数据库系统复习讲座

1 关系代数

1.1 在教学管理数据库中，有如下三个关系：
学生信息表：S(S#，SNAME，AGE，SEX)

课程表：C(C#，CNAME，TEACHER)

选课表：SC(S#，C#，GRADE)

其中S#、C#为S、C表的主码，(S#，C#)是SC表的主码，也分别是参照S、C表的外码。使用关系代数表达式回答下列问题：

（1）查询学习过课程号为001或002课程的学生的学号和成绩。

$$\Pi_{S\#,GRADE}(\sigma_{C\#='001'\vee C\#='002'}(SC))$$

（2）查询学习过“Database Systems”，但没有学习过“Data Mining”的学生的姓名和年龄。

$$\Pi_{SNAME,AGE}(\sigma_{CNAME='Database\ Systems'}(S\bowtie SC\bowtie C))-\Pi_{SNAME,AGE}(\sigma_{CNAME='Data\ Mining'}(S\bowtie SC\bowtie C))$$

（3）查询老师Wang所教过的学生中成绩为90分以上（包括90分）的学生学号。

$$\Pi_{S\#}(\sigma_{TEACHER='Wang'\wedge GRADE\geq 90}(SC\bowtie C))$$

（4）查询选修了全部课程的学生姓名。

$$\Pi_{SNAME}(S\bowtie(\Pi_{S\#,C\#}(SC)\div\Pi_{C\#}(C)))$$

（5）查询所选课程的平均分低于80的学生的学号及所选课程的平均分。

$$\Pi_{S\#,AvgGrade}(\sigma_{AvgGrade\leq 80}(_{S\#}\mathcal{G}_{avg(GRADE)\ as\ AvgGrade}(SC)))\ 或$$

$$\Pi_{S\#,AvgGrade}(\sigma_{AvgGrade\leq 80}(\gamma_{S\#;avg(GRADE)\rightarrow AvgGrade}(SC)))$$

2 SQL

2.1 在教学管理数据库中，有如下三个关系：

学生信息表：Student(Sno，Sname，Ssex，Sage，Sdept)

课程表：Cource(Cno，Cname，Teacher)

选课表：SC(Sno，Cno，Grade)

其中Sno、Cno为Student、Course表的主码，(Sno，Cno)是SC表的主码，也分别是参照Student、Course表的外码。使用SQL语言回答以下问题：

（1）查询所有系名。

```
SELECT DISTINCT Sdept
FROM Student;
```

（2）查询计算机系(CS)中姓名首字母为E的学生的学号和姓名。

```
SELECT Sno, Sname
FROM Student
WHERE Sdept = 'CS' AND Sname LIKE 'E%';
```

(3) 查询所在院系名为2个字母且首字母为C的学生的学号和姓名。

```
SELECT Sno, Sname
FROM Student
WHERE Sdept LIKE 'C_';
```

(4) 查询选了课但还未取得成绩的学生学号。

```
SELECT Sno
FROM SC
WHERE Grade IS NULL;
```

(5) 查询选修了"Database Systems"且分数大于等于90的全体学生的信息，结果按所在系升序排列，同一个系的学生按年龄降序排列。

```
SELECT *
FROM Student, Course, SC
WHERE Student.Sno = SC.Sno AND Course.Cno = SC.Cno
AND Cname = 'Database Systems' AND Grade >= 90
ORDER BY Sdept ASC, Sage DESC;
```

(6) 查询选修了1002号课或3006号课的学生的学号。

```
SELECT Sno FROM SC WHERE Cno = '1002' UNION
SELECT Sno FROM SC WHERE Cno = '3006'
```

(7) 查询课程"Database Systems"得分最高的前2名学生的学号和成绩。

```
SELECT Sno, Grade
FROM Course, SC
WHERE Course.Cno = SC.Cno AND Cname = 'Database Systems'
ORDER BY Grade DESC LIMIT 2;
```

(8) 统计每个系的男生人数和女生人数。

```
SELECT Sdept, Ssex, Count(*)
FROM Student
GROUP BY Sdept, Ssex;
```

(9) 查询2门及以上课程得分超过80的学生的学号及这些课程的平均分。

```
SELECT Sno, AVG(GRADE)
FROM SC
WHERE Grade > 80
GROUP BY Sno HAVING COUNT(*) >= 2
```

(10) 查询选修了1002号课的学生的姓名。

```
SELECT Sname
FROM Student, SC
Where Student.Sno = SC.Sno AND Cno = '1002';
```

(11) 查询选课了的学生的学号和姓名。

```
SELECT Student.Sno, Sname
FROM Student LEFT OUTER JOIN SC ON Student.Sno = SC.Sno
WHERE Cno IS NOT NULL;
```

