浙江大学 2017－2018 学年 春夏 学期

《数据库系统》课程期末考试试卷

课程号：21121350，　开课学院：计算机学院

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式：√闭、开卷（请在选定项上打√），允许带一张 A4 纸笔记入场

考试日期：2018 年 7 月 3 日，考试时间：120 分钟

诚信考试，沉着应考，杜绝违纪。

考生姓名：_____学号：_____所属院系：_____

题序	一	二	三	四	五	六	七	八	总 分
得分									
评卷人									

Problem 1：Relational Model and SQL（15 points，3 points per part)

Following relational schemas represent information of a campus card database in a university.

card(cno:char(5), name:char(8), depart:char(10), balance:integer)
pos(pno:char(4), campus:char(8), location:char(10))
detail(cno:char(5), pno:char(4), cdate:date, ctime:time, amount:integer,
remark:char(10))

Following tables show an instance of the database:

card

cno	name	depart	balance
c0001	张帅	CS	100
c0002	李丽	EN	200
c0003	王浩	CS	300
c0004	刘萌	CS	400
c0005	赵亮	MA	500

pos

pno	campus	location
p001	玉泉	教育超市
p002	玉泉	四食堂
p003	玉泉	四食堂
p004	紫金港	教育超市
p005	紫金港	教育超市
p006	紫金港	一食堂

detail

cno	pno	cdate	ctime	amount	remark
c0001	p002	2018-07-01	08:10:10	6	餐饮
c0001	p002	2018-07-01	12:05:12	12	餐饮
c0001	p002	2018-07-01	17:30:20	20	餐饮
c0001	p001	2018-07-01	18:10:10	60	购物
c0002	p002	2018-07-02	08:10:10	8	餐饮
c0002	p001	2018-07-02	08:10:10	20	购物
c0003	p003	2018-07-02	08:10:10	25	餐饮

Given following SQL query on above relations：

```
select cno, name
from card natural join detail
where depart= 'CS' and (cdate , pno) in
                    (   select cdate, pno
                        from detail
                        where cno='c0002' )
```

Please answer following questions：

(1) Transform above query to a SQL statement without nested subquery.

(2) Transform above query to an equivalent relational algebra expression.

(3) Write a SQL statement to find out cards consumed in only one campus in 2018.

(4) Write a SQL statement to find out the pos in "紫金港" campus that has the maximum total amount of card consumption in 2018.

(5) Write a sequence of SQL statements to complete following transaction:
card "c0002" consumes 20 at pos "p001" at 2018-07-02 08:08:08

Answers of Problem 1：

Problem 2：E-R Model （11 points)

Please design an ER diagram for a takeaway platform（外卖平台）to capture information about **restaurants**, **customers**, **couriers**（快递员）**, orders , payments,** and **deliveries,** etc. A **restaurant** sell **items** on the platform. A **customer** has several **addresses**. A **courier** delivers packages of order to customers.

Answers of Problem 2：

Problem 3: Relational Formalization(12 points, 3 points per part)

For relation schema R(A, B, C, D, E) with functional dependencies set F={A→CD, C→B, B→D, B→E}

(1) Compute the Canonical Cover of F.

(2) Compute the Closure of attribute set {B}.

(3) Decompose the relation R into a collection of BCNF relations, and give out the functional dependency set for each relation.

(4) Explain whether above decomposition be dependency preserving or not.

Answers of Problem 3：

Problem 4: XML(12 points, 4 points per part)

Following DTD depicts information of the campus card database given in **problem 1**:

```
<!DOCTYPE     campus_cards[
    <!ELEMENT     campus_cards( pos+,card+)>
    <!ELEMENT     pos (campus, location)>
    <!ATTLIST     pos pno ID #REQUIRED>
    <!ELEMENT     campus (#PCDATA)>
    <!ELEMENT     location (#PCDATA)>
    <!ELEMENT     card (name, depart, balance, detail*)>
    <!ATTLIST     card cno ID #REQUIRED>
    <!ELEMENT     name (#PCDATA)>
    <!ELEMENT     depart (#PCDATA)>
    <!ELEMENT     balance (#PCDATA)>
    <!ELEMENT     detail (cdate, ctime, amount, remark)>
    <!ATTLIST     detail pno IDREF    #REQUIRED>
    <!ELEMENT     cdate (#PCDATA)>
    <!ELEMENT     ctime (#PCDATA)>
    <!ELEMENT     amount (#PCDATA)>
    <!ELEMENT     remark(#PCDATA)>
]>
```

(1) Base on the instance of the campus card database given in **problem 1**, give an XML document that conforms to above DTD. The document is required to contain the information of **pos "p003"** and **card "c0003"** as well as its consumption details.

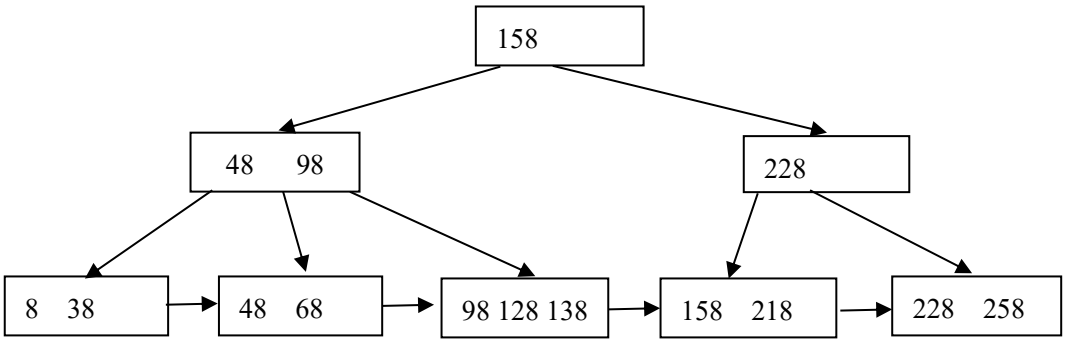
(2) Give a Path expression to find out the **location** of **pos** where 张帅 once paid 50.

(3) Give an XQuery expression to find out the **cno** of **card** that once paid at the same **pos** and **date** with 张帅.

Answers of Problem 4：

Problem 5: B+ -Tree (12 points, 4 points per part)

For the following B+-tree (n=4)：



(1) Draw the B+ -tree after inserting four entries 28,168, 58,18 sequentially.
(2) Draw the B+ -tree after deleting four entries 8, 38, 98,128 from the original B+-tree sequentially.
(3) Assuming (a) there are 3 blocks in main memory buffer for B+-tree operation, at first these blocks are empty. (b) Least Recently Used (LRU strategy) is used for buffer replacement. (c) each node of B+-tree occupies a block.
Please count the number of B+-tree blocks transferred to buffer in order to complete the operation in 1).

Answers of Problem 5：

Problem 6: Query Processing (16 points, 4 points per part)

For the relational schemas of the campus card database given in **problem 1**, there are following assumptions:

- n_{card}=10,000 , n_{pos}=100, n_{detail}=10,000,000
- l_{card}=25, l_{pos}=22, l_{detail}=29
- V(campus, pos) = 6, V(location, pos) = 20
- V(depart, card) = 100, V(name, card) = 5000
- The value of attribute **cdate** in **detail** table is uniformly distributed between '2017-01-01' and '2017-12-31'.
- block size is 4K bytes.
- size of B+-tree pointer is 4 bytes.
- **card** and **detail** tables are stored as sequential files based on search key **cno**.
- there is a B+-tree index on **detail(cno)**.

(1) Estimate the size (i.e. number of records) returned by following SQL statement :

```
select d1.cno, d2.cno
from detail d1, detail d2
where d1.pno=d2.pno and d1.cdate=d2.cdate and
      d1.cdate between '2017-05-01' and "2017-07-31'
```

(2) Estimate the number of blocks of **card** and **detail** tables respectively.

(3) Estimate the height of the B+-tree index on **detail(cno)**.

(4) Estimate the cost for evaluating expression "$\sigma_{name='张帅'}$**(card)** $\bowtie$ **detail** " using file scan for σ operation followed by indexed-loop join method for $\bowtie$ operation.

(Hind: the cost is measured by number of blocks transferred to main memory and times to seek disk.)

Answers of Problem 6：

Problem 7: Concurrency Control (12 points，3 points per part)

Consider following schedule S with five transactions T1,T2,T3,T4 and T5:

S: r1(A) w2(A) r2 (B) w3(A) w3(B) w4(C) w4(B) w5(C) w3(C)

where: ***ri(X)*** means transaction ***Ti*** read data ***X***.

wi(X) means transaction ***Ti*** write data ***X***.

(1) Draw the precedence graph of S.
(2) Explain whether S is serializable.
(3) Explain whether S be generated by the two–phase locking protocol.
(4) If S is serializable, give out all serial schedules to which S is equivalent. If S is not serializable, figure out an operation in S such that if this operation was removed from S，S would become serializable.

Answers of Problem 7：

Problem 8: Aries Recovery Method (10 points, 2 points per part)

A DBMS uses **Aries** algorithm for system recovery. Following figure is a log file just after system crashes. The log file consists of 16 log records with LSN from 1001 to 1016. The figure does not show PrevLSN and UndoNextLSN in log records. Assuming that last completed checkpoint is the log record with LSN 1012.

1001： <T1 begin>

1002： <T1 , 8001.1, 11, 22>

1003： <T2 begin>

1004： <T2 , 8001.2, 33, 44>

1005： <T3 begin>

1006： <T3, 8002.1, 55, 66 >

1008： <T1 commit>

1009： <T4 begin>

1010： <T4, 8001.1, 22, 77 >

1011： <T2, 8002.2, 88, 99>

1012： checkpoint

Txn	LastLSN
T2	1011
T3	1006
T4	1010

PageID	PageLSN	RecLSN
8001	1010	1010
8002	1011	1006

1013： <T2 commit>

1014： <T3, 8002.1, 66, 77>

1015： <T4, 8003.1, 11, 22>

1016： <T4 commit>

Please answer following questions：

(1) Which log record is the start point of Redo Pass?
(2) Which log record is the end point of Undo Pass?
(3) After Analysis Pass，what content is the dirty page table?
(4) After recovery, what is the value of data items identified by “8002.1” and “8002.2” respectively?
(5) After recovery, what additional log records appended to log fie?

