一、下面是某个公司人事数据库的两个全局关系

EMP={Eno, Ename, Title, Salary, Addr, Phone, Dno}; DEPT={Dno, Dname}该公司共有 3 个部门，Dno 分别为 0，1，2。要求将 DEPT 关系和 EMP 关系的部分属性（Ename, Addr, Phone）保存在部门 0 的场地上，EMP 关系的部分属性（Title, Salary）保存在所在部门场地上。根据上述要求，

1. 将全局模式进行分片，写出分片定义和分片条件。

DEPT 表：

DEPT 表保存在部门 0 的场地，因此 DEPT 表无需分片

EMP 表：

E_1 和 E_2 为 EMP 表的垂直分片

$$E_1 = \Pi_{Eno,Ename,Addr,Phone}(EMP)$$
$$E_2 = \Pi_{Eno,,Title,Salary,Dno}(EMP)$$

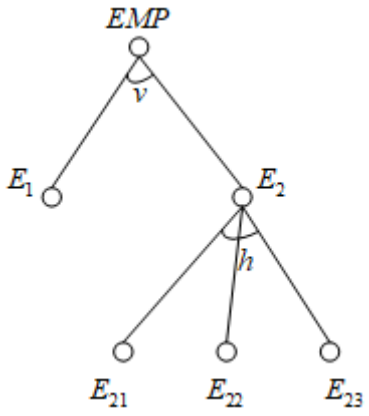
E_{21}、 E_{22} 和 E_{23} 为 E_2 表的水平分片

$$E_{21} = \sigma_{Dno=0}(E_2)$$
$$E_{22} = \sigma_{Dno=1}(E_2)$$
$$E_{23} = \sigma_{Dno=2}(E_2)$$

2. 指出各分片的类型，并画出分片树。

DEPT 表无分片

EMP 表既包括水平分片又包括垂直分片为混合分片

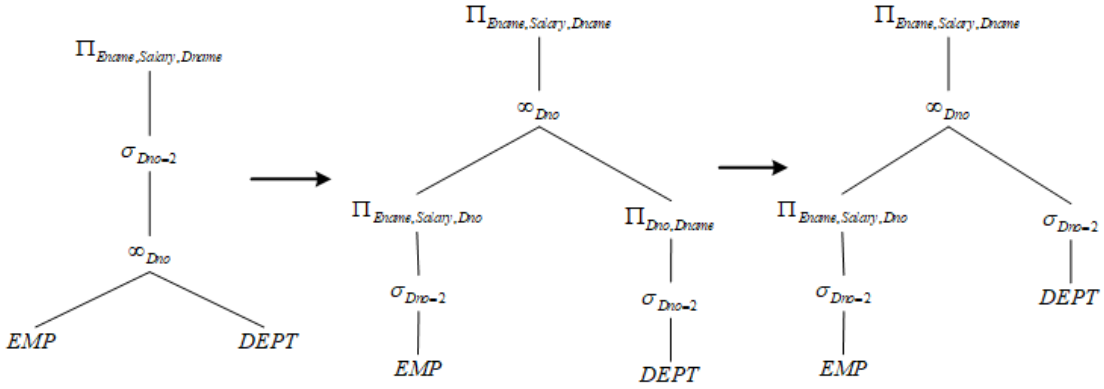


3. 对查询 SELECT Ename, Salary, Dname FROM Emp, Dept WHERE Dno=2. 进行全局优化，画出优化后的全局查询树，要求写出中间过程，

