一、 复旦大学需要设计一个用电管理系统，相关信息如下：

a) 学校所有职工(Employee)具有唯一的工号(Eno)，姓名(Ename)，性别(Esex)等属性；

b) 学校分为多个校区(Campus)，

c) 每个校区具有唯一编号(Cno)，还包括名称(Cname)，另外还包含地址(Caddress)，面积(Carea)属性；

d) 每个校区包含多个建筑(Building)，每个建筑具有唯一编号(Bno)，还包括名称(Bname)，高度(Bheight)(单位 m)，建筑类别(Btype)属性；

e) 建筑装有多个设备(Device)，每个设备具有唯一编号(Dno)，设备名称(Dname)，设备类型(Dtype)，功率(Dpower)等属性。

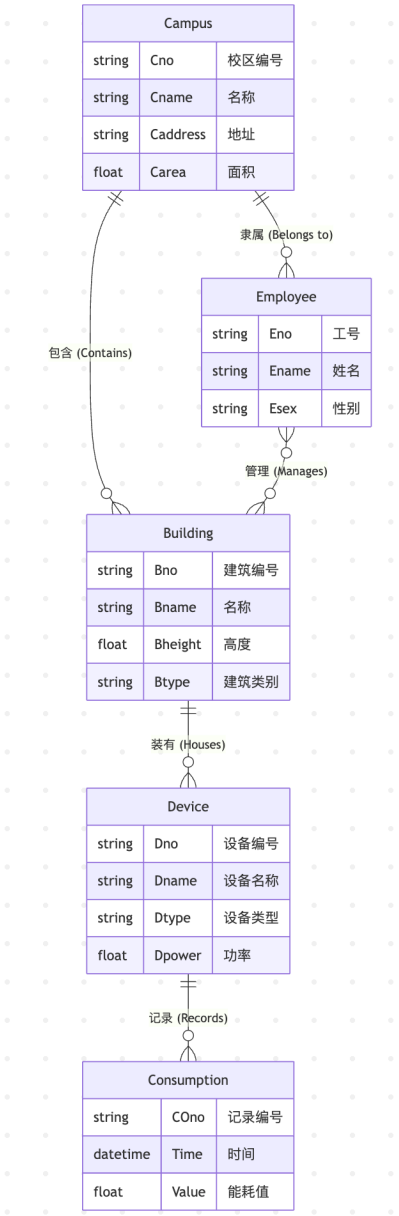
f) 设备的能耗信息记录在能耗统计(Consumption)中，每条记录具有唯一编号(COno)，时间(Time)，能耗值(Value)等属性。

该系统的需求如下：

每个校区雇佣多个职工，一个职工只能隶属于一个校区；职工负责管理学校的建筑，每个职工可管理多栋建筑，职工管理的建筑可能有交叉；同时，需要记录每个职工管理每栋建筑的开始时间；每个校区包含多栋建筑，每栋建筑只隶属一个校区；一栋建筑装有多个设备，每个设备以 1 次/小时的频率，记录其用电信息，包括用电数值(该时间内用电量的值)，采集时间，记录在 Consumption 中。

请完成以下问题：

1) 画出该数据库应用系统的 ER 图，标出实体之间的“一对多”，“一对一”或者“多对多”联系；



“一对多”，“一对一”或者“多对多”等约束考试时请使用教材或 PPT 中的形式。

2) 根据画出的 ER 图，设计该系统的关系模式，标出主码，并做到尽量避免冗余(如考虑可能的模式合并等)；

Campus (**Cno**, Cname, Caddress, Carea)

Employee (**Eno**, Ename, Esex, Cno)

Building (**Bno**, Bname, Bheight, Btype, Cno)

Manages (**Eno, Bno**, StartTime)

Device (**Dno**, Dname, Dtype, Dpower, Bno)

Consumption (**COno**, Time, Value, Dno)