Answers of Problem 8:

浙江大学 2017－2018 学年 春夏 学期

《数据库系统》课程期末考试试卷（A 卷）

参考答案及评分细则

Answers of Problem 1：
(12 points, 3 points per part)

（1）
```
select  c1.cno, c1.name
from    (card as c1) natural join (detail as d1),
          detail as d2
where c1.depart= 'CS' and d2.cno ='c0002' and
        d1.cdate = d2.cdate and d1.pno=d2.pno;
```

another answer:
```
select    c1.cno, c1.name
from      card as c1, detail as d1, detail as d2
where     c1.cno=d1.cno and
          c1.depart= 'CS' and d2.cno ='c0002'
          d1.cdate = d2.cdate and d1.pno=d2.pno;
```

评分细则：
每个 where 条件错扣 1 分，select 错扣 1 分，直至扣完

（2） $\Pi_{c1.cno,\ c1.name}\ (\sigma_{d1.cdate=d2.cdate\ \wedge\ d1.pno=d2.pno}$
$(\ (\sigma_{c1.depart='CS'}(\rho_{c1}(card))) \bowtie \sigma(\rho_{d1}(detail)) \times$
$(\sigma_{d2.cno='c0002'}(\rho_{d2}(detail)))\ )\ )$

评分细则：
每个操作错扣 1 分，直至扣完

（3）
```
select cno
from detail natural join pos
where year(detail.cdate)=2018
group by cno
having count(distinct campus)=1;
```

another answer:
```
select *
from card c1
where exists(
            select *
```

```
                from (detail natural join pos) as r1
                where r1.cno=c1.cno )
        and not exists(
                select *
                from    (detail natural join pos) as r1,
                        (detail natural join pos) as r2
                where   r1.cno=c1.cno and r2.cno=c1.cno and
                        year(r1.cdate)=2018 and year(r2.cdate)=2018
                        and r1.campus<> r2.campus
                )
```

评分细则：
每个子句错扣 1 分，直至扣完。

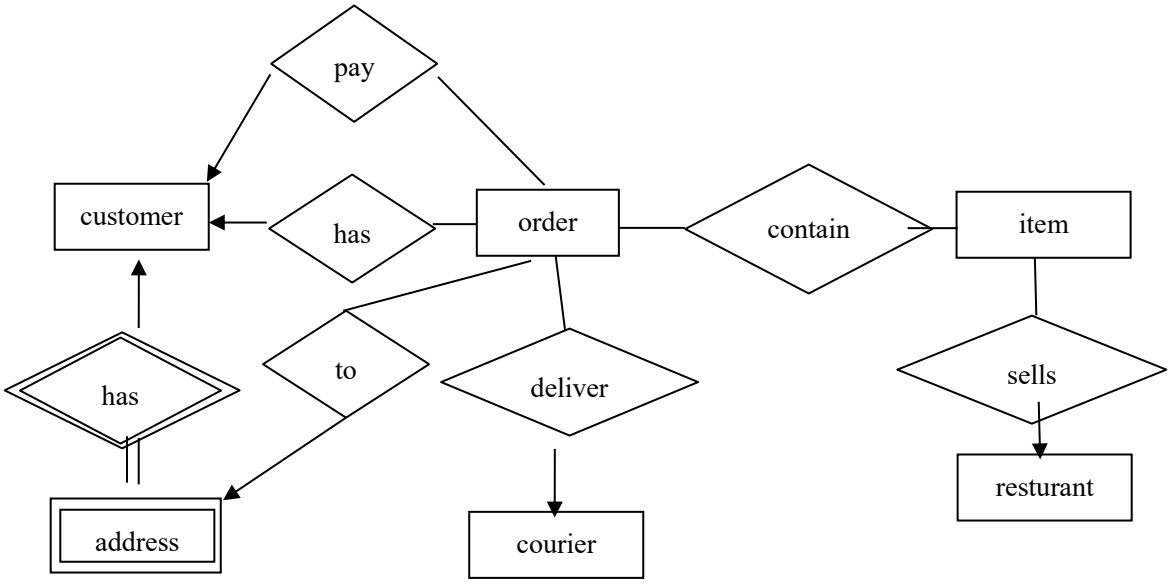
（4）
```
select cno
from detail natural join pos
where pos.campus='紫金港' and year(detail.cdate)=2018
group by pno
having sum(amount)>=all (
    select sum(amount)
    from detail natural join pos
    where pos.campus=' 紫金港' and year(detail.cdate)=2018
    group by pno
)
```

评分细则：
每个子句错扣 1 分，直至扣完

（5）
```
update card set balance = balance -20 where cno='c0002';
insert into detail(cno, pno, cdate, ctime, amount)
        values('c0002', 'p001','2018-07-02', '08:08:08', 20);
commit;
```

评分细则：
每条语句错扣 1 分。
最后一句 commit 可选。

Answers of Problem 2：
（11 points)



评分细则：
实体的属性不做要求。
基本 ER 图基础分 3 分。
每个实体或联系错扣 1 分。
无 order 扣 2 分，无 item 扣 1 分。

Answers of Problem 3：
(12 points, 3 points per part)

(1) Fc={A→C, C→B, B→DE}

(2) $(B)^+$ = (B,D,E)

(3) R1(B,D,E), F1={B→DE}
R2(C,B), F2={C→B}
R3(A,C), F3={A→C}

评分细则：
每个实体或联系错扣 1 分，直至扣完。
存在多种分解方法，需区分判题。

(4) The decomposition is dependency preserving,
because (F1∪F2∪F3)+ = F+
评分细则：
针对（3）中的不同分解方法，其函数保持的判断有不同，需区分判题。

Answers of Problem 4：
(12 points, 4 points per part)

（1）<campus_cards>
<pos pno=”p003”>
<campus> 玉泉 </campus>
<location> 四食堂 </location>
</pos>
<card cno=”c0003” >
<name> 王浩<name>
<depart> CS </depart>
<balance> 300<balance>
<detail pno=” p003”>
<cdate> 2018-07-03</cdate>
<ctime> 08:10:10 </ctime>
<amount>25 </amount>
<remark>餐饮<remark>
</detail>
</card>
</campus_cards>

评分细则：
每个 element 错扣 1 分，直至扣完

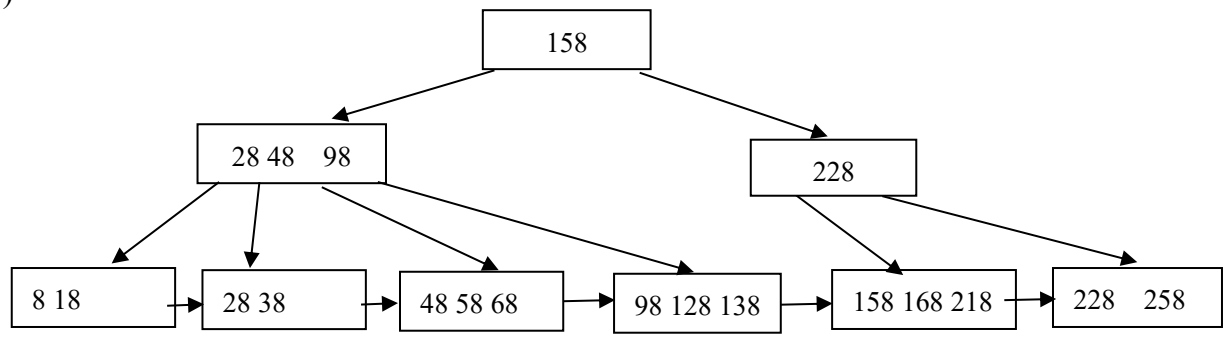
（2）
/campus_cards/card[name=”张帅”]/detail[amount=50] /id(pno) /location/text()
评分细则：
每个“/” 错扣 1 分，直至扣完
text()缺失不扣分

(3)
for $x in /campus_cards/card/detail
$y in /campus_cards/card[name=”张帅”]/detail
where $x/@pno=$y/@pno and $x/cdate=$y/cdate and
$x/@cno<>$y/@cno
return <cno> {$x/cno } </cno>
评分细则：
for 每个子句错扣 1 分
where 每个条件错扣 1 分
return 错扣 1 分
直至扣完

Answers of Problem 5:

(12 points, 4 points per part)

1)

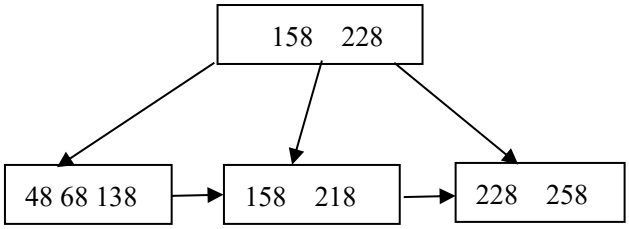


评分细则：

每个entry插入错扣1分

图中缺箭头或线扣1分

2)



评分细则：

每个entry删除错扣1分

图中缺箭头或线扣1分

3）3+2+2+1=8

评分细则：

答案为7、9之一，扣1分

答案为5、6、10之一，扣2分

答案为3、4、11之一，扣3分

其他扣4分

Answers of Problem 6:

(16 points, 4 points per part)

评分细则：每个运算结果错扣 1 分

1) (10000000*10000000) / (100 * 365)) *3/12 = 684.93M

   评分细则：

   三个条件：1/100,1/365, 3/12错各扣1分

   少一个10000000未乘扣1分

   按实际天数天数92/365代替3/12，也对。

2) Record number per block of card=4096/25=163
Blocks of card = 10000/163=61.3→62
Record number per block of detail=4096/29=141.24 →141
Blocks of detail = 10000000/141=70922
评分细则：
按不同的估算公式（要合理），答案略有差异，不扣分。

3) Fan-out rate n of the B+-tree =(4096-4)/(5+4) +1 = 455
4) Min height of B+tree = $\log_{455}$（10000） → 2（向上取整）
Max height of B+tree = $\log_{228}$（10000/2）+1= →2（向下取整）
So height of B+tree = 2
评分细则：
B+-tree高度为3，不扣分。

5) Cost for evaluating **σ operation （2分，各1分）**
block transfer = 62 t_T
seek time =1 t_S

cost for the natural join operation（2分，t_S 和 t_T各1分）
return number of σ name='张帅' (card) = (10000/5000)=2
block number for each card cno in detail = (10000000/10000)/141 = 7.09 =8

cost for the natural join operation = 2*(2 t_S +2 t_T +1 t_S +8 t_T)
=2*(3 t_S +10 t_T) = 6 t_S +20 t_T

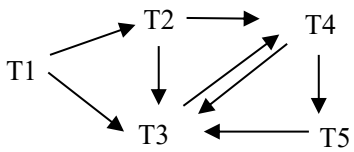
pipeline evaluation:
Total cost =（1 t_S +62 t_T）+（6 t_S +20 t_T） = 7 t_S +82 t_T

评分细则：
选择和连接操作的tS 和 tT各1分
若采用materialized evaluation方法，tS 多1， tT多2
若只给出一般公式，而未有具体数据带入，给1分。

Answers of Problem 7:

(12 points，3 points per part)

1) **评分细则：**
每个箭头错扣 0.5 分，直至扣完



2）
S is not serializable，because there are cycles in the graph ：T3~T4~T3；

T3~T4~T5~T3

评分细则：

结论 1 分，理由 2 分。

3）no，because every schedule generated by 2PL is serializable.

评分细则：

结论 1 分，理由 2 分。

4）w3(B) ，or w4(B)，，or w3(C)

评分细则：

以上3种答案都对，共3分。

如果（1）中图错引起（2）(3)错误，则（2）（3）如果理由正确各给1分

如果（2）（3）结论正确，理由不充分，则理由结论各给1分

Answers of Problem 8:

(10 points, 2 points per part)

1) 1006 （2分）
2) 1005 （2分）
3)

PageID	PageLSN	RecLSN
8001	1010	1010
8002	1014	1006
8003	1015	1015

评分细则：

少一个扣1分

4) “8002.1” = 55（1分）
“8002.2” = 99（1分）

5)

1017： <T3, 8002.1, 66>
1018： <T3, 8002.1, 55>
1019： <T3 abort>

评分细则：

少一个扣1分

浙江大学 2018–2019 学年 春夏 学期

《数据库系统》课程期末考试试卷

课程号：_____， 开课学院：_____

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式：√闭、开卷（请在选定项上打√）,允许带一张 A4 纸笔记入场

考试日期：2019 年 7 月 1 日,考试时间：120 分钟

诚信考试，沉着应考，杜绝违纪。

姓名：_____学号：_____所属院系：_____授课教师：_____

题序	一	二	三	四	五	六	七	八	总 分
得分									
评卷人									

Problem 1: Relational Model and SQL (16 points, 4 points each)

Consider the following relational schemas with the primary keys underlined.

