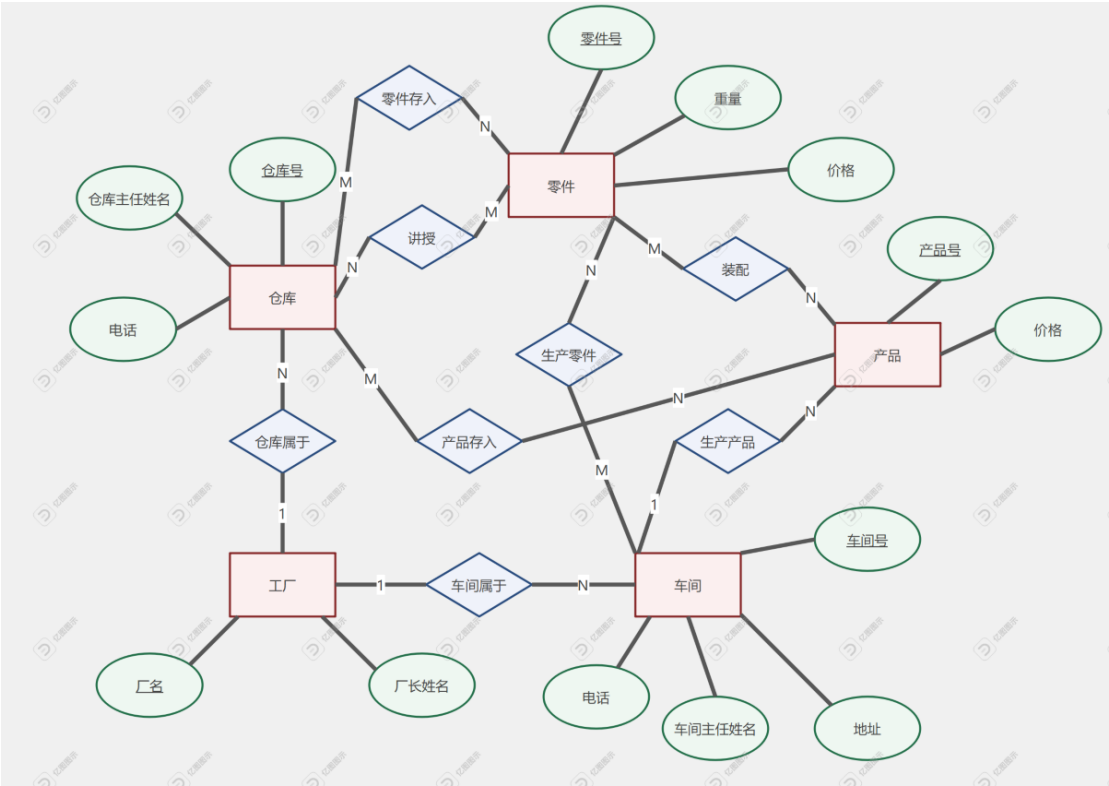
1)



2) 单位（单位名称、电话）
学生（学号、单位名称、姓名、性别、年龄）
教师（教师号、姓名、性别、职称、单位名称）
课程（编号、课程名、开课单位）
选修（编号、学号）
讲授（编号、教师号）

3.

1)



2) 关系模式：

工厂（<u>厂名</u>、厂长姓名）

车间（<u>车间号</u>、车间主任姓名、地址、电话、厂名）

仓库（<u>仓库号</u>、仓库主任姓名、电话、厂名）

零件（<u>零件号</u>、重量、价格）

产品（<u>产品号</u>、价格、车间）

生产零件（<u>车间号</u>、<u>零件号</u>）

装配（<u>零件号</u>、<u>产品号</u>）

零件存入（<u>零件号</u>、<u>仓库号</u>）

产品存入（<u>产品号</u>、<u>仓库号</u>）

作业 4

1.

1) $(AB)_F^+ = \{A, B, C, D, E\}$

2) F_c={AB→C, BC→A, BC→D, D→E, CF→B}

3) {ABF, CF}

4) $\rho =$ {ABC, DE, BCD, BCF}

2.

1) {B→C, B→E, C→B, AB→D, E→F}

2) {AB,AC}

3.