3) 根据 2)中设计的数据库系统的关系模式，写出以下满足条件的查询：

a) 建立一个视图，供快速查询校区的建筑信息；

```
CREATE VIEW Campus_Building_Info AS
SELECT
    c.Cno AS Campus_ID,              -- 校区编号
    c.Cname AS Campus_Name,          -- 校区名称
    c.Caddress AS Campus_Address,    -- 校区地址
    b.Bno AS Building_ID,            -- 建筑编号
    b.Bname AS Building_Name,        -- 建筑名称
    b.Bheight AS Building_Height,    -- 建筑高度
    b.Btype AS Building_Type         -- 建筑类别
FROM
    Campus c
JOIN
    Building b ON c.Cno = b.Cno;
```

b) 用 SQL 语句查询在建筑名为“GuangHua”的建筑中的设备，根据设备类别分类后，输出不同类别设备中，平均耗电量最高的设备名称及其平均耗电量；

```
SELECT
    D.Dtype,
    D.Dname,
    AVG(C.Value) AS AvgConsumption
FROM Device D
JOIN Building B ON D.Bno = B.Bno
JOIN Consumption C ON D.Dno = C.Dno
WHERE B.Bname = 'GuangHua'
GROUP BY D.Dtype, D.Dname
HAVING AVG(C.Value) = (
    SELECT MAX(AvgValue)
    FROM (
```

```
            SELECT Dtype, AVG(Value) AS AvgValue
            FROM Device
            JOIN Building ON Device.Bno = Building.Bno
            JOIN Consumption ON Device.Dno = Consumption.Dno
            WHERE Building.Bname = 'GuangHua'
            GROUP BY Dtype, Dname
        ) AS SubQuery
WHERE SubQuery.Dtype = D.Dtype
);
```

c) 用关系代数表示 Cno 为“C02”的校区在“2018-06-01”这一天，设备类型为“Air conditioner”的设备消耗的用电量总和，并画出该表达式的表达式树。

γ SUM(Value) (

σ Time='2018-06-01' ∧ Dtype='Air conditioner' ∧ Cno='C02'

(Consumption ⋈ Device ⋈ Building ⋈ Campus)

)

二、 考虑关系模式 r(A， B， C， D， E)，其函数依赖集 F = {B→ CD， A→ D， D→ C， CE →D， DE → B}。请完成以下问题：

1) 给出关系模式 r 中具有属性最少的候选键；

AE 是 r 的一个候选码。AE 属性集闭包(AE)+ = ABCDE

2) 给出上述函数依赖集 F 的正则覆盖，请写出推导步骤；

F 的正则覆盖 Fc = { B → D, D → C, A → D, CE → D, DE → B }

因为 D → C，所以 C 在 B → CD 右部是多余的，删除。没有其它冗余属性，所以 F 的正则覆盖是{ B → D, D → C, A → D, CE → D, DE → B }

3) 给出关系模式 r 的一个 BCNF 分解，请写出分解过程；

B → CD 是 r 上成立的非平凡函数依赖，且 B 不是 r 的超码(B+=BDC)，所以将 r(BADCE)分解 R1(BDC)和 R2(BAE)，此时 R2 满足 BCNF；

D → C 是 R1 上成立的非平凡函数依赖，且 D 不是 R1 的超码，所以将 R1 分解成 R11(DC)和 R12(BD)，此时 R11 和 R12 都满足 BCNF；