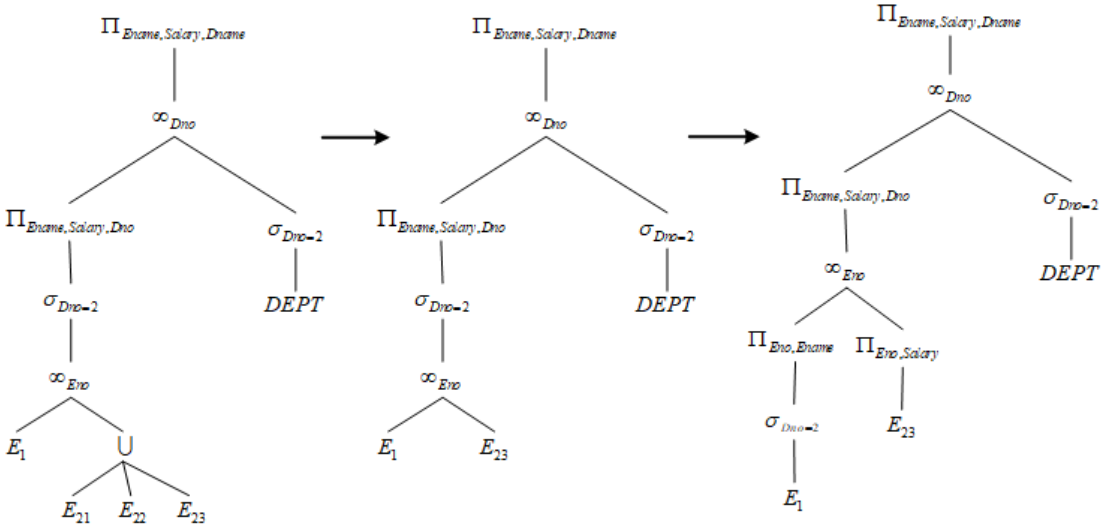
关系代数：

$$\Pi_{Ename,Salary,Dname}\left(\sigma_{Dno=2}\left(EMP\infty_{Dno}DEPT\right)\right)$$