1) F={A→BC, BC→DE, D→F, E→G}　候选码是（A）

2) 不符合，因为不满足每个非主属性都完全依赖于候选键。

4.

1）依赖集 F={CNo→CName, (CNo, PNo) →(STime, ETime), PNo→(PAddr, Rent, ONo), ONo→(OName,OPhone), OPhone→ONo}

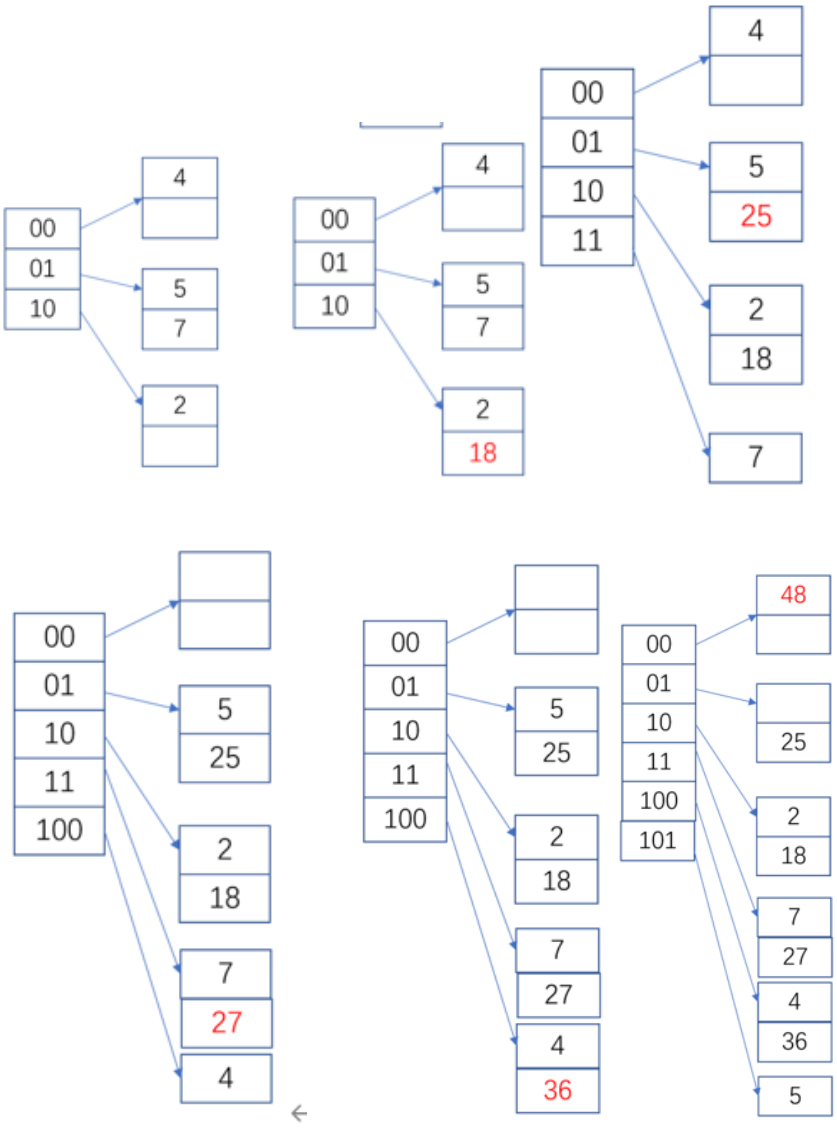
极小函数依赖集F_m={CNo→CName, (CNo, PNo) →STime, (CNo, PNo) →ETime, PNo→PAddr, PNo→Rent, PNo→ONo, ONo→OName, ONo→OPhone, OPhone→ONo}