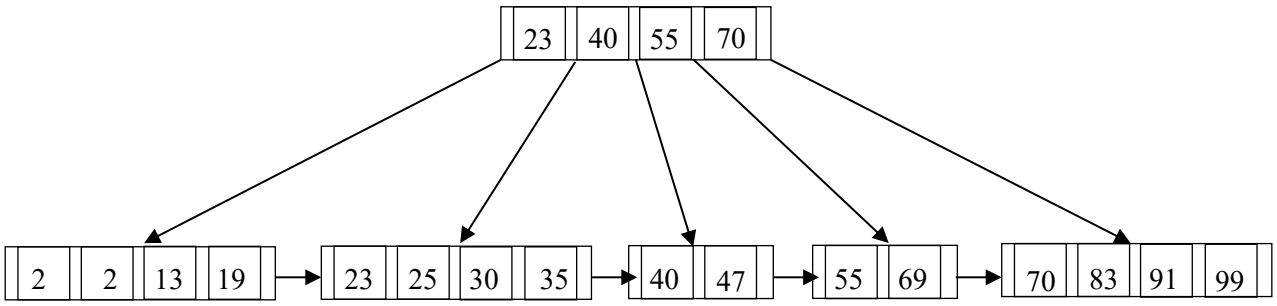
所以 r 的 BCNF 分解{BD, DC, BAE}

注：该题答案不唯一。例如{BD, AC, BEP}也是 r 的一个 BCNF 分解。

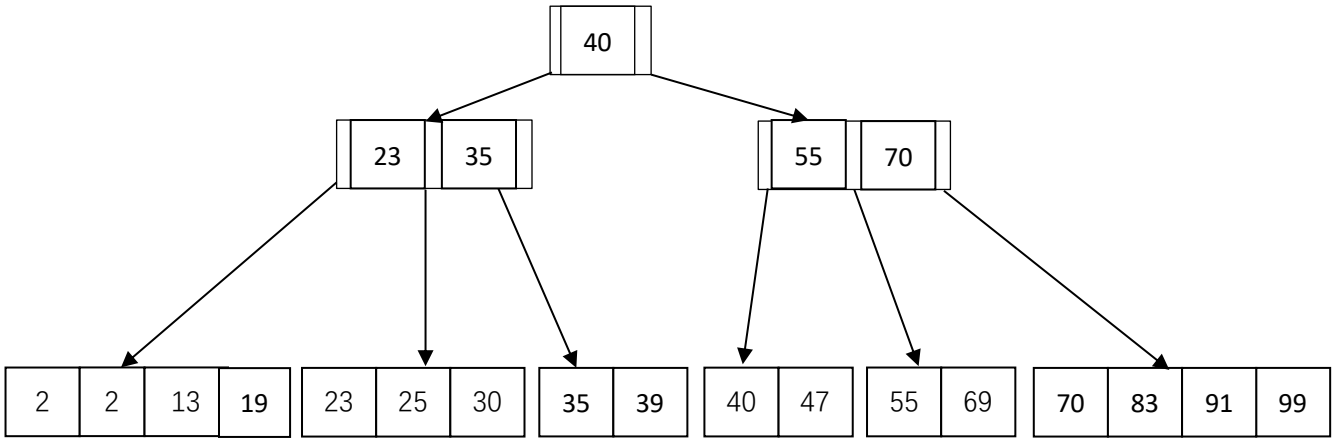
4) 讨论上题分解是否保证无损连接和依赖保持。

无损连接，不依赖保持

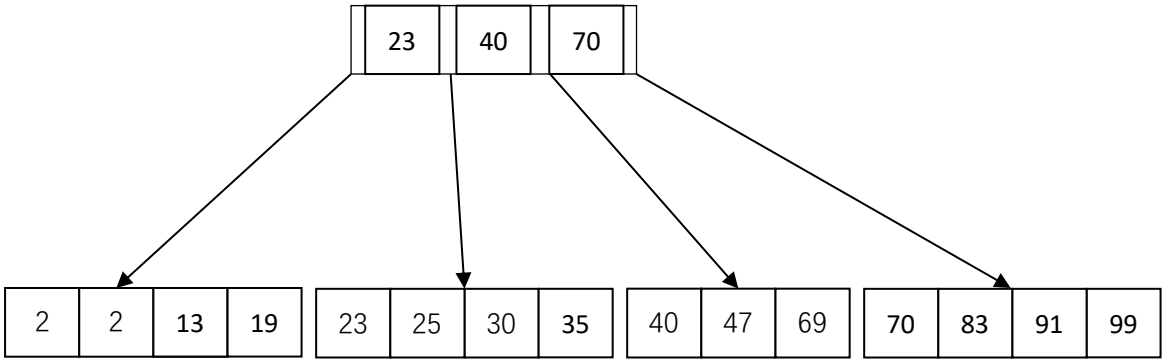
三、 **考虑用下图中的 B+树来构建索引，每个节点最多能放 5 个指针（n=5），请完成以下问题：**