全局查询树：



4. 进行分片优化，画出优化后的分片查询树，要求写出中间过程。



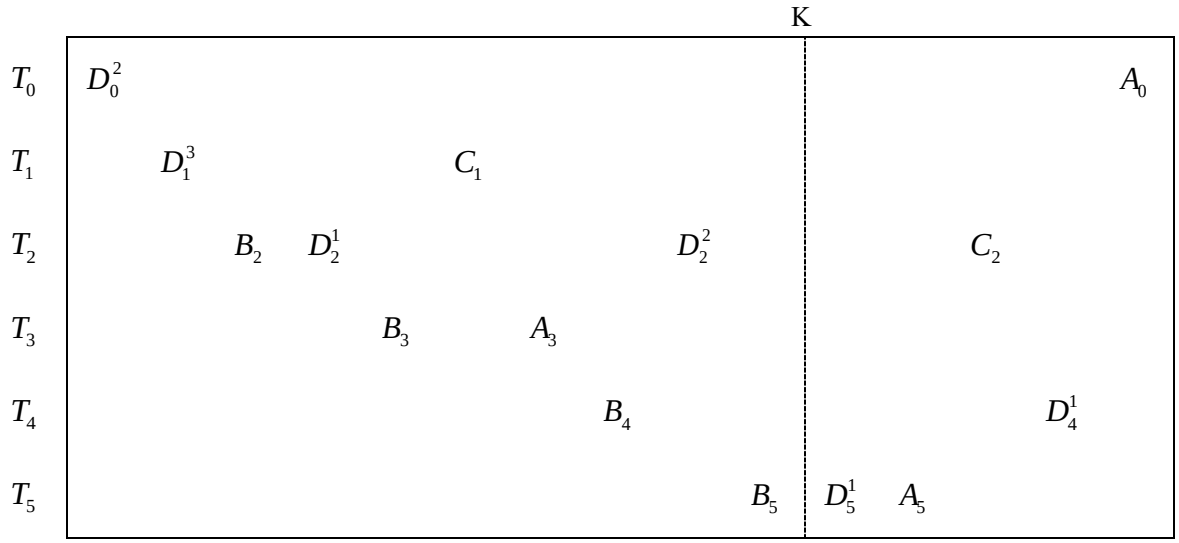
二、下面是一个数据库系统出现故障时，日志文件中的信息：

Crash

D_0^2	D_1^3	B_2	D_2^1	D_2^1	C_1	A_3	A_3	D_2^2	B_5	K	D_5^1	A_5	C_2	D_4^1	A_0	

根据上述 log 信息，完成下面的处理：

1. 画出对应的事务并发执行图。



2. 说明检查点的作用和检查点时刻数据库需要完成的主要操作。

作用：检查点是在日志中年周期设定的操作标志，目的是减少系统故障后的恢复的工作量。在检查点上，需要完成的操作包括：首先，将日志缓冲区的内容写入外存中的日志；然后，在外存日志中登记一个检查点记录；接下来，将数据库缓冲区的内容写入外存数据库中；最后，把外存日志中检查点的地址写入重启动文件，使检查点以前的工作永久化。

3. 确定出反做（undo）和重做（redo）事务集（写出详细过程）。

1) 初始化重做表（redo 表）和反做表（undo 表）；

2) 活动事务为在检查点上没有结束的事务，即T_0、T_2、T_4和T_5为活动事务，并放入 undo 表中。因此 redo 表={}，undo 表={活动事务}={T_0、T_2、T_4、T_5}；

3) 确定 redo 表和 undo 表，从检测点开始正向扫描日志文件。如有新事物开始放入 undo 表，如有事务提交则将 undo 表中的T_j移入 redo 表。即事务提交（Commit）后更新为 undo 表={T_2}，redo 表={T_0、T_4、T_5}；

4. 叙述 undo 和 redo 思想，详细写出其基于日志的数据处理过程。

反做和重做是数据库事务恢复过程中采用的两个典型策略。反做也称撤销是将一个数据项的值恢复到其修改之前的值，即取消一个事务所完成的操作结果

当一个事务尚未提交时，如果缓冲区管理器允许该事务修改过的数据写到外存数据库，一旦此事务出现故障需废弃时，就需对被这个事务修改过的数据项进行反做，即根据日志文件恢

复到前像。反做的目的时保持数据库的原子性。
重做是将一个数据项的值恢复到其修改的值，即恢复一个事务的操作结果。当一个事务提交时，如果缓冲区管理器允许该事务修改过的数据不立刻写到外存数据库，一旦此事务出现故障需对其修改的数据项进行重做，即根据日志文件将其恢复到后像，目的是保持数据库的耐久性。

处理过程

1) 局部恢复管理器 LRM 首先从重启动文件中取得最近检查点的地址，然后建立两个表——重做表和反做表，初始化这两个表：重做表初态为空，反做表初态为检查点上的活动事务；
2) 确定反做事务表：在日志中从检查点向前搜索直到日志结尾找出只有 B 记录而没有 C 的事务写入反做事务表；
3) 确定重做事务表：从检查点向前搜索，找出既有 B 记录也有 C 记录的事务直到日志结尾；
4) 反做反做事务表中的所有事务，根据日志反向进行撤销操作直到到达事务的 B；
5) 重做重做事务表中的所有事务，根据日志，从检查点正向进行重做操作，直到到达相应事务的 C。

三、对 3 个关系 R，S 和 T 的分布式连接 $R\infty S\infty T$，已知有如下的概要图：

Card(R)=500 S_1

	A
Length	4
Val	500

Card(S)=4000 S_2

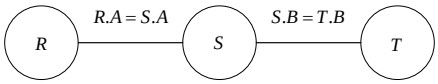
	A	B
Length	4	4
Val	2000	1000

Card(T)=200 S_3

	B
Length	4
Val	200

假设通信代价系数 $C_0=0$，$C_1=1$，$\mathrm{DOM}(R,A)\subseteq\mathrm{DOM}(S,A)$，$\mathrm{DOM}(T,B)\subseteq\mathrm{DOM}(S,B)$。