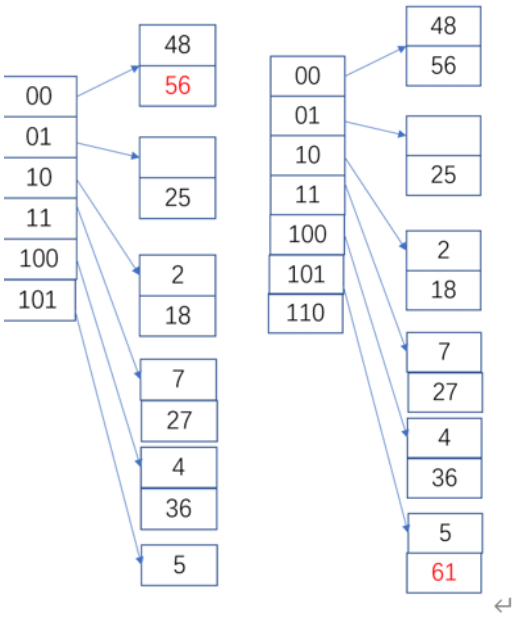
2）关系 Rental 的候选键为（CNo，PNo），满足 1NF，由于 CNo→CName，即存在非主属性依赖于候选键，所以不符合 2NF，因此 Rental 所能达到的最高范式等级为 1NF。

3）$\{R_1(CNo, CName), R_2(CNo, PNo, STime, ETime), R_3(PNo, PAdrr, Rent, ONo), R_4(ONo, OName, OPhone)\}$

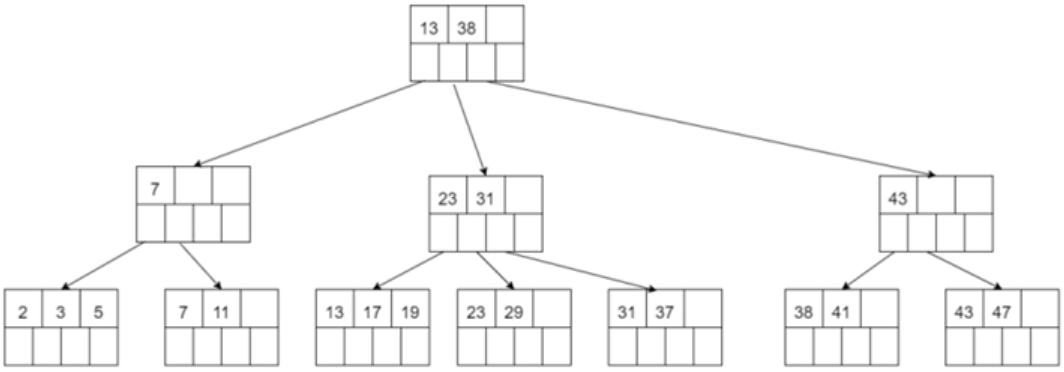
作业 5

1.

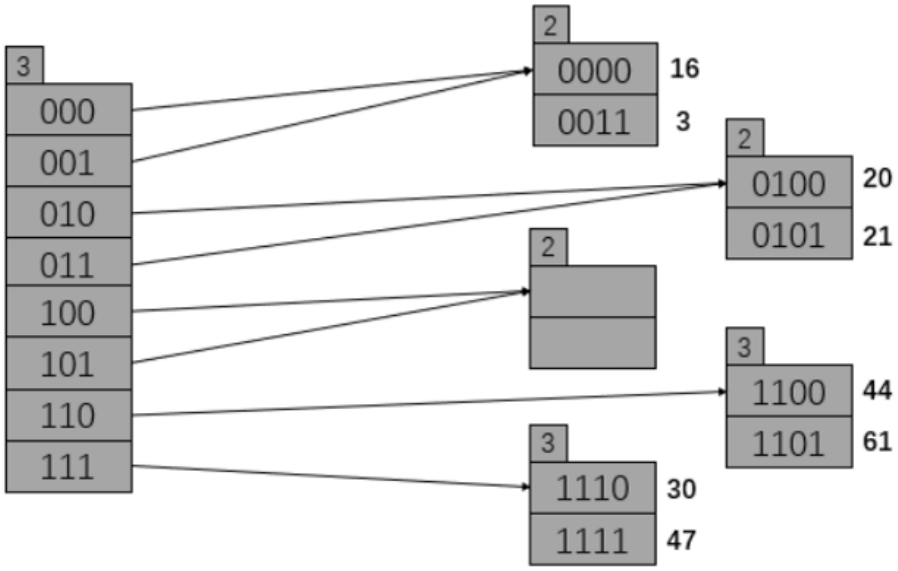



00
01
10
11
100
101
48
56
25
2
18
7
27
4
36
5
00
01
10
11
100
101
110
48
56
25
2
18
7
27
4
36
5
61

2.