Movie(title, type, director)

Comment(title, user_name, grade)

1) Write a *relational algebra expression* to find all the movie titles that are directed by "Yimou Zhang" and exist the comment grade of greater than or equal to 4.

2) Write a *SQL statement* to change the null value of grade to 0.

3) Write a *SQL statement* to find which movies have the highest average grade.

4) Write a SQL statement to find all the movie titles where every user gives higher grade than movie "the avenger".

Answers of Problem 1：

Problem 2: E-R Model (9 points)

We need to store information about football teams in league matches. The information includes matches (identified by match_id, with attributes location, time) as well as the score of each team for the match, teams (identified by name, with attribute city), players (identified by name, with attributes age and several phone numbers), and individual player statistics (such as score, shooting, foul, etc.) for each match. Note that one player only belongs to one team.

Please answer the following questions:

1) Draw an *E-R diagram* for this database model with primary key underlined. (5 points)

2) Transform the E-R diagram into a number of *relational database schemas*, with the primary key underlined. (4 points)

Answers of Problem 2：

Problem 3: Relational Formalization (12 points, 4 points each)

For relation schema R(A, B, C, D, E) with functional dependencies set F={A→B, BC→D, C→A}

1) Find all *candidate keys* of R.

2) Decompose the relation R into a collection of *BCNF relations*.

3) Explain whether above decomposition be *dependency preserving* or not.

Answers of Problem 3：

Problem 4: XML (12 points, 4 points each)

Following is an example XML document describing the information in **Problem 1**.

```
<movie_comment>
    <movie title=”wandering earth”>
        <type>science fiction</type>
        <director> Fan Guo </director>
        <comment>
            <user name>Alice</user name>
            <grade>5</grade>
        </comment>
        <comment>
            <user_name>Bob</user_name>
            <grade>4</grade>
        </comment>
    </movie>
    <movie title=”the avenger”>
        <type>action</type>
        <director> Joss Whedon </director>
        <comment>
            <user_name>John</user_name>
            <grade>4</grade>
        </comment>
    </movie>
</movie_comment>
```

1) Give the *DTD* for the XML representation, requiring that at least one comment for each movie.

2) Give a *Path expression* to find all the action movie titles where Alice gives grade 5.

3) Give an *XQuery expression* to find all the movie titles that are directed by “Yimou Zhang” and have at least one grade 5.

Answers of Problem 4：

Problem 5: B^+-Tree (12 points, 3 points each)

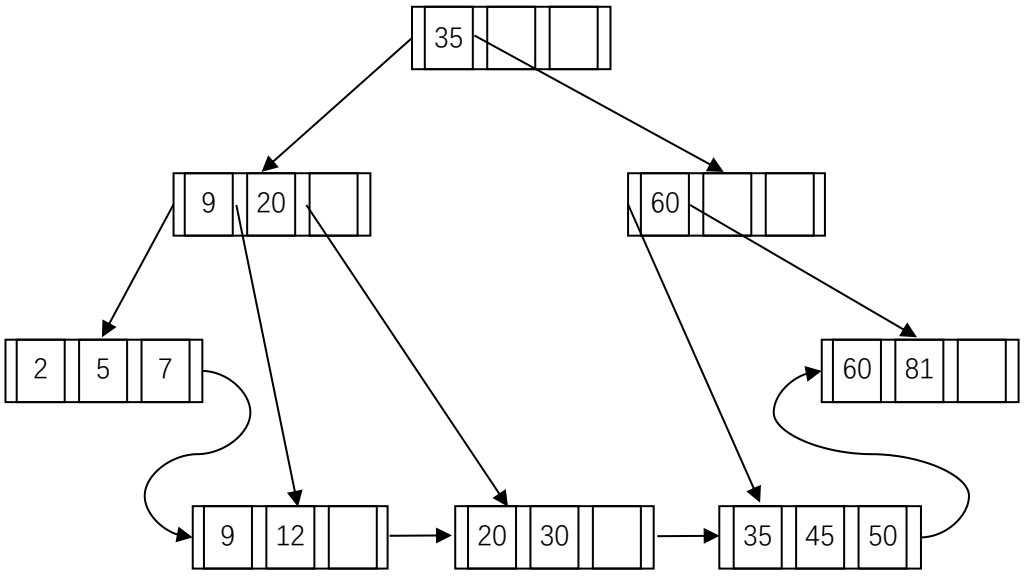
For the following B^+-tree (n = 4), answer following questions:

1) Show the structure of the tree after inserting sequentially 8, 6, and 3 into the original tree.

2) Show the structure of the tree after deleting sequentially 81 and 45 from the original tree.

3) If the height of tree is 5, please tell the minimal and maximal numbers of key values in the tree.

4) Assume that (a) there are 3 blocks in main memory buffer for B^+-tree operation, and at first those blocks are empty; (b) the Least Recently Used (LRU) strategy is used for buffer replacement; and (c) each node of B^+-tree occupies a block. Please count the number of B^+-tree blocks transferred to buffer in order to complete the operations in 2) of this problem.



Answers of Problem 5：

Problem 6: Query Processing (12 points, 4 points each)

For the relational schemas in problem 1, there are following assumptions:

Number of tuples, movie: 5,000, comment:1,000,000;

Blocking factor, movie:50, comment:100;

Number of distinct values, V(director, movie) = 500, V(grade, comment) = 5;

Block size is 4K bytes;

Movie has a B^+-tree index on title, and each index block contains 60 entries (i.e. n=60)

Answer following questions (*We do not consider cost to write output to disk*):

1) Estimate the size of the result of 1) in **Problem 1**.

2) Suppose that the *block nested-loop join* is used to implement movie $\bowtie$ comment, buffer in main memory has 12 blocks, and movie is chosen as inner relation. In order to obtain the best performance, how to assign buffer blocks to movie and comment respectively? Please estimate the number of block accesses and the number of seeks required by the solution.

3) Suppose that the *index nested-loop join* is used to implement $\sigma_{\text{director="Yimou Zhang"}}$ (movie) $\bowtie$ comment. Please estimate the number of block accesses and the number of seeks required by the solution (assume the worst case of memory, and that one extra buffer block is used for the root index block and it needs to be read from disk only once).

Answers of Problem 6：

Problem 7: Concurrency Control (12 points, 4 points each)

Consider following three transactions:

T1: read(A)
Read(B)
Write(B)

T2: read(B)
Read(A)
Write(A)

T3: read(A)
Write(A)

1) For the following schedule, please draw the precedence graph, and explain whether it is conflict serializable.

T1	T2	T3
Read(A)		
Read(B)		
		Read(A)
		Write(A)
	Read(B)	
	Read(A)	
Write(B)		
	Write(A)	

2) For the following schedule, please explain whether it is cascadeless.

T1	T2	T3
Read(A)		
Read(B)		
Write(B)		
	Read(B)	
	Read(A)	
	Write(A)	
		Read(A)
Abort		

3) Please explain whether the two-phase locking protocol can be used to implement the schedule in 1).

Answers of Problem 7：

Problem 8: Aries Recovery Method (15 points, 3 points each)

A DBMS uses Aries algorithm for system recovery. Following figure is a log file just after system crashes. The log file consists of 14 log records with LSN from 1001 to 1014. The figure does not show PrevLSN and UndoNextLSN in log records. Assume that last completed checkpoint is the log record with LSN 1008.

1001: <T1, begin>
1002: <T1, 101.1, 11, 21>
1003: <T2, begin>
1004: <T2, 102.1, 52, 62>
1005: <T2, commit>
1006: <T3 begin>
1007: <T3, 102.2, 73, 83>
1008: checkpoint

Tx	LastLSN
T1	1002
T3	1007

PageID	PageLSN	RecLSN
101	1002	1002
102	1007	1004

1009: <T1, 101.2, 31, 41>
1010: <T4 begin>
1011: <T3, 102.2, 73>
1012: <T3, abort>
1013: <T4, 102.1, 62, 64>
1014: <T1, commit>

Please answer following questions：
1) Which log record is the start point of Redo Pass?
2) Which log record is the end point of Undo Pass?
3) After Analysis Pass, what is the undo list?
4) After recovery, what is the value of data items identified by “102.1” and “102.2”, respectively?
5) What additional log records are appended to log file during recovery?

Answers of Problem 8：

浙江大学 2018－2019 学年 春夏 学期

《数据库系统》课程期末考试试卷（A 卷）

参考答案及评分细则

Answers of Problem 1：
(16 points, 4 points per part)

1) $\Pi_{Title}(\sigma_{director="Yimou\ Zhang"}(movie) \bowtie \sigma_{grade>=4}(comment))$

评分细则：
错一处扣 4 分

2) Update comment set grade=0 where grade is null

评分细则：
错一处扣 4 分，grade=null, grade is null 等类似答案均给分

3) Select type from movie, comment
Where movie.title=comment.title
Group by title
Having avg(grade) >=all (Select avg(grade)
From movie, comment
Where movie.title=comment.title
Group by title)

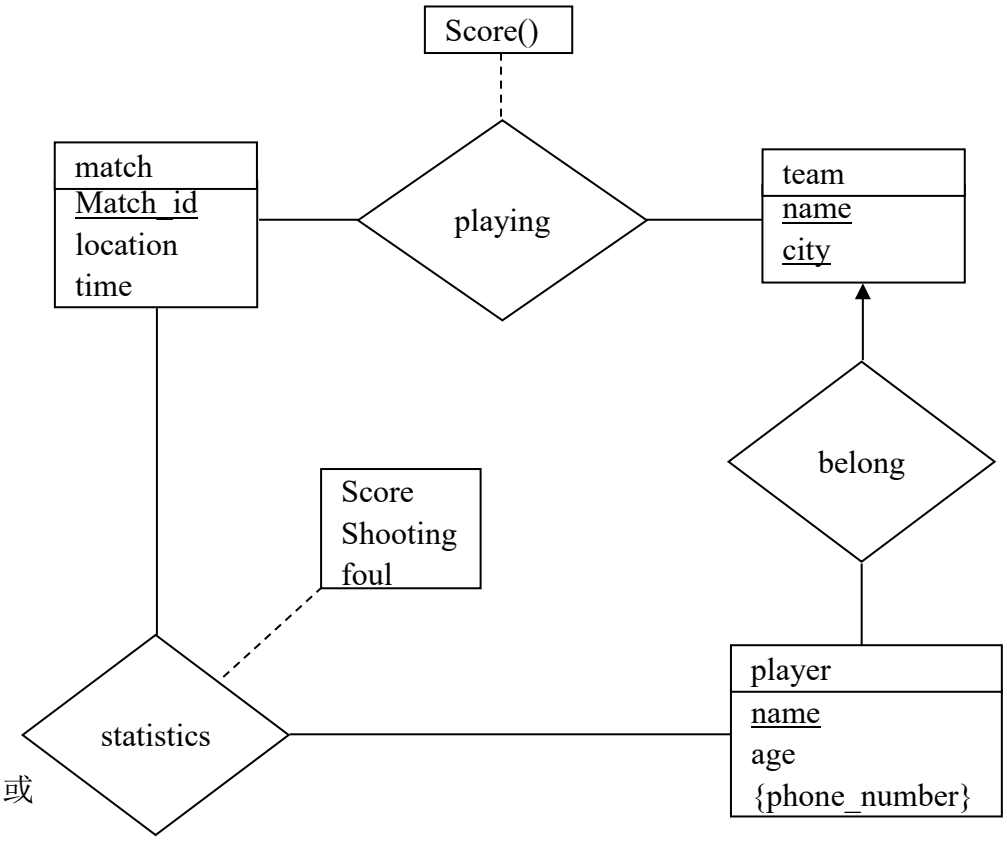
评分细则：
写出 having……且对均给 2 分，用其他 SQL 语句写出相同效果均给分