(12) 查询和Elsa在同一个系学习的学生的学号和姓名。

```
SELECT Sno, Sname FROM Student WHERE Sdept IN
(SELECT Sdept FROM Student WHERE Sname = 'Elsa');
```

(13) 查询Amy选过，但Bob没有选过的课程的编号。

```
SELECT Course.Cno
FROM Student, Course, SC
WHERE Student.Sno = SC.Sno AND Course.Cno = SC.Cno
AND Sname = 'Amy' AND Course.Cno NOT IN
(SELECT Course.Cno
FROM Student, Course, SC
WHERE Student.Sno = SC.Sno AND Course.Cno = SC.Cno
AND Sname = 'Bob')
```

(14) 查询比计算机系全体学生年龄都大的学生的学号。

```
SELECT Sno FROM Student WHERE Sage > ALL
(SELECT Sage FROM Student WHERE Sdept = ‘CS);
```

(15) 查询选修了全部课程的学生的学号。

```
SELECT Sno FROM Student WHERE NOT EXISTS (
SELECT * FROM Course WHERE NOT EXISTS (
SELECT * FROM SC
WHERE SC.Sno = Student.Sno AND SC.Cno = Course.Cno));
```

(16) 查询选修了2门以上课程的学生的学号和选课数。

```
SELECT Sno, Count(*) FROM SC
GROUP BY Sno HAVING COUNT(*) >= 2;
# 使用嵌套查询
SELECT Sno, T.Amt FROM
(SELECT Sno, COUNT(*) AS Amt FROM SC GROUP BY Sno) AS T
WHERE T.Amt >= 2;
```

(17) 创建视图CS_Student_on_DB，列出选修了3006号课的计算机系学生的学号、姓名和成绩。

```
CREATE VIEW CS_Student_on_DB AS
  SELECT Sno, Sname, Grade
  FROM Student NATURAL JOIN SC
  WHERE Sdept = 'CS' AND Cno = '3006';
```

(18) 将学号为MA-002的学生的年龄修改为20岁。

```
UPDATE Student SET Sage = 20 WHERE Sno = 'MA-002';
```

(19) 将计算机系学生的选课信息删除。

```
# 使用IN
DELETE FROM SC WHERE Sno IN
(SELECT Sno FROM Student WHERE Sdept = 'CS');
# 使用EXISTS
DELETE FROM SC WHERE EXISTS
(SELECT * FROM Student
WHERE Student.Sno = SC.Sno AND Sdept = 'CS');
```

（20）将计算机系学生的学号、姓名、选课数、平均分插入到关系CS_Grade(Sno, Sname, Amt, AvgGrade)中。

```
INSERT INTO CS_Grade
(SELECT Sno, Sname, COUNT(*), AVG(Grade)
FROM Student NATURAL JOIN SC
WHERE Sdept = 'CS'
GROUP BY Sno, Sname);
```