按照 SDD-1 半连接优化算法，逐步求出半连接优化集和最终执行场地。



1. 无一元操作

可能的连接集

$P_1=S\ltimes R$

$P_2=S\ltimes T$

$P_3=R\ltimes S$

$P_4=T\ltimes S$

初始化利益代价表

$P_1=S\ltimes R$

$\mathrm{Cost}_1=\mathrm{Val}(R,A)\times\mathrm{Length}(R.A)=500\times4=2KB$

$\because \mathrm{DOM}(R,A)\subseteq\mathrm{DOM}(S,A)$

$\therefore \rho_1=\mathrm{Val}(R,A)/\mathrm{Val}(\mathrm{DOM(S,A)})=500/2000=0.25$

$\mathrm{Benefit}_1=(1-\rho_1)\mathrm{Card}(S)\times\mathrm{Length}(S)=0.75\times4000\times(4+4)=24KB$

$P_2=S\ltimes T$

$\mathrm{Cost}_2=\mathrm{Val}(T,A)\times\mathrm{Length}(T.A)=200\times4=0.8KB$

$\because \mathrm{DOM}(T,B)\subseteq\mathrm{DOM}(S,B)$

$\therefore \rho_2=\mathrm{Val}(T,B)/\mathrm{Val}(\mathrm{DOM(S,B)})=200/1000=0.2$

$\mathrm{Benefit}_2=(1-\rho_2)\mathrm{Card}(S)\times\mathrm{Length}(S)=0.8\times4000\times(4+4)=25.6KB$

$P_3=R\ltimes S$

$\mathrm{Cost}_3=\mathrm{Val}(S,A)\times\mathrm{Length}(S.A)=2000\times4=8KB$

$\because \mathrm{DOM}(R,A)\subseteq\mathrm{DOM}(S,A)$

$\therefore \rho_3=1$

$\mathrm{Benefit}_3=0$

$P_4 = T \ltimes S$

$\text{Cost}_4 = \text{Val}(S,B) \times \text{Length}(S.B) = 1000 \times 4 = 4KB$

$\because \text{DOM}(T,B) \subseteq \text{DOM}(S,B)$

$\therefore \rho_4 = 1$

$\text{Benefit}_4 = 0$

因此初始的利益代价表如下：

半连接	Cost	ρ	Benefit
P1	2KB	0.25	24KB
P2	0.8KB	0.2	25.6KB
P3	8KB	1	0
P4	4KB	1	0

受益的半连接集 P={P1,P2}

2. 选择半连接

1) 循环 1

选择利益代价最小者 P2，将其加入策略集 P' 中 $P' = \{P_2\}$

重新计算概要图

$\text{Card}(S') = \rho_2 \text{Card}(S) = 0.2 \times 4000 = 800$

非选择谓词 A

$\because \text{Val}(S,A) > 2\text{Card}(S')$

$\therefore \text{Val}(S',A) = \text{Card}(S') = 800$

选择谓词 B

$\text{Val}(S',B) = \rho_2 \text{Val}(S,B) = 0.2 \times 1000 = 200$

概要图更新为

Card(R)=500 S_1

	A
Length	4
Val	500

$\text{Card}(S') = 800$ S_2

	A	B
Length	4	4
Val	800	200

Card(T)=200 S_3

	B
Length	4
Val	200

重新计算利益代价表

$P_1 = S' \ltimes R$

$P_2 = S \ltimes T$ √

$P_3 = R \ltimes S'$

$P_4 = T \ltimes S'$

$P_1 = S' \ltimes R$

$\text{Cost}_1 = \text{Val}(R, A) \times \text{Length}(R.A) = 500 \times 4 = 2KB$
$\because \text{DOM}(R, A) \subseteq \text{DOM}(S', A)$
$\therefore \rho_1 = \text{Val}(R, A) / \text{Val}(S', A) = 500 / 800 = 0.625$
$\text{Benefit}_1 = (1 - \rho_1)\,\text{Card}(S') \times \text{Length}(S') = 0.375 \times 800 \times (4 + 4) = 2.4KB$