13 38
7
23 31
43
2 3 5
7 11
13 17 19
23 29
31 37
38 41
43 47

3.


3
000
001
010
011
100
101
110
111
2
0000
16
0011
3
2
0100
20
0101
21
2
3
1100
44
1101
61
3
1110
30
1111
47

作业 6

1.

1)R 中有 20000/20=1000 块，S 中有 60000/30=2000 块

嵌套循环连接算法，要把外关系 R 中的每块读入缓冲池，再对内关系 S 的每个（M-1）块遍历其中的元组进行链接。因此需要 B(R)+B(R)*B(S)/(M-1)=1000+(1000*2000)/40=51000 次

2）R 和 S 的连接属性为 B，由于 S 中 B 作为主键已递增排序，则无需为 S 创建归并段。在对 R 创建归并段时需要 B(R)次 I/0，把归并段写入文件又需要 B(R)次 I/0，最后在归并阶段使用一趟连接算法需要 B(R)+B(S)次 I/0。综上，采用归并连接需要 B(S)+3*B(R)=2000+3*1000=5000 次

3）因为 R.B 是参照 S.B 的外键，所以每一个 R 中的元组能且仅能与 S 中的一个元组发生自然连接，则最后结果元组的个数就是 R 元组的个数 20000 个。R 中元素为(A，B)，S 中元素为(B，C)。由题目可得空间大小得关系(A+B+C):(A+B)=1.2:1，(A+B+C):(B+C)=2:1 解得 A=1.5B，C=0.5B 由于一块可以容纳 20 个 R 元组或者 30 个 S 元组所以可以容纳 max(20/1.2，30/2)（向下取整）=16 个 R⋈S 元组，所以需要占用 20000/16=1250 个块

2.

1）建立聚簇所欲，首先需要扫描关系 R，该关系共有 1000 个元组，每块可容纳 20 个元组，因此 R 占用 50 个数据块，扫描 R 的 I/O 代价为 50 次。接下来，对于 R 中的每一个元组，通过聚簇索引在 S 中查找匹配的元组。因为 S.Y 有 20 个不同的值，平均每个 Y 值对应 75 个 S 的元组。由于使用的是聚簇索引，这些匹配的 75 个元组在 S 中是连续存储的，分布在 3 个数据块内（每块 30 个元组）。假设 B+树索引的高度为 2，则每次索引查找需要 2 次 I/O 来定位叶子节点，再顺序读取 3 个连续的数据块。因此，每个 R 的元组在索引连接中的总 I/O 代价为 5 次（2 次索引查找 + 3 次顺序读取）。对于 1000 个 R 的元组，总的索引查找 I/O 代价为 5000 次。最终，整个索引连接的总 I/O 代价为扫描 R 的 50 次加上索引查找的 5000 次，共计 5050 次 I/O

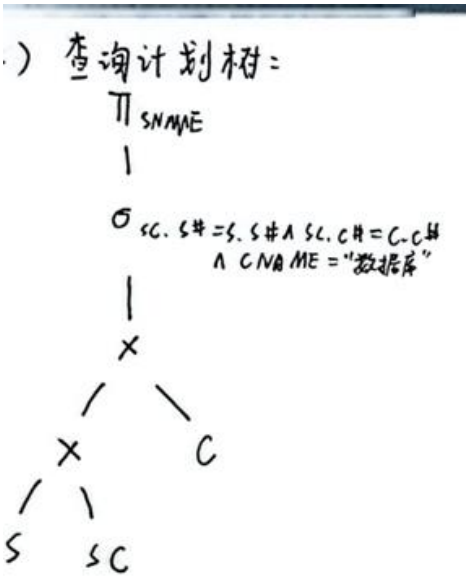
2)若建立非聚簇索引，同样需要首先扫描关系 R，占用 50 个数据块，I/O 代价为 50 次。然后，对于 R 中的每一个元组，通过非聚簇索引在 S 中查找匹配的元组。尽管 S.Y 仍有 20 个不同的值，平均每个 Y 值对应 75 个 S 的元组，但由于非聚簇索引的特性，这些匹配的 75 个元组在 S 中是随机分布的，可能分布在 75 个不同的数据块中。每次索引查找仍需 2 次 I/O 来定位索引叶子节点，但由于数据块的分散性，读取每个匹配的元组需要单独的随机 I/O。因此，每个 R 的元组在索引连接中的总 I/O 代价为 77 次（2 次索引查找 + 75 次随机读取）。对于 1000 个 R 的元组，总的索引查找 I/O 代价为 77,000 次。最终，整个索引连接的总 I/O 代