3 E-R图

3.1 工厂需建立一个管理数据库存储以下信息：
工厂：厂名、厂长姓名；

车间：车间号、车间主任姓名、地址、电话；

仓库：仓库号、仓库主任姓名、电话；

零件：零件号、重量、价格；

产品：产品号、价格。

上述实体存在如下联系：

一个工厂内有多个车间和多个仓库，一个车间或一个仓库都只能属于一个工厂；

一个车间生产多种产品，每种产品只能产自一个车间；

一个车间生产多种零件，一种零件也可能为多个车间所制造；

一个产品由多种零件组成，一种零件也可装配出多种产品；

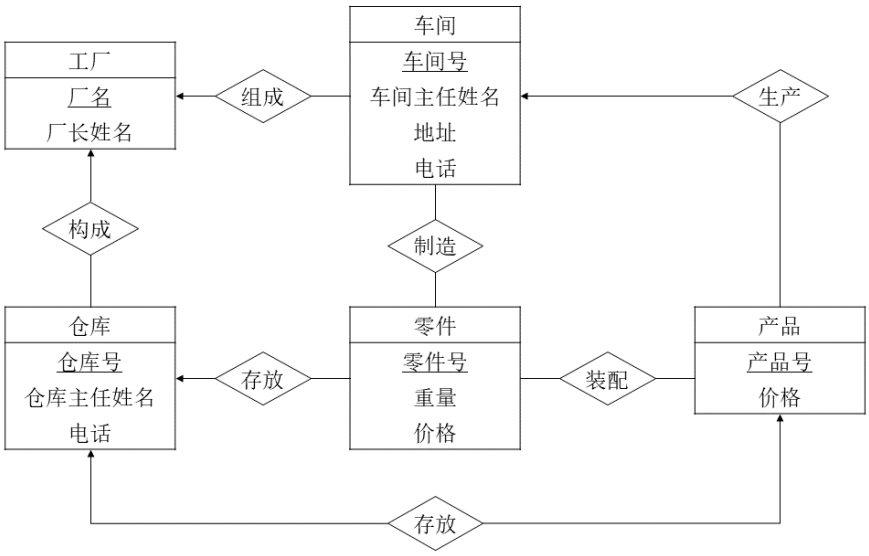
一个仓库只能存放一种产品，一种产品只能存入一个仓库；

一个仓库可以存放多种零件，一种零件只能存入一个仓库；

根据上述要求，完成如下工作：

（1）画出该系统的E-R图。

- 该系统的E-R图如图所示。



（2）写出对应的关系模式，标明主码。

- 对应的关系模式及各关系的主码如下所示。

  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)

  产品(<u>产品号</u>, 价格, 车间号, 仓库号)

  零件(<u>零件号</u>, 重量, 价格, 仓库号)

  制造(<u>车间号, 零件号</u>)

  装配(<u>产品号, 零件号</u>)

4 关系模式

4.1 已知关系模式$R(A,B,C,D,E,F,G,H)$上的函数依赖集合F如下：

$$F=\{A\to B,B\to D,AD\to EG,AGH\to C\}$$

回答下列问题：

（1）使用Armstrong公理证明F逻辑蕴含$A\to EG$。

- 由$F\models A\to B,F\models B\to D$可得$F\models A\to D$（传递律）；

  由$F\models A\to D$可得$F\models A\to AD$（增广律）；

  由$F\models A\to AD,AD\to EG$可得$F\models A\to EG$（传递率）。证毕。

（2）求A关于F的属性集闭包。

- $A_F^+=\{A,B,D,E,G\}$

（3）求关系R的候选键，并说明R最高属于第几范式。

- A,H为L类属性，F为N类属性，必包含在R的任一候选码中。

  又因为$(AFH)_F^+=\{A,B,C,D,E,F,G,H\}$，所以$AFH$为$R$的候选键。

（4）求F的最小覆盖（极小函数依赖集）。

- $F_c=\{A\to B,B\to D,A\to E,A\to G,AH\to C\}$

4.2 已知关系模式$R(A,B,C,D,E)$上的函数依赖集合F如下：

$$F=\{A\to B,A\to C,BC\to A,CD\to E,B\to D,E\to A\}$$

回答下列问题：

（1）计算属性集合BC关于F的闭包$(BC)_F^+$。

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

（2）找出R的全部候选键。

- 因为$A_F^+ = (BC)_F^+ = (CD)_F^+ = E_F^+ = \{A, B, C, D, E\}$，
  所以R的全部候选键为A, BC, CD, E。

（3）判断R属于第几范式。

- 由（2）知R的所有属性均为主属性，所以$R \in 3NF$。
  由于$F \models B \to D$，所以主属性D部分函数依赖于候选键BC，故$R \notin BCNF$。
  综上，$R \in 3NF$。

（4）计算F的最小覆盖。

- $F_c = \{A \to B, A \to C, CD \to E, B \to D, E \to A\}$

（5）给出R的一个既满足无损连接性，又满足函数依赖保持性的$3NF$分解。

- $\rho = \{R_1(A, B, C), R_2(C, D, E), R_3(B, D), R_4(A, E)\}$

4.3 已知关系模式$R(A, B, C, D, E)$上的函数依赖集合F如下：

$F = \{A \to C, C \to D, B \to C, DE \to C, CE \to A\}$

回答下列问题：

（1）求R的全部候选键。

- 因为B, E为R的L类属性，所以B, E必包含在R的任一候选键中。
  又因为$(BE)_F^+ = A, B, C, D, E$，所以BE为R的全部候选键。

（2）判断$\rho = \{R_1(A, D), R_2(A, B), R_3(B, C), R_4(C, D, E), R_5(A, E)\}$是否为无损连接分解。

- rho的无损连接判断结果如下所示，由于表中没有一行是(a, b, c, d, e)，由此判定ρ不具有无损连接性。

R_i	A	B	C	D	E
$R_1(A, D)$	a		c	d	
$R_2(A, B)$	a	b	c	d	
$R_3(B, C)$		b	c	d	
$R_4(C, D, E)$	a		c	d	e
$R_5(A, E)$	a		c	d	e