4) Select title from movie
Except
Select title from movie
Where exists (select *
From comment A, comment B
Where A.title=movie.title and A.user_name = B.user_name
And B.title=' the avenger'
And A.grade <=B.grade)

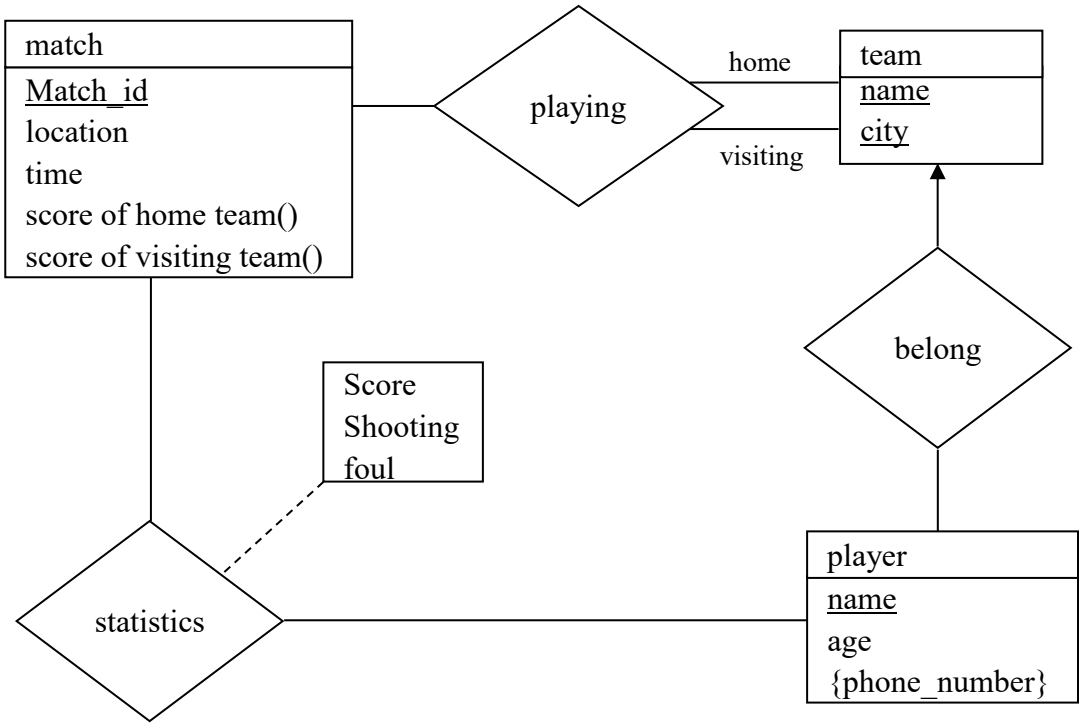
评分细则：
写出 4 个正确条件给 2 分，全对给 4 分

Problem 2: E-R Model (9 points)

1) (5 points)



或



评分细则：

联系写成实体扣 1 分，没写联系扣 1 分，属性写错漏写扣 1 分，少写实体或联系扣 1 分，扣完为止。

2) (4 points)

Match(match_id, location, time)

Team(name, city)

Playing(match_id, name, score)

Player(name, age)

Phone(player_name, phone_number)

Statistics(match_id, player_name, score, shooting, foul)

或者其中的 match 和 playing 改为：

Match(match_id, location, time, home_team_name, visiting_team_name, score_of_home_team, score_of_visiting_team)

Each match has one home team and one visiting team.

评分细则：

每个关系没有主键或者写错扣 1 分，扣完为止

Problem 3: Relational Formalization (12 points, 4 points each)

1) {C E}

评分细则：

写对一个键给两分，多写一个扣 1 分。诸如写{ACE,CE,BCE}的不得分

2) Decompose R into R1(A, B) and R2(A, C, D, E), decompose R2 into R21(A, C) R22(C, D, E), and further decompose R22 into R221(C, D) and R222(C, E)

评分细则：

由于根据不同模式分出来的步骤可能不一，但是由于最终关系为{C->A,C->B,C->D}，所以最终结果一定是诸如{A,C}{B,C}{C,D}{D,E}等二元组，根据实际情况没分彻底的如(B,C,D)每个 0.5 分，分彻底的 1 个 1 分，分错但是结果正确的酌情给 2-3 分。

3) The decomposition is dependency preserving.

评分细则：

(2)中答案正确并且此处正确的给 4 分，

(2)中答案错误并且根据实际拆分情况，若判断一致此处给 3 分

(2)中答案错误并且此处正确的给 2 分，（不给出解释扣 1 分）

其余情况不给分

Problem 4: XML (12 points, 4 points each)

1)

<!DOCTYPE movie_comment[

```
        <!ELEMENT    movie_comment ( movie*)>
        <!ELEMENT    movie (type, director, comment+)>
        <!ATTLIST          movie title ID #REQUIRED>
        <!ELEMENT    type (#PCDATA)>
        <!ELEMENT    director (#PCDATA)>
        <!ELEMENT    comment (user_name, grade)>
        <!ELEMENT    user_name (#PCDATA)>
        <!ELEMENT    grade (#PCDATA)>
    ]>
```

评分细则：

错 1-2 处扣 1 分，较多错误酌情扣 2-3 分

2) /movie_comment/movie[type="action" and ./comment/user_name="Alice" and ./comment/grade=5]/@title

评分细则：

漏一个条件扣 1 分，路径错误扣 1 分

3) for $p in /movie_comment/movie[director="Yimou Zhang"]
where count($p/comment[grade=5])>=1
return $p/@title

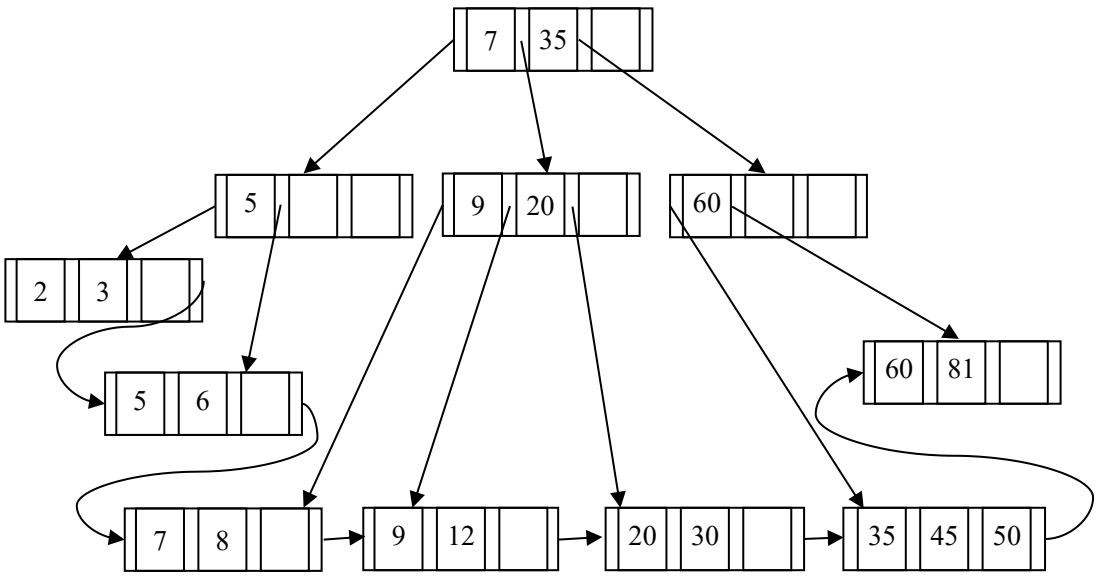
评分细则：

少一个条件扣一分，逻辑错误扣 2-3 分

Problem 5: B^+-Tree (12 points, 3 points each)

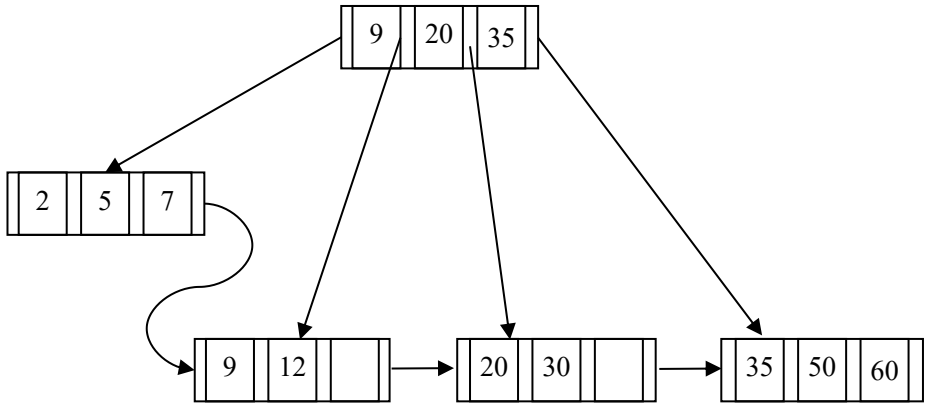
1)

After inserting 8, 6 and 3:



评分细则：
每个 value 错扣 1 分，最多扣 2 分
叶子正确最少得 1 分

2)
After deleting 81 and 45:



评分细则：
索引 merge 错误扣 1 分，插入叶子错误扣 1 分
叶子正确最少得 1 分

3) Maximal number of key values: 4*4*4*4*3=768
Minimal number of key values: 2*2*2*2*2=32

评分细则：
公式列正确即给分

4) (3 + 1) + 1 = 5 或 (3 + 1) + 2 = 6

评分细则：
错，扣 3 分

Problem 6: Query Processing (12 points, 4 points each)

1) 5,000/500/5 = 2

评分细则：
2,4 均可，没有计算扣 2 分
400 扣 1 分，10 扣 1 分

2) Number of blocks of movie is 5000/50=100

Number of blocks of comment is 1,000,000/100=10,000

Since the equi-join attribute title forms a key on inner relation, we can stop inner loop on the first match.

Assign 10 blocks to comments, 1 block to movies, and 1 block for output.