1) 向图中给出的 B^+树中插入 39，画出完成该操作后的 B^+树；



2) 向图中给出的 B^+树中删除 55，画出完成该操作后的 B^+树（忽略 1）的操作）；



3) 从图中的 B^+树中查找搜索码值为 2 的记录，并简述步骤（注意存在不唯一搜索码）；

首先搜索 35 应该在哪个叶节点中，从那个叶节点开始顺序查找，直到第一个大于 34 的值，接下来根据叶节点相连的指针，顺序查找，直到搜索码值大于 35。

四、 假设在一个文件上使用可扩充散列，散列函数为 h(x)= x mod 8， 且每个桶可以容纳 2 条记录。该文件所含记录的搜索码值如下：

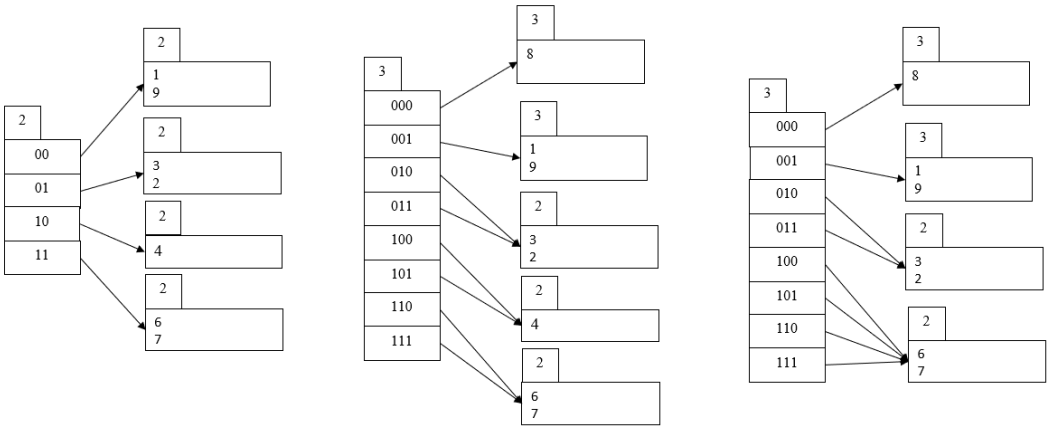
6，9，3，7，4，2，1

1) 根据搜索码和散列函数，求出初始散列值，以及初始的可扩充散列结构(散列值左起为第一位，从高位到低位)；

6	9	3	7	4	2	1
110	001	011	111	100	010	001

2) 给出向 1)中生成的结构插入记录 8 后的可扩充散列结构；

3) 给出向 2)中生成的结构删除记录 4 后的可扩充散列结构(注意桶合并，不必关心桶地址表大小)。



五、 **在某图书借阅管理系统中有下列 3 个关系：用户 USER、图书 BOOK、图书借阅 BORROW，其中 USER 的主码为属性 uid， BOOK 的主码为属性 bid，BORROW 包含参照前两个关系的外键 uid 和 bid。目前已知如下信息，USER 中有 6400 个记录，BOOK 中有 3300 个记录，BORROW 中有 75000 个记录。每个数据块可以存储 20 条 USER 记录、15 条 BOOK 记录、30 条 BORROW 记录。**

请完成以下问题，并给出计算过程：

1) 假设有一个基于 BOOK 表的查询，查询结果需要按照 BOOK.bid 排序，假设系统使用基于外部归并排序的投影算法，内存缓冲中可以用于排序的块数为 5，每次从一个归并段仅读取 1 块数据，计算该排序算法磁盘块传输代价和磁盘搜索代价（忽略最后结果写到外存的开销）；

   传输代价：$220*(2*\lceil \log_{(5-1)}^{(220/5)} \rceil +1) = 1540$

   搜索代价：2*220/5+220*(2*3-1) =1188

2) 假设有一个查询，需要将 USER 和 BORROW 做连接操作，属性 uid 为关联属性。如果采用块嵌套循环连接（Block Nested Loops Join），请计算磁盘块传输代价和磁盘搜索代价，假设可用的内存有 7 块（设 USER 作为外关系）；

   传输代价：$\lceil 320/(7-2) \rceil *2500 \rceil +320=160320$

   搜索代价：$2*\lceil 320/(7-2) \rceil =128$

3) 给出 USER、BORROW 和 BOOK 三个关系做散列连接的策略，请估计连接结果的大小。

   只要不是 user 跟 book 先连接的策略就可以，答案不唯一。需给出类似课后题答案给出的流程

   User * Borrow*Book=75000

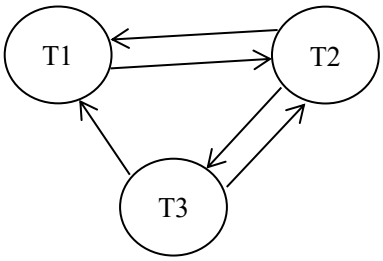
六、 请完成以下关于事务调度与并发控制的相关问题：

1) 考虑下面这个事务调度：

T1	T2	T3
read(B)		
		read(A)
	read(A)	
	A:= A - 100	
	write(A)	
		read(C)
		C := C - 50
		A := A + 100
		write(A)
read(A)		
B := B + 150		
A := A + 20		
		write(C)
write(A)		
write(B)		
	read(B)	
	commit	
commit		
		commit