（3）保持无损连接性，将R分解为$BCNF$。

- $\rho = \{R_1(A, C), R_2(B, D), R_3(A, B, E)\}$

5 哈希表

5.1 （可扩展哈希表）利用可扩展hash方法对以下记录进行hash存储：3, 5, 7, 12, 16。

设hash函数$h(x) = x \ mod \ 8$，其中散列函数$h(k)$是一个b(足够大)位二进制序列，序列的前d位用作索引，来区分每个元素属于哪个桶。

现要求每个桶至多包含2个元素，以上元素按从左往右的顺序依次添加。开始时只使用序列的前1位作为索引（即$d = 1$），当桶满时进行分裂，d相应增大。请画出添加完以上所有元素后，最终的索引结构。

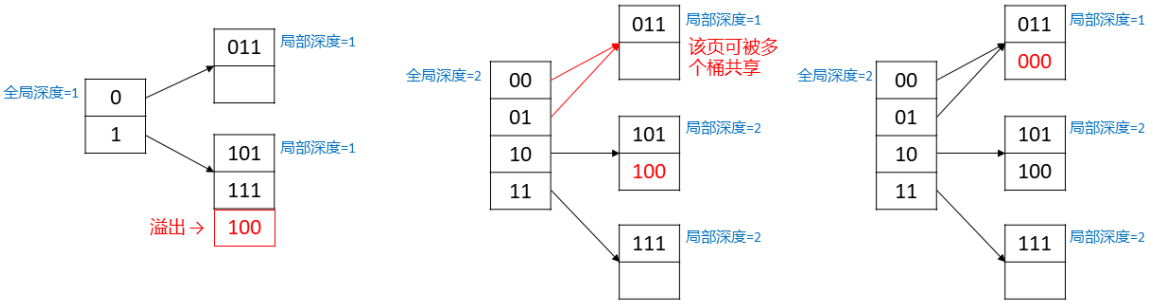
- 以下给出各关键步骤的索引结构。

① 插入3, 5, 7时，未发生页分裂，索引位数不变。

3 mod 8 = 3（011）
5 mod 8 = 5（101）
7 mod 8 = 7（111）
12 mod 8 = 4（100）
16 mod 8 = 0（000）

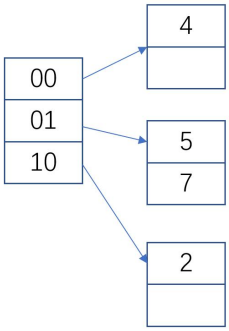
② 插入12时，桶满进行分裂，此时全局深度=局部深度，故全局深度+1，表进行扩展，发生溢出的页局部深度也+1，根据索引重新分配表项。
　　由于原桶0指向的页未发生分裂，其局部深度 < 全局深度，故可被分裂后的桶00和01共享。

③ 最后插入16，由于桶00和01指向的页容纳得下这一索引项，故直接放入该页。



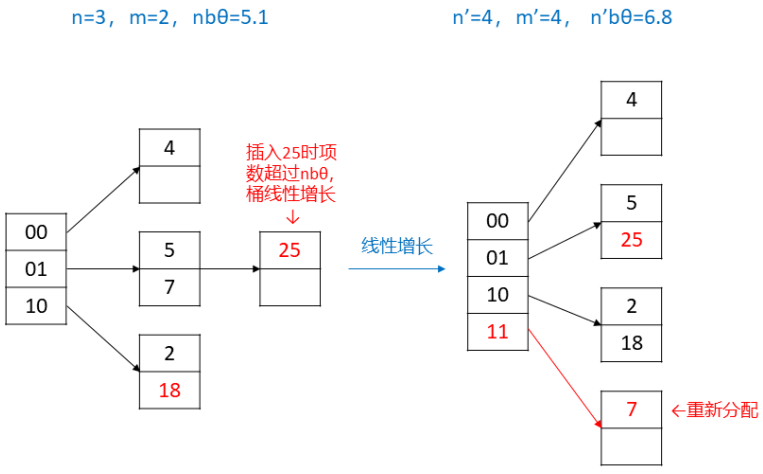
5.2 （线性哈希表）利用线性hash方法对以下记录进行hash存储，在初始hash表中加入以下数字：18, 25, 27, 36, 48, 56, 61。

请画出添加完以上所有元素后，最终的索引结构。线性hash表中最终容纳$nb\theta$个记录，$b = 2, \theta = 0.85$。初始哈希桶结构如下图。