Number of block accesses: (10000/10)*100+10000 = 110000 或

10000 * 100/10 + 100 = 100100

Number of seeks: 2*10000/10=2000

评分细则：

Movie，comment 各 1 分

Block 110000,100100 均给 1 分

Seeks 2000,20 均给分

有公式答案错酌情扣分

3) Minimal height = $\log_{60}(5000)$ → 3（向上取整）

Max height = $\log_{30}(5000)$ →3（向上取整）

So, the height of the B^+-tree index on movie(title) is 3.

Number of block accesses: 10000+1000000/500*3+1

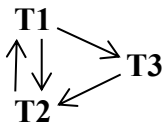
Number of seeks: 10000+1000000/500*3+1

评分细则：

不是白卷且答案合理均给分

Problem 7: Concurrency Control (12 points, 4 points each)

1)



The schedule is not serializable, because there are cycles in the graph.

评分细则：

少一个依赖扣 1 分，

如前趋图错，若冲突串行化与画出图一致，也给全分

2) The schedule is not cascadeless.

评分细则：

结论错，论述正确得 2 分

结论对，论述错得 3 分

结论对，论述正确得 4 分

其余不给分

3) No. This is because the schedule in 1) exists cycles.

评分细则：
结论错，论述正确得 2 分
结论对，论述错，酌情得 2-3 分
结论对，论述正确得 4 分
其余不给分

Problem 8: Aries Recovery Method (15 points, 3 points each)

1) 1002

评分细则：
多答扣 1-3 分

2) 1010

评分细则：
多答扣 1-3 分

3) T4

评分细则：
（T4,1013）也给分，其余不给分

4) “102.1” = 62, “102.2” = 73

评分细则：
错一个扣 1 分，错 2 个扣完
多一个扣 1 分，多 2 个不给分

5)

1015: <T4, 102.1, 62>
1016: <T4, abort>
评分细则：
见（4）

浙江大学 2019－2020 学年 春夏 学期

《数据库系统》课程期末考试试卷

课程号：21121350，　开课学院：计算机学院

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式：√闭、开卷（请在选定项上打√），允许带一张 A4 纸笔记入场

考试日期：2020 年 9 月 5 日，考试时间：120 分钟

诚信考试，沉着应考，杜绝违纪。

考生姓名：_____学号：_____所属院系：_____

题序	一	二	三	四	五	六	七	八	总 分
得分									
评卷人									

Problem 1：Relational Model and SQL（18 points)

A software development company develops software projects for different clients. It has the following relational schemas for its internal management system:

client (cId, cName, cCity)
project (pId, pName, cId, startTime, endTime, budget, paid)
employee (eId, eName, eAddress, eSalary, eBonus)
participate (pId, eId, role)

The underlined are primary keys. “cId” in “project”, “pId” and “eId” in “participate” are foreign keys. “paid” is the cost that the client has paid for the project. One employee can participate in different projects with a specific role for each project. Only three different roles are permitted: “project manager”, “developer”, and “tester”. Based on the above schemas, answer the following questions:

(1) Write a relational algebra expression to find the project names that the client “X Bank” has, where “X Bank” is the client name. (3 points)

(2) Write a SQL statement to create table participate with all the necessary constraints. (5 points)

(3) Write a SQL statement to find the employee names who participate in different projects with all the three different roles. (3 points)

(4) Write a SQL statement to find the employee names who have the maximum salary in project “p1102”, where “p1102” is the project id. (3 points)

(5) If there are more than three participating roles and the software company requires to maintain a permitted role list in the application, what will be your suggestion to modify the schemas? (4 points)

Answers of Problem 1：

Problem 2：E-R Model （11 points)

Suppose you want to design a simple video sharing web site. **Users** can upload and watch **videos**. There are **channels** for videos. A video may belong to several channels. Users can subscribe to different channels. Channels have hierarchy structure, e.g., “chicken eating” channel is the sub channel of “game” channel. For better user recommendation, one channel may have relationships with any other channels, e.g., “travel” channel has relationship with “photography” channel, although “travel” is not a super or sub channel of “photography”.

Please draw an E-R diagram for the database design of the web site.

Answers of Problem 2：

Problem 3: Relational Formalization (12 points)

For relation schema R (A,B,C,D,H,I) with functional dependencies set F = {B->C, AC->D, BC->H}. Answer the following questions:
1) Find all the candidate keys; (3 points)
2) Find the canonical cover Fc; (3 points)
3) If R is not in BCNF, decompose it into BCNF schemas. (4 points) Is this decomposition dependency preserving? (2 points)

Answers of Problem 3：

Problem 4: XML(9 points, 3 points per part)

The following is a simplified DTD for the software company in **problem 1**:

```
<!DOCTYPE     software_company[
    <!ELEMENT     software_company( client*,employee+)>
    <!ELEMENT     client (cname, ccity, project*)>
    <!ATTLIST     client cid ID #REQUIRED>
    <!ELEMENT     project (pname,paid)>
    <!ATTLIST     project pid ID #REQUIRED>
    <!ELEMENT     employee (ename, esalary, participate*)>
    <!ATTLIST     employee eid ID #REQUIRED>
    <!ELEMENT     participate (role)>
    <!ATTLIST     participate pid IDREF #REQUIRED>
    <!ELEMENT     cname (#PCDATA)>
    <!ELEMENT     ccity (#PCDATA)>
    <!ELEMENT     pname (#PCDATA)>
    <!ELEMENT     paid (#PCDATA)>
    <!ELEMENT     ename (#PCDATA)>
    <!ELEMENT     esalary(#PCDATA)>
    <!ELEMENT     role(#PCDATA)>
]>
```

Please answer the following questions:

(1) Give an XPath expression to find the project names that the client “X Bank” has paid for more than 50000 Yuans each. “X Bank” is a client name.

(2) Give an XPath expression to find the project names that employee “John” participates in as project manager. “John” is an employee name.

(3) Give an XQuery expression to find the project names from clients in “Shanghai” and employee “John” participates in. “Shanghai” is a city. “John” is an employee name.

Answers of Problem 4：

Problem 5: B+ -Tree (10 points)

Table employee in **Problem 1** is stored sequentially on eId. To build a B+-tree for table employee on column eName, buckets are used between index entries in leaf node and data records in data file. One bucket is a collection of pointers that point to data records with same employee name. One bucket is stored in one block except it exceeds the block size. If there are 13 employees initially as follows：

Wu, Mozart, Einstein, Mark, Susan, Gold, Katz, Singh, Crick, Mark, Brandt, Kim, Gold

(1) Suppose the max number of pointers in a B+-tree node is 4 (n = 4), draw a B+-tree with buckets based on the above employee names. (5 points)
(2) Suppose there are 20000 employees, within which there are 18000 distinct employee names. B+-tree node size is 4096 (n will be far bigger than 4). Employee name length is 32, pointer length in B+-tree node is 4. There are 5 employees with the same name “Mark”. To find all the information of employees named “Mark” using B+-tree with buckets, in the worst case, how many blocks will be read? (5 points)

Answers of Problem 5：

Problem 6: Query Processing (15 points)

For the relational schemas in **problem 1**, there are following information:

- Record numbers: n_{client}=500 , $n_{project}$=2000, $n_{employee}$=30000, $n_{participate}$=40000
- Records in a block: f_{client}=80, $f_{project}$=50, $f_{employee}$=40, $f_{participate}$=100
- Distinct values: V(cCity, client)=100, V(cId, project)=400, V(eId, participate)=28000, V(pId, participate)=2000
- block size is 4K bytes.
- Numbers of clients are assumed to be the same for different cities.

(1) Estimate the size (i.e. number of records) returned by the following SQL statements:
   a. **select** cId **from** client **where** cCity = "Shanghai"; (3 points)
   b. **select** e.eId, e.eName, pro.pId, pro.pName, par.role
      from client as c, project as pro, employee as e, participate as par
      where c.cId = pro.cId and pro.pId = par.pId and e.eId = par.eId and c.cCity="Shanghai"; (3 points)

(2) Estimate the number of blocks for employee and participate respectively. (4 points)

(3) Suppose there are 50 buffer blocks to use, estimate the cost of "employee natural join participate" using hash join. (5 points)

Answers of Problem 6：

Problem 7: Concurrency Control (11 points)

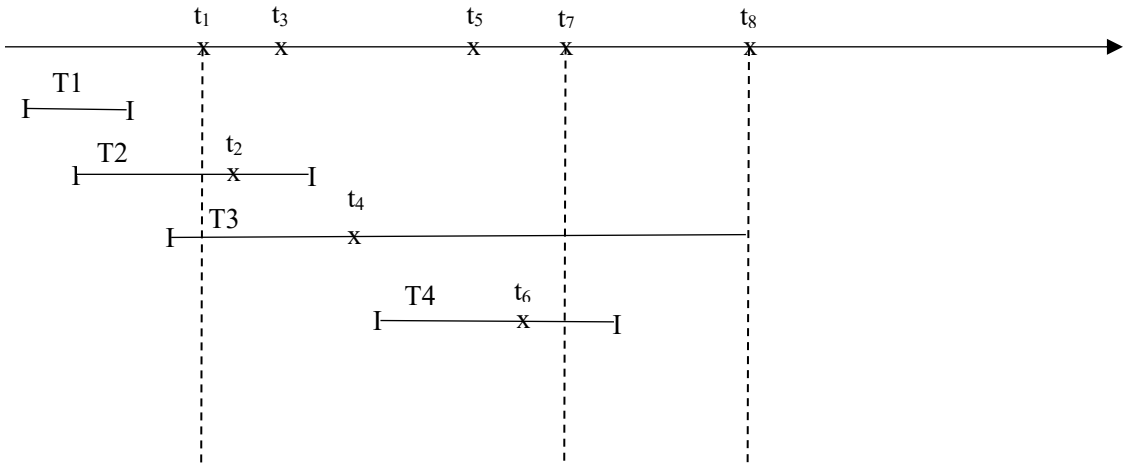
For a database with problem 1, suppose there are three transactions executing concurrently:
T1: select sum(eSalary) from employee;
T2: update employee set eSalary = eSalary + 800 where eId = “190012”;
T3: update employee set eSalary = eSalary + 500 where eId = “190012”;
update employee set eSalary = eSalary + 1000 where eId = “181020”;
The total salary of all the employees is 250000 Yuans before the execution of these three transactions. The database management system has implemented strict two-phase locking for concurrent control with record level locks only. Please answer the following questions:

(1) If all the transactions commit successfully, what are the possible outputs for T1? (4 points)
(2) Is there any possibility of deadlock? If there is, can you illustrate one transaction schedule that will cause the deadlock. (5 points)
(3) T4 inserts some new employees into table employee. If T4 executes concurrently with T1, T2 and T3, can the above-mentioned DBMS ensure serializable schedule? (2 points)

Answers of Problem 7：

Problem 8: Recovery (14 points)

A fuzzy checkpoint finds a list of modified buffer blocks (LBlocks) and a list of active transactions (LTransactions); writes a checkpoint log record with LTransactions in it; works with other transactions concurrently to output buffer blocks in LBlocks to the disk; and saves the pointer of the checkpoint log record as last-checkpoint information to a safe place on disk. Below is the timeline of a fuzzy checkpoint and related transactions. The arrow line is the time line. t_1 to t_8 are points of time ($t_1<t_2\ldots<t_8$). T1 to T4 are transactions. "I" represents the beginning or end of a transaction. Commit of a transaction does not mean that all the data the transaction modified should be output to the disk.