a) 画出该调度的优先图，并判断该调度是否是冲突可串行化的；

优先图：



该调度不是冲突可串行化调度，原因：有环。

b) 假设只考虑 T1 与 T2 组成的调度，判断该调度是否是无级联调度？是否是可恢复调度？并分别说明理由。

不是无级联调度，原因： T1 事务 write(A)之后，T2 事务读取了 A 的值并在 T1 之前完成提交。

2) 考虑下面两个事务：

T1	T2
read(A) read(B) A := A - 1 read(A) C := C + A B := B + C write(B)	read(C) read(B) if C < B then C:= C - B B := B + 1 write(C) write(B)

a) 给事务 T1 和 T2 增加加锁和解锁指令，使它们遵从两阶段锁协议；

增加加锁和解锁指令

T1	T2
lock-S(A) **read**(A) **lock-S**(B) **read**(B) A := A + 1 **lock-X**(C) **read**(C) C := C + A B := B + C **write**(B) **unlock**(C) **unlock**(B) **unlock**(A)	**lock-X**(A) **read**(A) **lock-X**(B) **read**(B) **if** C < B && B > 0 **then** C := C + B B := B + 1 **write**(C) **write**(B) **unlock**(B) **unlock**(C)

b) 这两个事务的执行可能引起死锁吗？如果不可能，请说明理由；如果可能，请给出一个出现死锁调度的例子。

这两个事务的执行可能引起死锁，下面给出的就是一个死锁调度：

T1	T2
lock-S(A) **read**(A) **lock-S**(B) **read**(B) **lock-X**(C) **read**(C) …	 **lock-X**(C) **read**(C) **lock-X**(B) **read**(B) …

3) 考虑下图所示的一个部分调度，该调度遵循时间戳协议，

T1	T2	T3
read(A)	write(B)	read(B) write(A)

a) 请写出完成以上指令后，数据项 A 和 B 的读写时间戳；

R-timestamp(A) = TS(T3)

W-timestamp(A) = TS(T1)

R-timestamp(B) = TS(T3)

W-timestamp(B) = TS(T2)

b) 如果现在事务 T2 紧接着发出 write(A)指令，请分别根据时间戳排序协议和 Thomas 写规则判断系统会如何响应。

TS(T2) 小于 TS(T1)，所以当 T2 发出 write(A)指令时，TS(T2) 小于 W-timestamp(B)，根据 Thomas 写规则，write(B)操作被忽略。

七、 **假设某数据库系统在数据库崩溃重启后使用恢复算法进行恢复，考虑下面数据库崩溃前日志：**

<T0 start>
<T0, B, 500, 1300>
<T0, A, 200, 400>
<T1 start>
<T1, C, 900, 450>
<T1, A, 400, 300>
<checkpoint {T0, T1}>
<T0, commit >
<T2 start>
<T2, B, 1300, 650>

假设当以上日志输出到稳定存储器后，系统发生崩溃。在系统重启后，需要对数据库进行恢复，请完成以下问题：

1) 数据库崩溃重启后如何恢复，包含哪几个阶段的操作，并简述这几个阶段的主要任务；

系统崩溃重启恢复主要包括两个阶段：Redo 和 Undo

Redo 阶段：对稳定存储中的日志由上之下遍历重做日志中的事务的操作，遍历到 checkpoint 记录时，将在 checkpoint list 中的事务 T0，T1 加入到 undo list 中；继续向下读取、重做日志中的操作，当遍历到<T0, commit>的记录，将 T0 从 undo list 中删除；然后继续向下遍历，当发现<T2, start>记录将 T2 加入到 undo list 中；继续向下遍历、重做日志中的事务的操作，直到日志 redo 完成。

Undo 阶段：根据 redo 阶段生成的 undo list{T1, T2}，对 undo list 中的事务通过从下往上遍历日志进行回滚，直到 undo list 中为空后停止。

2) 如果系统正常完成了恢复过程，请列出恢复结束后的日志。

正常恢复后的日志：

<T2, B, 1300>

<T2, abort>

<T1, A, 400>

<T1, C, 900>

<T1, abort>