价为扫描 R 的 50 次加上索引查找的 77,000 次，共计 77,050 次 I/O。

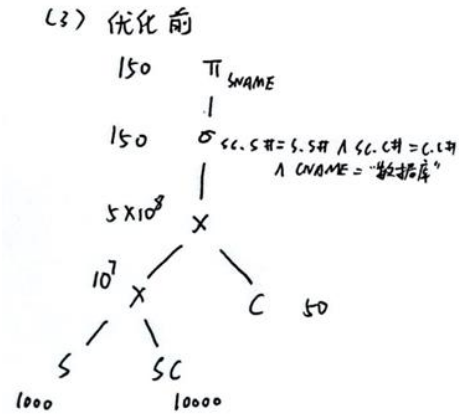
3.

1) $\prod_{SNAME}(\sigma_{SC.S\#=S.S\#\wedge SC.C\#=C.C\#\wedge CNAME='数据库'}(S\times SC\times C)$

2）



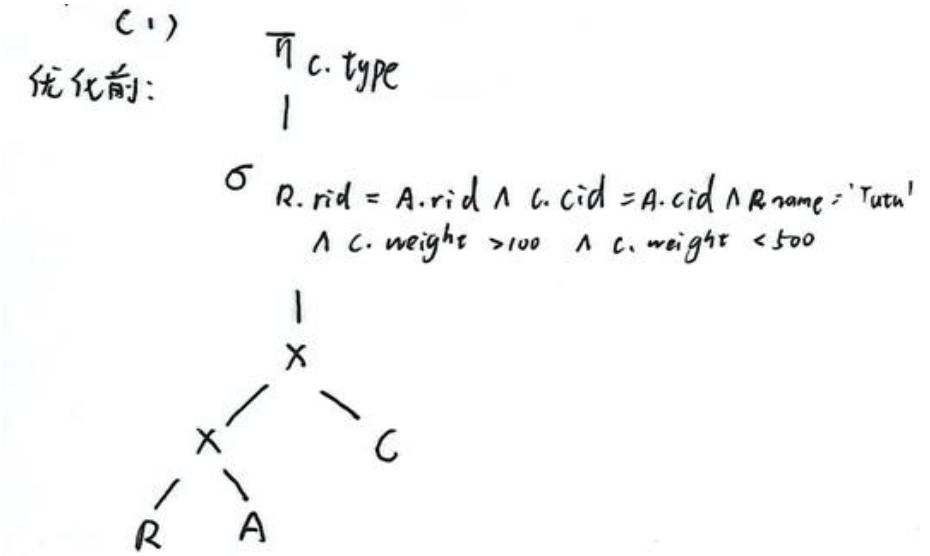
3）



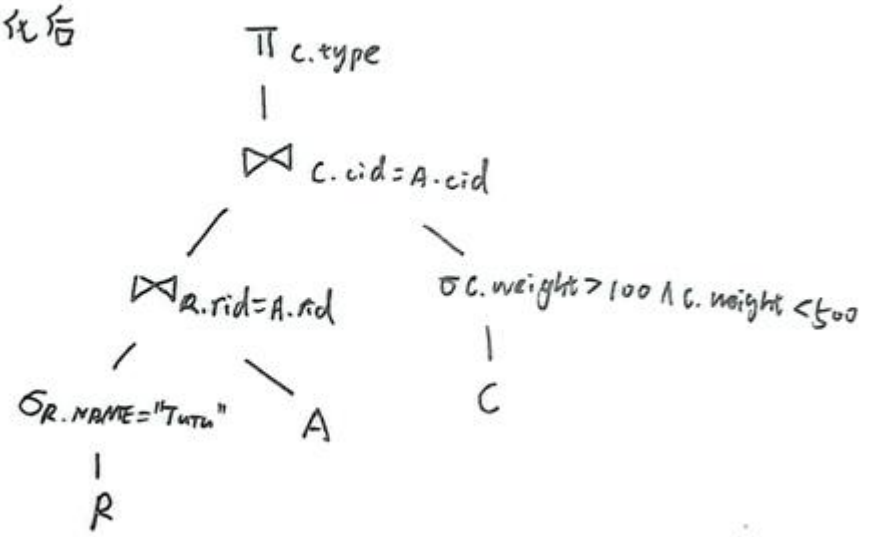
4.

(1)