Here is what happened at each of the time point:
t_1: Fuzzy checkpoint began. There were two modified buffer blocks at that time: B1 and B2; t_2: T2 updated data in B1; t_3: Checkpoint output B1 to the disk; t_4: T3 updated data in B3; t_5: Checkpoint output B2 to the disk; t_6: T4 updated data in B2; t_7: Fuzzy checkpoint finished; t_8: System failure.

Suppose there were no other DBMS components that output log buffer or buffer blocks to the disk in the meantime. Please answer the following questions:

(1) What did the fuzzy checkpoint do to log buffer, disk log file, buffer blocks, and disk data blocks at times of t_1, t_3 and t_7 respectively? (4 points)
(2) What did T2 do to log buffer, disk log file, buffer blocks, and disk data blocks at time of t_2? (2 points)
(3) Which transactions are to be redone and which are to be undone during recovery? (4 points)
(4) Please mark on the diagram the beginning point of redo with "@" and the end point of undo with "#" during recovery. (4 points)

Answers of Problem 8:

Answers of Problem 1：

(1) $\Pi_{pname}(project \bowtie (\sigma_{cname="X\ Bank"}(client)))$

(2)
```
Create table participate (
        eId char(12),
        pId char(12),
        role char(24),
        primary key (eId,pId),
        foreign key (eId) references employee,
        foreign key (pId) references project,
        check role in ("project manager", "developer", "tester"))
```

(3) select eName from employee where eId in (select eId from participate group by eId having count(distinct role) = 3)

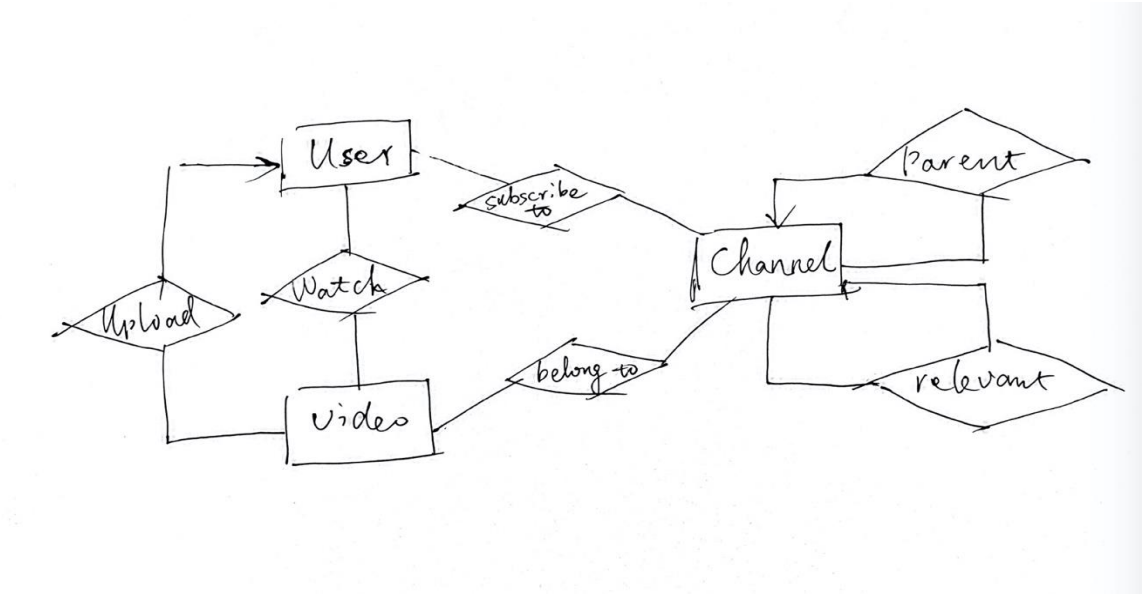
(4) select eName from employee natural join participate where pId = "p1102" and eSalary in (select max(salary) from employee natural join participate where pId = "p1102")

or

with pemployee(eId, eName, salary) as (select eId,eName, salary from employee where eId in (select eId from participate where pId = "p1102"))
select eName from pemployee where salary in (select max(salary) from pemployee)

(5) participate (<u>eId, pId</u>, roleId)
role (<u>roleId</u>, roleName)

Answers of Problem 2：



Answers of Problem 3：

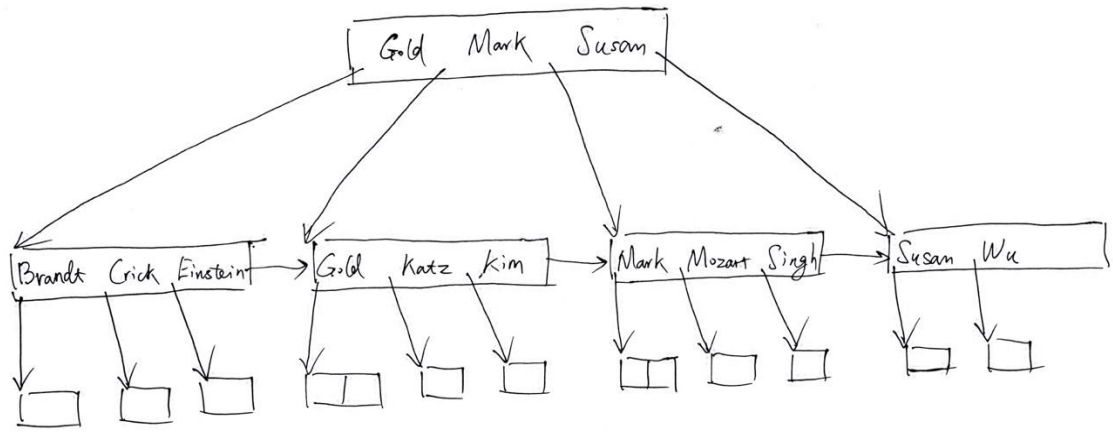
(1) ABI
(2) Fc = {B->CH, AC->D}
(3) (ACD, BCH, ABI) : dependency preserving; or (BCH, ABD, ABI): not dependency preserving

Answers of Problem 4：

(1) /client[cname = “X Bank”]/project[paid>50000]/pname
(2) /employee[ename=”John”]/participate[role = “project manager”]/id(@pid)/pname
(3) for $x in /employee[ename=”John”]/participate,
$y in /client[ccity=”shanghai”]/project
where $x/@pid = $y/@pid
return {$y/pname}

Answers of Problem 5：

(1)



---

(2) max entries per node : $\lfloor (4096-4)/(32+4) \rfloor = 113$, n = 114; B+-tree max height = $\lfloor \log_{57} 9000 + 1 \rfloor = 3$; worst case blocks to read: 3(B+-tree blocks)+1(bucket block)+5(one block for each employee) = 9

Answers of Problem 6：

(1) a. Shanghai client number: 500/100 = 5
b. estimated size = 5*40000/ V(cId, project) = 500
(2) blocks of employee: 30000/40 = 750; blocks of participate: 40000/100 = 400
(3) most optimistic: M = 50， n = 400/50 = 8， $b_b = \lfloor 50 / (8+1) \rfloor = \lfloor 50/ 9 \rfloor = 5$
Transfers: $3(b_r + b_s) + 4n = 3*(1150)+32=3450+32$
Seeks: $2(\lceil b_r / b_b \rceil + \lceil b_s / b_b \rceil) + 2n = 2*(\lceil 750/ 5 \rceil + \lceil 400/ 5 \rceil) + 16 = 460+16$

or

considering fudge factor = 1.2:
M = 50， n = ⌈ 8*1.2⌉= 10， $b_b = \lfloor 50 / (10+1) \rfloor = \lfloor 50/ 11 \rfloor = 4$
……

Answers of Problem 7：

(1) normal: 250000; 250000+800; 250000+1500; 250000+2300; extra: 250000+1000 (there is no “190012”) ; 250000+500; 250000+1300 (there is no “181020”).
(2) Yes. T1 read “181020”; T3 write “190012”; T1 read “190012”; T3 write “181020”
(3) No

Answers of Problem 8:

(1) t1: write <checkpoint LTransactions> to log buffer; output log buffer to log file;
t3: output log buffer to log file; output B1 to disk;
t7: do nothing for log or data. (save pointer of <checkpoint L> to a save place)
(2) t2: write a data modification log to log buffer; modify B1; do nothing to others
(3) redo: T2, T4; undo: T3
(4) redo start: t1; undo end: beginning of T3

浙江大学 2020－2021 学年 春夏 学期

《数据库系统》课程期末考试试卷参考答案和评分标准

课程号：21121350，　开课学院：计算机学院

考试试卷：√A 卷、B 卷（请在选定项上打√）

考试形式：√闭、开卷（请在选定项上打√），允许带一张 A4 纸笔记入场

考试日期：2021 年 7 月 2 日，考试时间：120 分钟

诚信考试，沉着应考，杜绝违纪。

考生姓名：_____学号：_____所属院系：_____

题序	一	二	三	四	五	六	七	八	总 分
得分									
评卷人									

Problem 1: Relational Model and SQL (18 points)

Following are the relational schemas of a SRTP (Student Research Training Program) project database.

student (sId, sName, dId)
teacher (tId, tName, dId)
department (dId, dName)
project (pId, pName, tId, startTime, endTime)
participate (pId, sId, role)

The underlined attributes are primary keys, and foreign keys are listed as follows:
"dId" in "student" references "department";
"dId" in "teacher" references "department";
"tId" in "project" references "teacher";
"pId" and "sId" in "participate" reference "project" and "student", respectively.
In "participate", only two different roles are permitted: "leader" and "member". Based on the above relational schemas, please answer the following questions:

(1) Write a relational algebra expression to find the names of the projects that are instructed by a teacher from the department "Computer Science". (4 points)
(2) Write SQL statements to create tables project and participate with all the necessary

constraints (Note: Tables student, teacher, and department have already been created and can be referenced). (6 points)

(3) Write a SQL statement to find the names of the teachers that instruct at least one project started in the year 2020. (4 points)

(4) Write a SQL statement to find the names of the students participating more than 2 projects. (4 points)

Answers of Problem 1:

(1)

$\prod_{pName}(project \bowtie teacher \bowtie (\sigma_{dName=\text{“Computer Science”}}(department))$

评分细则：每个操作错扣 1 分，全对给 4 分

(2)

```
CREATE TABLE project
    (pId    char(10),
    pName    varchar(20),
    tId    char(10),
    startTime    date,
    endTime    date,
    primary key (pId),
    foreign key (tId) references teacher);
```

评分细则：写出 schema、primary、foreign key 给 2 分，类型全对给 3 分

```
CREATE TABLE participate
    (pId    char(10),
    sId    char(10),
    role    varchar(20),
    primary key (pId, sId),
    foreign key (pId) references project,
    foreign key (sId) references student,
    check (role=“leader” or role=“member”));
```

评分细则：写出 primary、foreign key、check 给 2 分，全对给 3 分

(3)

```
select distinct tName
from project, teacher
where project.tId=teacher.tId and startTime between '2020-01-01' and '2020-12-31'
```

评分细则：每个 where 条件错扣 1 分，全对给 4 分

(4)

```
select sName
from student
```

where sId in
(select sId
from participate
group by sId
having count(pId) > 2)