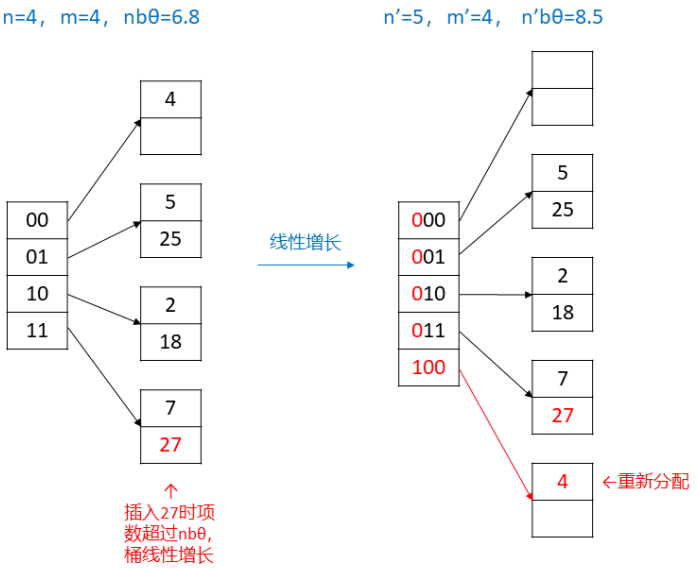


- 以下给出各关键步骤的索引结构。

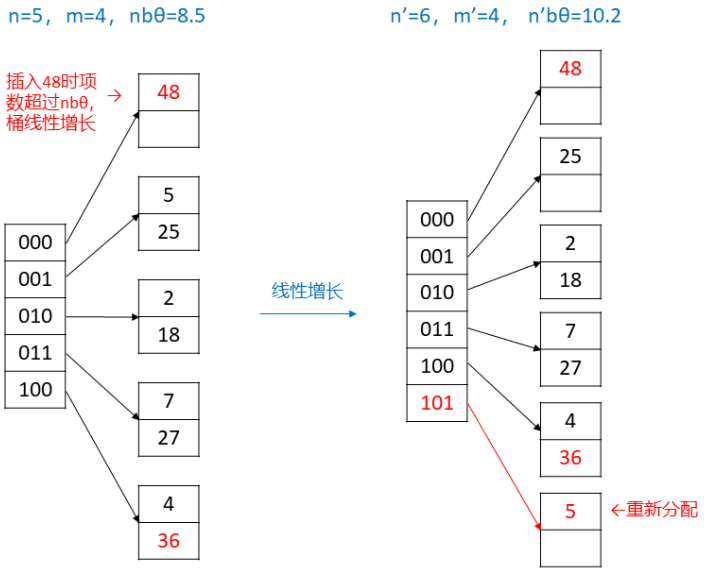
① 初始状态n=3，m=2，nbθ=5.1。
插入18和25。
18 mod 2m = 2（10）
25 mod 2m = 1（01）
当插入25后，表中索引项数6 > nbθ，桶进行线性增长。
n'=4，m'=4，n'bθ=6.8。
新创建的桶编号为11，故将原01号桶中应属于11号桶的索引项重新分配到11号桶中。
5 mod 2m' = 5 > n'，5 mod m' = 1，
故仍在01号桶中，
7 mod 2m' = 7 > n'，7 mod m' = 3，
故放入11号桶中，
25 mod 2m' = 1 < n'，故仍在01号桶中。



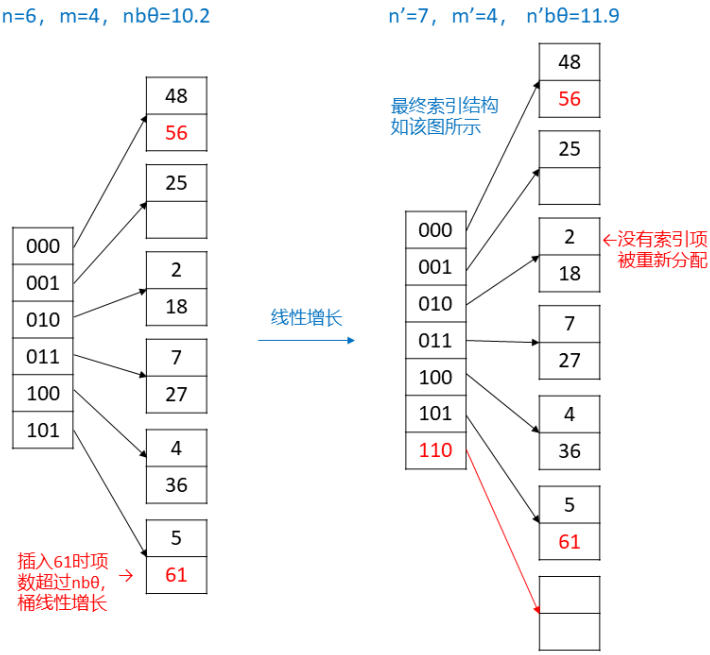
② n=4，m=4， nbθ=6.8。
插入27，放入27 mod 2m = 3（11）号桶中，此时表中索引项数为7 > nbθ，桶进行线性增长。
此时n'=5，m'=4， n'bθ=8.5。
新创建的桶编号为100，故将原00号桶中应属于100号桶的索引项重新分配到100号桶中。
4 mod 2m' = 4 < n'，故放入100号桶中。