$P_3 = R \ltimes S'$

$\text{Cost}_3 = \text{Val}(S', A) \times \text{Length}(S'.A) = 800 \times 4 = 3.2KB$
$\because \text{DOM}(R, A) \subseteq \text{DOM}(S', A)$
$\therefore \rho_3 = 1$
$\text{Benefit}_3 = 0$

$P_4 = T \ltimes S'$

$\text{Cost}_4 = \text{Val}(S', B) \times \text{Length}(S'.B) = 1000 \times 4 = 4KB$
$\rho_4 = \text{Val}(S', B) / \text{Val}(T, B) = 1$
$\text{Benefit}_4 = 0$

更新后的利益代价表如下：

半连接	Cost	ρ	Benefit
P1	2KB	0.625	2.4KB
P2	0.8KB	None	None
P3	3.2KB	1	0
P4	0.8KB	1	0

受益的半连接集 P={P1 }

2) 循环 2

选择利益代价最小者 P1，将其加入策略集 P' 中 $P' = \{P_2, P_1\}$

重新计算概要图

$\text{Card}(S'') = \rho_1\,\text{Card}(S') = 0.625 \times 800 = 500$

选择谓词 A

$\text{Val}(S'', A) = \rho_1\,\text{Val}(S', A) = 0.625 \times 800 = 500$

非选择谓词 B

$\because \text{Card}(S'') > 2\,\text{Val}(S', B)$
$\therefore \text{Val}(S'', B) = \text{Val}(S', B) = 200$

概要图更新为

Card(R)=500 S_1

	A
Length	4
Val	500

$\text{Card}(S'')=500$ S_2

	A	B
Length	4	4
Val	500	200

Card(T)=200 S_3

	B
Length	4
Val	200

重新计算利益代价表

$P_1 = S' \ltimes R \surd$

$P_2 = S \ltimes T \surd$

$P_3 = R \ltimes S''$

$P_4 = T \ltimes S''$

$P_3 = R \ltimes S''$

$\text{Cost}_3 = \text{Val}(S'', A) \times \text{Length}(S''.A) = 500 \times 4 = 2KB$

$\rho_3 = \text{Val}(S'', A) / \text{Val}(R, A) = 1$

$\text{Benefit}_3 = 0$

$P_4 = T \ltimes S''$

$\text{Cost}_4 = \text{Val}(S'', B) \times \text{Length}(S''.B) = 200 \times 4 = 0.8KB$

$\rho_4 = \text{Val}(S'', B) / \text{Val}(T, B) = 1$

$\text{Benefit}_4 = 0$

更新后的利益代价表如下：

半连接	Cost	ρ	Benefit
P1	2KB	None	None
P2	0.8KB	None	None
P3	2KB	1	0
P4	0.8KB	1	0

无受益半连接集循环结束

3. 计算各场地数据量

$\text{Size}(S_1) = \text{Size}(R) = 500 \times 4 = 2KB$

$\text{Size}(S_2) = \text{Size}(S'') = 500 \times 8 = 4KB$

$\text{Size}(S_3) = \text{Size}(T) = 200 \times 4 = 0.8KB$

可以看出场地 S_2 包含数据量最多

选择场地 S_2 为执行场地

$$\text{Cost} = \text{Cost}(Semijion) + \text{Cost}(assembly) = (2KB + 0.8KB) + (2KB + 0.8KB) = 5.6KB$$

半连接优化集

$$P' = \{P2, P1\} = \{S \ltimes T, S' \ltimes R\}$$

最终执行场地为 S_2

四、设数据项 x，y 存放在 S1 场地，u，v 存放在 S2 场地。有分布式事务 T1 和 T2，判断下面的每个执行是否是局部可串行的，是否是全局可串行的，并分别说明理由。