评分细则：写出 having…且对给 1 分，内表写对给 2 分，全对给 4 分

Problem 2: E-R Model (11 points)

Based on the SRTP project management scenario in Problem 1, some new requirements are added as follows:

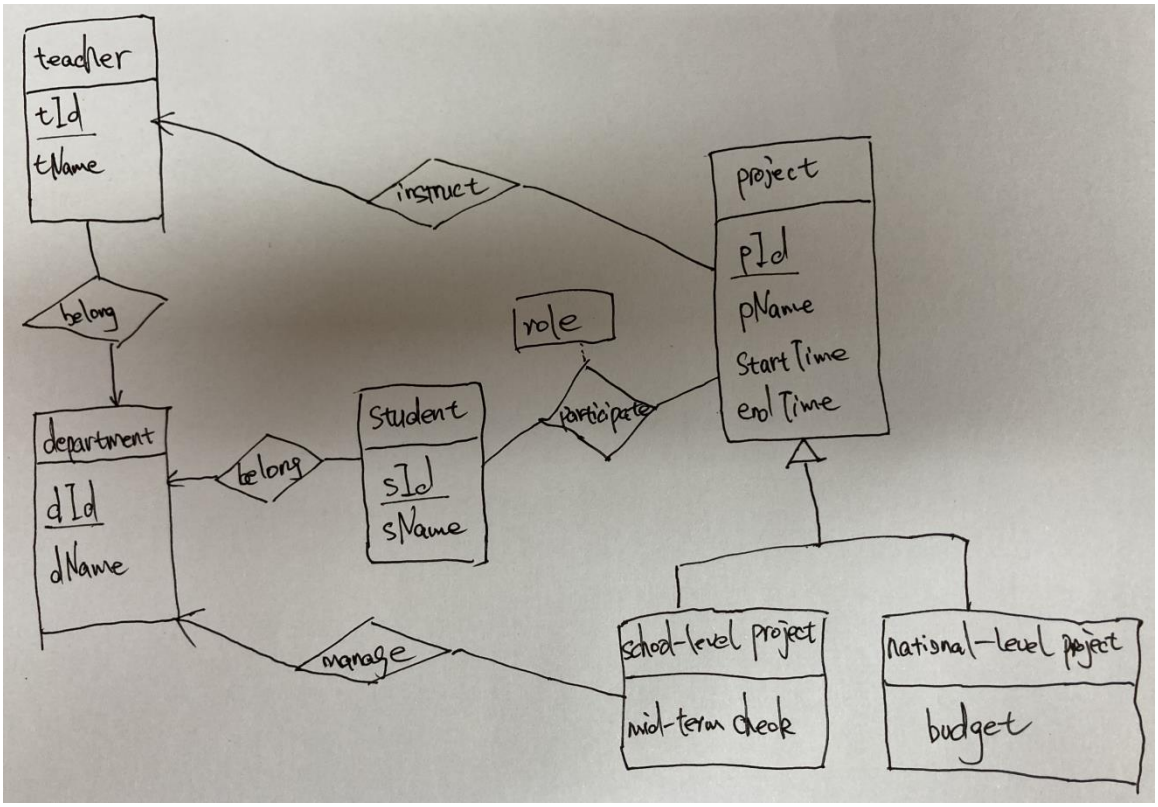
(1) There are two kinds of SRTP projects, i.e., school-level projects and national-level projects, and a project is either school-level or national-level.

(2) National-level projects have budget information, and school-level projects have mid-term check information.

(3) A school-level project is associated with exactly a department that is in charge of the management of the project.

Please draw an E-R diagram for the scenario.

Answers of Problem 2:



评分细则：联系写成实体扣 1 分，没写联系扣 1 分，属性写错漏写扣 1 分，少写实体或联系扣 1 分，全对给 11 分

Problem 3: Relational Formalization (12 points)

For relation schema R (A, B, C, D, E, F) with functional dependencies set F = {A->B, A->C, B->C, D->E, D->F, EF->D}. Answer the following questions:

(1) Find all the candidate keys. (3 points)

(2) Find the canonical cover Fc. (3 points)

(3) If R is not in BCNF, decompose it into BCNF schemas. (4 points) Is this decomposition dependency preserving? (2 points)

Answers of Problem 3:

(1)
AD AEF
评分细则：少写一个扣 1 分，多写扣 1 分，全对给 3 分

(2)
A->B, B->C, D->EF, EF->D
评分细则：少写一个扣 1 分，全对给 3 分

(3)
There are different decomposition results and the following is just an example.
R1=(A, B), R2=(A, C, D, E, F) (A->B)
R21=(A, C), R22=(A, D, E, F) (A->C)
R221=(D, E), R222(A, D, F) (D->E)
R2221(D, F), R2222(A, D) (D->F)
This decomposition is not dependency preserving (e.g., B->C is not preserved).

Following is another solution:
R1=(B, C), R2=(A, B, D, E, F) (B->C)
R21=(A, B), R22=(A, D, E, F) (A->B)
R221=(D, E, F), R222(A, D) (D->EF)
This decomposition is dependency preserving, because A->B can be checked on R21, B->C can be checked on R1, D->EF and EF->D can be checked on R221.
评分细则：分解部分少写一个扣 1 分，全对给 4 分，依赖保持判断错误扣 1 分，原因错误扣 1 分，全对给 2 分

Problem 4: XML (8 points)

The following is a simplified DTD for the SRTP project database given in Problem 1:

```
<!DOCTYPE SRTP[
    <!ELEMENT SRTP(department+, teacher+, student+, project*)>
    <!ELEMENT department (dname)>
```

```
    <!ATTLIST       department dId ID #REQUIRED>
    <!ELEMENT       teacher (tname)>
    <!ATTLIST       teacher
                    tId ID #REQUIRED
                    dId IDREF #REQUIRED>
    <!ELEMENT       student (sname)>
    <!ATTLIST       student
                    sId ID #REQUIRED
                    dId IDREF #REQUIRED>
    <!ELEMENT       project (pname, starttime, endtime)>
    <!ATTLIST       project
                    pId ID #REQUIRED
                    tId IDREF #REQUIRED
                    sIds IDREFS #REQUIRED >
    <!ELEMENT       dname (#PCDATA)>
    <!ELEMENT       tname (#PCDATA)>
    <!ELEMENT       sname (#PCDATA)>
    <!ELEMENT       pname(#PCDATA)>
    <!ELEMENT       starttime(#PCDATA)>
    <!ELEMENT       endtime(#PCDATA)>
]>
```

Please answer the following questions:

(1) Give an XPath expression to return the names of all the teachers who supervise SRTP projects. (4 points)

(2) Give an XQuery expression to return all the projects and their corresponding instructors, in the form of project_instructor elements that have a project subelement and a teacher subelement. (4 points)

Answers of Problem 4:

(1)

```
/SRTP/project/id(@tId)/tname/text()
```

评分细则：每个路径错误扣 1 分，全对给 4 分

(2)

```
for  $p in /SRTP/project,
     $t in /SRTP/teacher,
where $p/@tId= $t/@tId
return <project_instructor> { $p $t } </project_instructor>
```

评分细则：漏一个条件扣 1 分，逻辑错误扣 2 分，全对给 4 分

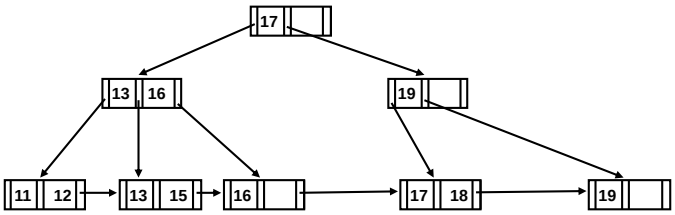
Problem 5: B+ -Tree and Query Processing (10 points)

Table student in Problem 1 is stored sequentially on sId. The following B+-tree is built for the table on attribute dId. Please answer the following questions:

(1) Is the built index a primary index? Why? (2 points)

(2) Draw the B+-tree after inserting entry 14. (4 points)

(3) Draw the B+-tree after deleting entry 19 from the original B+-tree. (4 points)



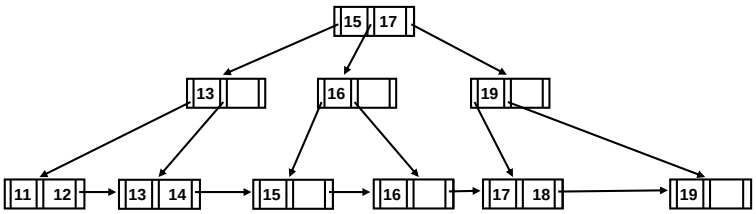
Answers of Problem 5:

(1)

The built index is not a primary index, as the search key of the index is not the search key of the sequentially ordered data file.

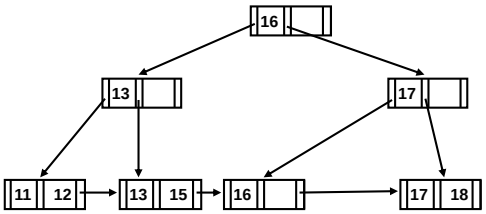
评分细则：结论正确给 1 分，解释正确给 1 分，结论错误不给分

(2)



评分细则：每个 entry 插入错扣 1 分，图中缺箭头或线扣 1 分，全对给 4 分

(3)



评分细则：每个entry删除错扣1分，图中缺箭头或线扣1分，全对给4分

Problem 6: Query Processing (14 points)

There are two relations *r* (100 blocks) and *s* (20 blocks), and hash-join algorithm is used to perform natural join between these two relations (memory size M=6 blocks). Please answer the following questions:

(1) How many partitions can be constructed? Why? (3 points)
(2) Which relation is best to choose as the build relation? Why? (3 points)
(3) Is recursive partition needed? Why? (3 points)
(4) Please compute the cost (numbers of seeks and block transfers) of the hash-join. (5 points)

Answers of Problem 6:

(1)
5 partitions, as the number of partitions is M-1.
评分细则：结论正确给 1 分，解释正确给 2 分，结论错误酌情给分

(2)
Relation *s*, as relation *s* is smaller than relation *r*.
评分细则：结论正确给 1 分，解释正确给 2 分，结论错误酌情给分

(3)
Recursive partition is not needed, as the size of the partitions of relation *s* (i.e., 4) is less than or equal to M-2 (i.e., 4).
评分细则：结论正确给 1 分，解释正确给 2 分，结论错误酌情给分

(4)
Number of block transfers: $3\times(100+20)+4\times5$
Note: 4×5 is not necessary, which considers partially filled blocks.
Number of seeks: $2\times(100+20)+2\times5$
评分细则：block transfers 的数量 3 分，seeks 的数量 2 分，全对给 5 分

Problem 7: Concurrency Control (13 points)

Given the following schedule, please answer the following questions:

(1) Draw the precedence graph for the schedule. (3 points)

(2) Is the schedule conflict serializable? Why? (2 points)

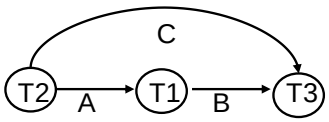
(3) Is it possible that the schedule is generated by the 2PL protocol with lock conversions? Explain. (5 points)

(4) Which conditions should be satisfied if we want the schedule to be recoverable? (3 points)

T1	T2	T3
	read C	
read B		
	write C	
	read A	
		read C
	write A	
read A		
		write C
write B		
		read B

Answers of Problem 7:

(1)



评分细则：图中结点错误或连接错误，每错一处扣 1 分，全对给 3 分

(2)

The schedule is conflict serializable, as the precedence graph is acyclic.