③ n=5，m=4， nbθ=8.5。
插入36和48。
36 mod 2m = 4（100）
48 mod 2m = 0（000）
当插入48后，表中索引项数9 > nbθ，桶进行线性增长。
n'=6，m'=4， n'bθ=10.2。
新创建的桶编号为101，故将原001号桶中应属于101号桶的索引项重新分配到101号桶中。
5 mod 2m' = 5 < n'，故放入101号桶中，
25 mod 2m' = 1 < n'，故仍在001号桶中。

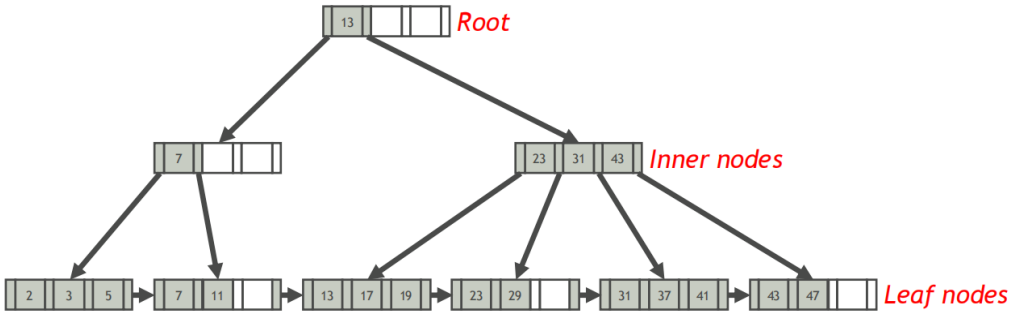


④ n=6，m=4， nbθ=10.2。
插入56和61。
56 mod 2m = 0（000）
61 mod 2m = 5（101）
当插入61后，表中索引项数11 > nbθ，桶进行线性增长。
n'=7，m'=4， n'bθ=11.9。
新创建的桶编号为110，故将原010号桶中应属于110号桶的索引项重新分配到110号桶中。
2 mod 2m' = 2 < n'，故仍在010号桶中，
18 mod 2m' = 2 < n'，故仍在010号桶中。



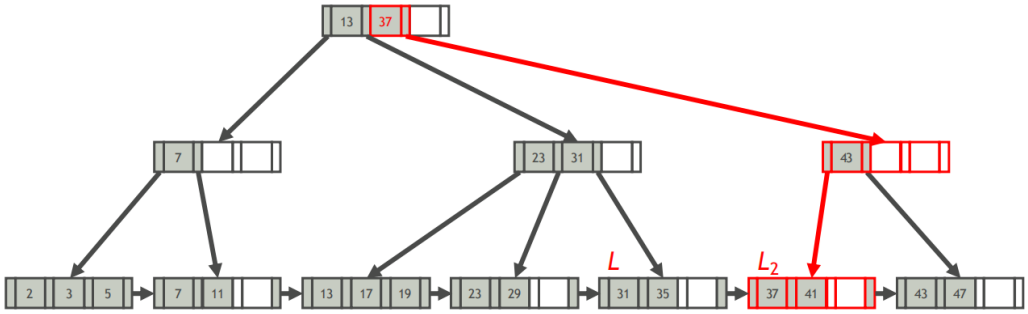
6 B+树

6.1 已知如下B+树：

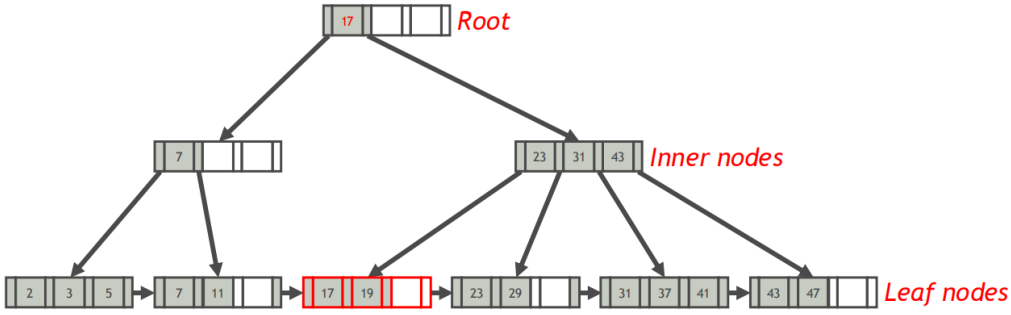


回答下列问题。

（1）插入键值为35的索引项后，该B+树变成什么样？请绘制出来。



（2）删除键值为13的索引项后，该B+树变成什么样？请绘制出来。



7 关系代数操作算法

7.1 已知关系$R(w,x),S(x,y),T(y,z)$的块数分别为5000, 10000, 10000。假设缓冲池有$M=101$个页可用，R,S,T上均无索引且未按连接属性排序。请回答下列问题。