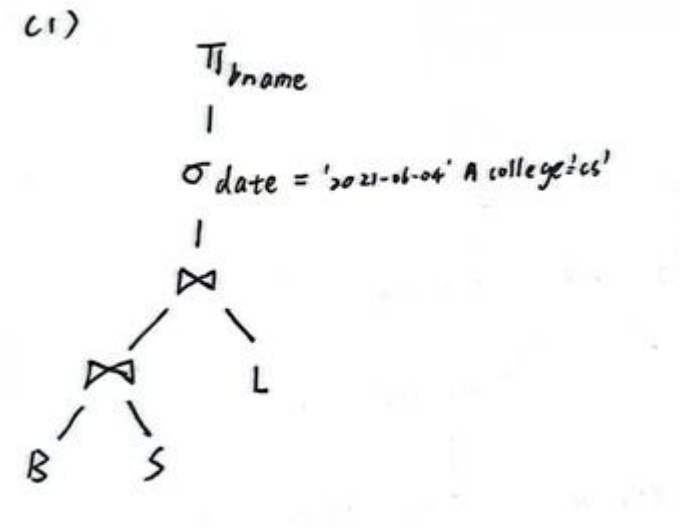
优化前：


π c.type
σ R.rid = A.rid ∧ C.cid = A.cid ∧ R.name = 'Tutu'
∧ C.weight > 100 ∧ C.weight < 500
X
X
C
R
A

(2) 优化后


π c.type
⋈ C.cid = A.cid
⋈ R.rid = A.rid
σ C.weight > 100 ∧ C.weight < 500
σ R.NAME = "Tutu"
A
C
R

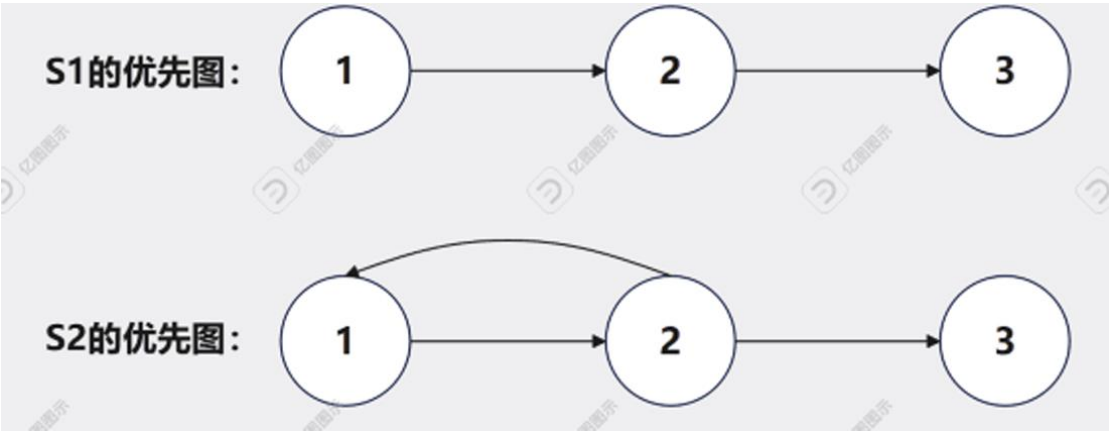
5.