1. 执行 1：在 S1 场地 R1(x)R2(x)W2(y)W1(x)，在 S2 场地 R1(u)W1(u)R2(v)W2(u)。

H1: R1(x)R2(x)W2(y)W1(x)

H2: R1(u)W1(u)R2(v)W2(u)

H1：设可串行化历程 H:R2(x)W2(y)R1(x)W1(x)

冲突 H1　　H

R2(x)<W1(x)　↔　R2(x)<W1(x)

即 H1 与 H 等价，则 H1 可串行化且 T2<T

H2：T2 的操作在 T1 之后

所以 H2 为可串行化且 T1<T2

由于两个局部历程事务串行化而顺序不一致因此全局不可串行化

2. 执行 2：在 S1 场地 R1(x)R2(x)W1(x)W2(y)，在 S2 场地 W2(u)R1(u)R2(v)W1(u)。

H1: R1(x)R2(x)W1(x)W2(y)

H2: W2(u)R1(u)R2(v)W1(u)

H1：设可串行化历程 H: R2(x)W2(y)R1(x)W1(x)

冲突 H1　　H

R2(x)<W1(x)　↔　R2(x)<W1(x)

即 H1 与 H 等价，则 H1 可串行化且 T2<T

H2: 设可串行化历程 H': W2(u)R2(v)R1(u)W1(u)

冲突 H2　　H'

W2(u)<R1(u)　↔　W2(u)<R1(u)

W2(u)<W1(u)　↔　W2(u)<W1(u)

即 H2 与 H'等价，则 H2 可串行化且 T2<T

Site1		Site2	
T1	T2	T1	T2
R1(x)	W2(y)		W2(u)
W1(x)		R1(u)	R2(v)
	R2(x)	W1(u)	

但是，T1 在没有获得对 u 的锁之前不会释放对 x 的锁，而 T2 在没有获得对 x 锁之前不会释放对 u 的锁。T1 与 T2 之间发生了死锁，故根据 2PL 协议 T1 和 T2 在全局不可串行化

3. 若在 S1 上有操作序列 R1(x)W1(y) R1(y)W1(y)，在 S2 场地上有操作序列 R1(u)R1(v)W1(u)，假设 T1 的操作实行完成后将进行提交，请按照 2PC 协议说明 T1 的提交处理过程，并要求按照严格 2PL 协议，对 T1 的错做处理加上显示的封锁操作和解锁操作。用 R11(x)表示对 x 加读锁，W11(x)表示对 x 加写锁，U11(x)表示解锁。

S1: R1(x)W1(y) R1(y)W1(y)

S2: R1(u)R1(v)W1(u)

数据项 x, y	数据项 u, v
Site S1	Site S2
T1	T2
R11(x)	R11(u)
R1(x)	R1(u)
W11(x)	R11(v)
W1(x)	R1(v)
R11(y)	W11(u)
R1(y)	W1(u)
W11(y)	U11(v)
W1(y)	U11(u)
U11(x)	
U11(y)	

五、理解分布式事务的 ACID 四个特性，说明它同集中式数据库的异同。此外，分布式事务的实现模型主要有哪些？

1) 分布式事务的 ACID 四个特性分别为原子性（atomicity）、一致性（consistency）、隔离性（isolation）和耐久性（durability）。

2) 与集中式数据库：继承了集中式数据库管理系统中事务的特性，但在事务执行过程中，分布式事务除了要保证各个子事务进行协调，决定事务的提交与撤销，以确保全局事务的 ACID 特性。另外分布式事务除了要考虑对数据的存取操作序列之外，还需要涉及大量的通信原语和控制报文。

3) 分布式事务的实验模型

- 进程模型：全局事务必须为每一个子事务在相应场地创建代理者进程，由其执行该场地上有关操作
- 服务器模型：事务的每一个执行场地创建服务器进程，用于执行发生在该场地上的所有子事务