（1）使用什么算法执行$R\bowtie S$最适合？

- Grace哈希连接算法最合适，因为*R*和*S*的块数都超过了*M*，一趟算法不可用；*R*和*S*无索引，索引连接不可用；*R*和*S*未排序，排序归并连接不可用。

（2）使用（1）中选择的算法执行$R\bowtie S$的I/O代价是多少？给出分析过程。

- $3B(R)+3B(S)=45000$。

在对R进行哈希分桶时，R的每块读1次，合计$B(R)$次I/O；

将R的桶全部写入文件，需$\sum_{i=1}^{M-1}B(R_i)\approx B(R)$次I/O；

在对S进行哈希分桶时，S的每块读1次，合计$B(S)$次I/O；

将S的桶全部写入文件，需$\sum_{i=1}^{M-1}B(S_i)\approx B(S)$次I/O；

使用一趟连接算法计算$R_i \bowtie S_i$的I/O代价是$B(R_i)+B(S_i)$。

所以总的I/O代价为$3B(R)+3B(S)$。

8 查询优化

8.1 设教学管理数据库有如下3个关系模式：

学生信息表：S(S#，SNAME，AGE，SEX)

课程表：C(C#，CNAME，TEACHER)

选课表：SC(S#，C#，GRADE)

其中S#、C#为S、C表的主码，(S#，C#)是SC表的主码，也分别是参照S、C表的外码。

用户有一查询语句：

```
SELECT SNAME
FROM S，SC，C
WHERE SC.S#=S.S# AND SC.C#=C.C# AND CNAME="数据库";
```

检索选学"数据库"课程的学生的姓名。

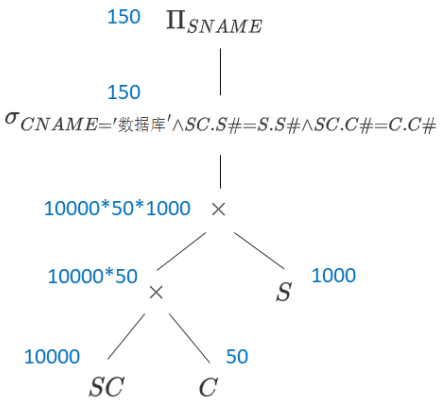
（1）写出以上SQL语句所对应的关系代数表达式。

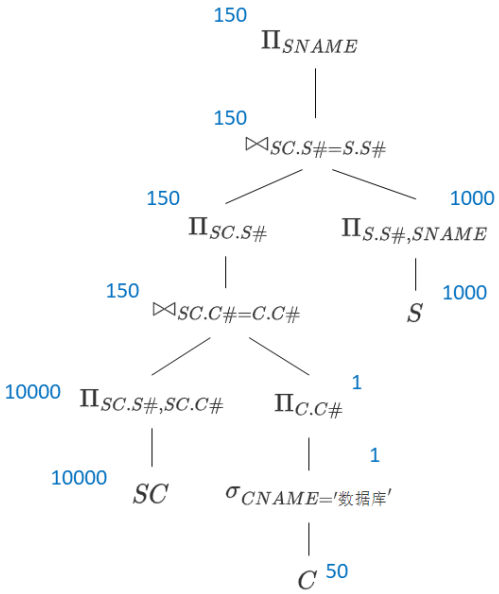
- $\Pi_{SNAME}(\sigma_{CNAME='数据库' \wedge SC.S\#=S.S\# \wedge SC.C\#=C.C\#}(SC \times S \times C))$

（2）画出上述关系代数表达式所对应的查询计划树。使用启发式查询优化算法，对以上查询计划树进行优化，并画出优化后的查询计划树。

（3）设SC表有10000条元组，C表有50条元组，S表有1000条元组，SC中满足选修数据库课程的元组数为150，计算优化前与优化后的查询计划中每一步所产生的中间结果大小。

- 优化前和优化后的查询计划树以及每一步所产生的中间结果大小如下图所示。





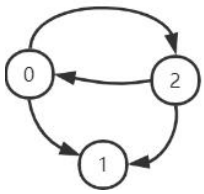
9 并发控制

9.1 考虑下面的三个事务T_0, T_1, T_2和它们的一个调度：

$S: r_0(A), w_0(A), r_2(A), W_2(A), r_1(A), r_0(B), r_2(B), w_0(B), w_2(B), R_1(B)$