理由：先进行选择下推，尽量过滤大量无关元组，从而减少 I/O 次数；再进行投影下推，减少空间开销，从而减少每次的 I/O 开销

作业 7

1.



根据优先图，S2 有环，S1 无环，故 S2 不可串行化，S1 可串行化

2.

1）两阶段锁协议中，每个事务的执行分为两个阶段：增长阶段，事务向锁管理器请求需要的锁；萎缩阶段，事务释放它获得的锁，但不能再请求加锁。

添加锁和解锁后遵从 2PL 协议的事务 T1 和 T2：

```
T1:                          T2:
S-LOCK(A);                   S-LOCK(B);
X-LOCK(B);                   X-LOCK(A);
read(A);                     read(B);
read(B);                     read(A);
if A>B then B:=A;            if B<0 then A:=B*B;
write(B);                    write(A);
UNLOCK(A);                   UNLOCK(B);
UNLOCK(B);                   UNLOCK(A);
```

2）

```
T1:                T2:
S-LOCK(A);
X-LOCK(B);
read(A);
read(B);
if A>B then B:=A;
write(B);
UNLOCK(A);
UNLOCK(B);
                    S-LOCK(B);
                    X-LOCK(A);
                    read(B);
                    read(A);
                    if B<0 then A:=B*B;
                    write(A);
                    UNLOCK(B);
                    UNLOCK(A);
```

3）当过程 1 等待过程 2 释放锁，而过程 2 也在等待过程 1 释放锁的时候，就会发生死锁

死锁调度：

```
- 如果T1 和 T2 都在等待彼此释放锁，则发生死锁
- 给出一个会发生死锁的调度：
 T1:                        T2:
                            S-LOCK(B);
 S-LOCK(A);
 X-LOCK(B);<拒绝！>
                            X-LOCK(A);<拒绝！>
 ————————————————    产生死锁    ————————————————
 read(A);
 read(B);
```

```
if A>B then B:=A;
write(B);
UNLOCK(A);
UNLOCK(B);
                            read(B);
                            read(A);
                            if B<0 then A:=B*B;
                            write(A);
                            UNLOCK(A);
                            UNLOCK(B);
```

4）死锁的检测有两种方法：

方法一：超时检测，如果在给定的时间内没有任何事务执行，则认为死锁发生

方法二：等待图检测，DBMS 用等待图表示事务关于锁的等待关系，DBMS 定期检查等待图中是否存在环，如果等待图中有环，则死锁发生。

3.

1）对应的冲区处理策略是：STEAL + NO-FORCE

STEAL 的内容是：允许将未提交事务所做的修改写到磁盘并覆盖现有数据

NO-FORCE 的内容是：不强制事务在提交前必须将其所做的修改全部写回磁盘

2）T2 和 T3 在故障发生时还没有 commit，但已有部分内容写入磁盘，需要 undo

T1 在故障发生时已经 commit，但可能还有内容未写入磁盘，需要 redo

3）故障恢复完成时，A=114514, B=”hitcs”

故障恢复具体过程：

- redo 阶段从前向后扫描日志

  redo:<T1,A,114,114514>将磁盘上 A 的值覆写为 114514

  redo:<T1,B,'hit','hitcs'>将磁盘上 B 的值覆写为'hitcs'

- undo 阶段从后向前扫描日志

  undo:<T2,A,114514,1919810> A 的值恢复为 114514

  undo:<T3,B,'hitcs','hicsdb'> B 的值恢复为'hitcs'

4.

1）故障发生时，T1 在检查点前提交，不需要操作；

T2 和 T4 在检查点后提交，需要 redo；

T3 和 T5 未提交，需要 undo

2） redo:

<T6, X, 100, 1>

undo:

<T7, X, 1, 3>

<T7, Y, 50, 6>

<T8, Y, 6, 8>

<T8, Z, 10, 9>

<T8, Z, 9, 10>

3）考虑日志最后一个 <checkpoint,L>，仅考虑 L 中的事务以及 checkpoint 后 begin 的事务：即 T6-T8

1. T7-T8 并没有对应的 commit 记录或 abort 记录，需要进行 undo
2. T6 有对应的 commit 记录，需要进行 redo