判断S是否是冲突可串行化的调度？要求画出优先图并给出判断依据。

- 优先图如下图所示。因为优先图中存在环，所以S不是一个冲突可串行化调度。



9.2 设 T1、T2、T3 是如下三个事务：

```
T1:
A:=A+4
T2:
A:=A*3
T3:
A:=A*A
```

初始时A=2。设三个事务都遵守两段锁协议，按T2-T3-T1的顺序执行，为事务添加加锁和解锁指令，给出一个不产生死锁的可串行化调度，并给出最终A的结果。

- 两阶段锁协议要求每个事务的执行分两个阶段提出加锁和解锁请求：在增长阶段（加锁阶段），事务向锁管理器请求需要的锁，不能释放任何锁；在萎缩阶段（解锁阶段），事务释放它所获得的锁，不能获取任何新锁。事务的调度如下所示。最终A=40。

时间	**T1**	**T2**	**T3**
1		LOCK-X(A)	
2		READ(A)	
3		A:=A*3	
4		WRITE(A)	
5		UNLOCK(A)	
6			LOCK-X(A)
7			READ(A)
8			A:=A*A
9			WRITE(A)
10			UNLOCK(A)
11	LOCK-X(A)		
12	READ(A)		
13	A:=A+4		
14	WRITE(A)		
15	UNLOCK(A)		

10 故障恢复

10.1 已知某数据库采用即时更新方法（undo-redo 方法）记录WAL日志。设故障发生时WAL日志文件内容如下：

```
<T1, begin>
<T1, A, 114, 114514>
<T2, begin>
<T1, B, "hit", "hitcs">
<T1, commit>
<T3, begin>
<T3, B, "hitcs", "hitcsdb">
<T2, A, 114514, 1919810>
```

当系统重启后，DBMS基于该WAL日志文件进行故障恢复。回答下列问题：

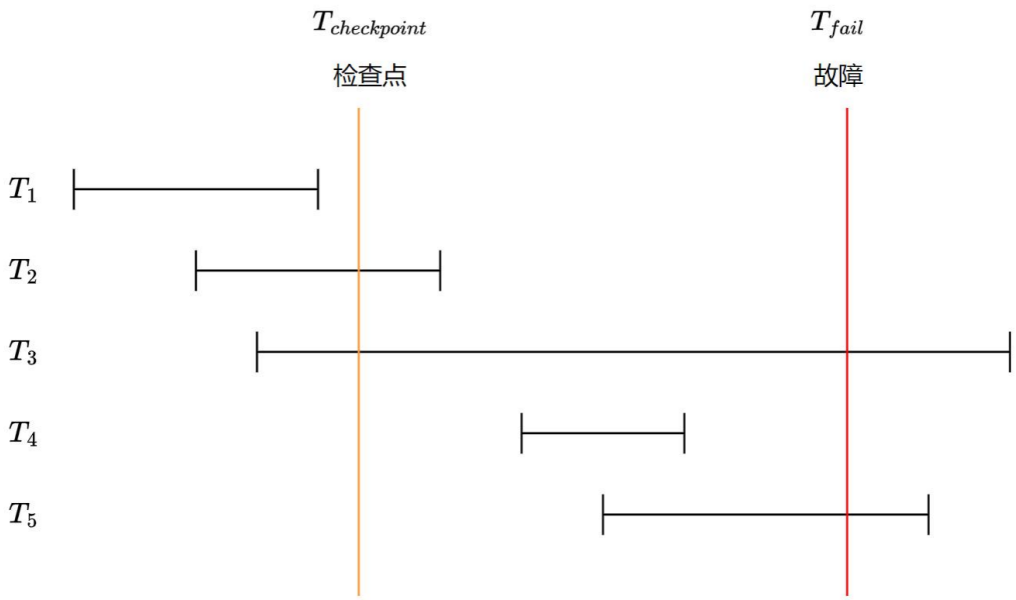
（1）当DBMS进行故障恢复时，需要对哪些事务进行undo？对哪些事务进行redo？

- 需要undo：T2, T3；需要redo：T1。

（2）当故障恢复完成时，对象A和B的值分别是什么？

- A的值为114514，B的值为“hitcs”。

10.2 使用检查点的数据库恢复系统将根据事务的状态和检查点的关系采取相对应的恢复策略，现在有事务T1 -T5，其执行过程如下图所示（线段左端和右端分别表示事务开始与提交），其执行过程中数据库系统发生如图所示的故障，请问在故障恢复时事务T1 -?5那些需要撤销，哪些需要重做，哪些不需要操作？



- 需要撤销（undo）：T3, T5；需要重做（redo）：T2, T4；不需要操作：T1。

预祝大家考试顺利，祝大家都能取得好成绩~

哈工大计算学部金牌讲师团 林之彦

2023年12月