评分细则：结论正确给 1 分，解释正确给 1 分，结论错误不给分

(3)

It is possible that the schedule is generated by the 2PL protocol with lock conversions.

T1	T2	T3
	Lock-S (C) read C	
Lock-S (B) read B		
	Upgrade (C) write C	
	Lock-X (A) read A	
	UL (C)	
		Lock-S (C) read C
	write A	
	UL (A)	
Lock-S (A) read A		
		Upgrade (C) write C
Upgrade (B) write B		
UL (A, B)		
		Lock-S (B) read B
		UL (B, C)

评分细则：结论正确给 1 分，流程图正确给 4 分，或者文字解释，表明题意也可给

分，结论错误不给分

(4)
T1 must commit before T3 does.
T2 must commit before T1 does.
T2 must commit before T3 does.
评分细则：每条执行顺序正确给 1 分，全对给 3 分

Problem 8: Recovery (14 points)

Given the following log file that supports logical undo, please answer the following questions:

(1) The system crashes just after the last log record. What are the values of B and C in the database after system crash? (3 points)
(2) Which transactions should redo and undo, respectively? (3 points)
(3) What are the start and end points for redo and undo, respectively? (3 points)
(4) What are the log records added during recovery? (5 points)

1 <T_0 start>
2 <T_0, B, 2000, 2050>
3 <T_1 start>
4 <T_1, B, 2050, 2100>
5 <T_1, O_1, operation-begin>
6 <checkpoint {T_0, T_1}>
7 <T_1, C, 700, 400>
8 <T_0 commit>
9 <T_1, O_1, operation-end, (C, +300)>
10 <T_2 start>
11 <T_2, O_2, operation-begin>
12 <T_2, C, 400, 300>
13 <T_2, O_2, operation-end, (C, +100)>
14 <T_2, commit>

Answers of Problem 8:

(1)
B=2100
C= 300 or 400 or 700
评分细则：B 值正确给 1 分，C 值漏写扣 1 分，三个值完整写出给 2 分

(2)
redo: T_0 and T_2 undo: T_1
评分细则：redo 完整写出给 2 分，漏写扣 1 分，undo 写对给 1 分

(3)
redo: 7-14　　undo: 14-3
评分细则：redo、undo 顺序错写一个扣 1 分，redo 顺序写成 6-14 也可给分，全对给 3 分

(4)
<T_1, C, 600>
<T_1, O_1, operation-abort>
<T_1, B, 2050>
<T_1, abort>
评分细则：错一个扣 1 分，写出一个给 1 分，全对给 5 分

2022~2023 学年春夏学期数据库系统期末考试（回忆卷）

一、选择题（每题 3 分，共 8 题）

1、对于如下定义的一个表

```
create table course (
    course_id char(2) primary key,
    credit int,
    prereq_id char(2),
    foreign key prereq_id references course(course_id)
        on delete cascade
        on update cascade
);
```

其中有如下数据：

course_id	credit	prereq_id
11	2	
22	4	11
33	3	
44	3	22
55	4	22
66	3	33

在该表上执行如下 SQL 语句后，最后一条语句返回的结果是：

```
(1) insert into course('77', 4, '11');
(2) delete from course where course_id = '11';
(3) update course set course_id = '11' credit = 2 where course_id = '33';
(4) select count(*) from course where credit > 2;
```

A. 1

B. 2

C. 3

D. 4

2、对于如第一题中定义的表 course，对于如下 SQL 查询语句：

```
select c1.course_id, c1.credit, c2.course_id, c2.credit
from course as c1, course as c2
where c1.prereq_id = c2.course_id and c1.credit = 4 and c2.credit > 2;
```

下列关系代数表达式中与上面的 SQL 查询语句不等价的是：*(这道题的选项有点记不清了，写了一个差不多的)*

A. $\prod_{\text{c1.course\_id, c1.credit, c2.course\_id, c2.credit}}(\sigma_{\text{c1.credit}=4 \wedge \text{c2.credit}>2}(\rho_{\text{c1}}(\text{course}) \bowtie \rho_{\text{c2}}(\text{course})))$

B. $\prod_{\text{c1.course\_id, c1.credit, c2.course\_id, c2.credit}}(\sigma_{\text{c1.credit}=4 \wedge \text{c2.credit}>2 \wedge \text{c1.prereq\_id}=\text{c2.course\_id}}(\rho_{\text{c1}}(\text{course}) \times \rho_{\text{c2}}(\text{course})))$

C. $\prod_{\text{c1.course\_id, c1.credit, c2.course\_id, c2.credit}}(\sigma_{\text{c1.prereq\_id}=\text{c2.course\_id}}(\rho_{\text{c1}}(\sigma_{\text{credit}=4}(\text{course})) \times \rho_{\text{c2}}(\sigma_{\text{credit}>2}(\text{course}))))$

D. $\prod_{\text{c1.course\_id, c1.credit, c2.course\_id, c2.credit}}(\sigma_{\text{c1.prereq\_id}=\text{c2.course\_id}}(\rho_{\text{c1}}(\sigma_{\text{credit}=4}(\text{course})) \times \prod_{\text{course\_id, credit}}(\rho_{\text{c2}}(\sigma_{\text{credit}>2}(\text{course})))))$

3、关系模式R(A,B,C,D,E,F)上有函数依赖关系 {A->B, B->C, C->B, D->E, D->F}。下列选项中是该关系模式的候选键的是：

A. AC

B. AD

C. AE

D. AF

4、函数依赖集{A->B, A->D, BC->E, AC->D}的正则覆盖是 *(这道题选项记不太清了)*

A. {A->BDE, BC->E}

B. {A->BD, ABC->E}

C. {A->BD, C->E}

D. {A->BD, BC->E}

5、关系 A 上有 20,000 条记录。对于关系 A 的非键属性建立 B+树主索引。每个 B+树节点的大小是 2KB，每个 B+树内键值的大小是 32B，每个指针的大小是8B。若某次查询得到的结果包含在 5 个块中，则此次查询的开销是

A. 4 seek + 8 transfer

B. 4 seek + 9 transfer

C. 5 seek + 8 transfer

D. 5 seek + 9 transfer

6、关系 r 上有 50,000 条记录，每个块上可以容纳 r 上的 200 条记录；关系 s 上有 20,000 条记录，每个快上可以容纳 s 上的 50 条记录。在不考虑递归分块的情况下，对 r 和 s 进行 hash join 的开销约为

A. 1950 transfer + 1350 seek

B. 1950 transfer + 650 seek

C. 3300 transfer + 1350 seek

D. 3300 transfer + 650 seek

7、关系 r 上有 30,000 条记录，每个块上可以容纳 200 条记录。若在 r 的候选键上进行等值查找，则返回的结果的大小约为

A. 1

B. 20

C. 1,500

D. 30,000

8、在下面的执行计划中，lock-S(A)表示给 A 加上共享锁，lock-X(B)表示给 B 加上互斥锁。*(这道题的执行计划有点记不清了，写了一个差不多的)*

T1	T2	T3
lock-S(A)		
	lock-X(A)	
	lock-X(B)	
		lock-X(B)
		lock-S(C)
lock-X(C)		

对于上面的执行计划，下列说法正确的是：

A. 计划中没有死锁

B. 计划中有死锁，T1 在等待 T2 放锁的同时 T2 也在等待 T1 放锁

C. 计划中有死锁，T2 在等待 T3 放锁的同时 T3 也在等待 T2 放锁

D. 计划中有死锁，T2 在等待 T1 放锁，T3 在等待 T2 放锁，T1 在等待 T3 放锁

二、判断题（每题 2 分，共 6 题）

1. 任意关系模式均可分解为三范式，且分解是保持依赖的。

2. 在事务频繁更新数据的场景下，按列存储的效率比按行存储高。
3. 建立辅助索引的时候，为了降低空间开销，可以对每一个数据块建立稀疏索引。
4. 连接前尽可能早地进行投影操作可以降低投影操作的开销，所以投影操作应该尽早做。
5. 按索引查找的时间复杂度总是比顺序查找低。
6. 不能通过 2PL 产生的调度，可能可以由树协议产生。

三、SQL 语句（16 分）

对于如下定义的关系模式

user(userID, name, gender, age, group_name)

followship(userID, followerID)

写出可以进行如下查询的 SQL 语句：

（1）查询所有含有男性成员的群组（group）

（2）查询 userID 为'1001'的用户的追随者的信息

（3）查出女性成员最多的群组

（4）查询名称为'game'的群组当中追随者（follower）数量高于该群组中所有用户的追随者数量的平均值的用户。

四、数据库设计（16 分）

（1）请根据以下需求，为跟团旅行预订平台设计 E-R 图

- 每个用户都有用户 ID、用户名、密码和联系方式
- 每个旅行社需要提供旅行社的名称、法人代表身份证号、企业银行账号、位置，同时平台会给每个旅行社分配 ID
- 旅行社会出售旅行计划，旅行计划的内容包含标题、时长、描述
- 每个用户都可以预订旅行计划，订单信息包含用户信息、旅行计划信息、订单发起时间、旅行计划数量、总价格、支付状态
- 旅行结束后用户可以对旅行计划进行评价

（2）请将第 1 问设计的 E-R 图转化为关系模式，并指明主键和外键

五、并发控制（8 分）

对于如下的访问序列：*（题目中具体的序列记不清了，凭印象写了一个和题目差不多的）*

`r1(A) w2(A) w2(B) w3(A) r1(A) w3(B) r4(C) r4(B) w5(C) w5(B) r3(C)`

注：其中`ri(D)`表示事务`Ti`对数据`D`进行读操作，`wi(D)`表示事务`Ti`对数据`D`进行写操作。

（1）根据访问序列画出各个事务的前驱图

（2）判断这一访问序列是否冲突可串行化并说明理由

（3）判断这一访问序列是否可以由两阶段锁协议产生并说明理由

六、ARIES 恢复算法（12 分）

给出如下的 ARIES 日志： *（题目中具体的日志记不清了，这里写了一个差不多的）*

8001: <T1, begin>
8002: <T1, 3025.1, 20, 40>
8003: <T2, begin>
8004: <T2, 2628.2, 55, 555>
8005: <T3, begin>
8006: <T3, 2628.1, 30, 60>
8007: <T2, commit>
8008: <checkpoint>

Transaction	LastLSN
T1	8002
T3	8006

PageID	PageLSN	RecLSN
3025	8002	8002
2628	8006	8004

8009: <T4, begin>
8010: <T3, 2628.1, 60, 90>
8011: <T4, 6601.1, 30, 40>
8012: <T3, 2628.2, 555, 550>
8013: <T1, commit>
8014: <T4, 6601.2, 20, 30>

（1）写出在分析过程中被加入到脏页表的记录

（2）重做过程从哪一条日志开始

（3）写出恢复过程中增加的日志

（4）恢复过程结束后，6601.1和2628.2的值分别是多少

七、Buffer Tree（12 分）

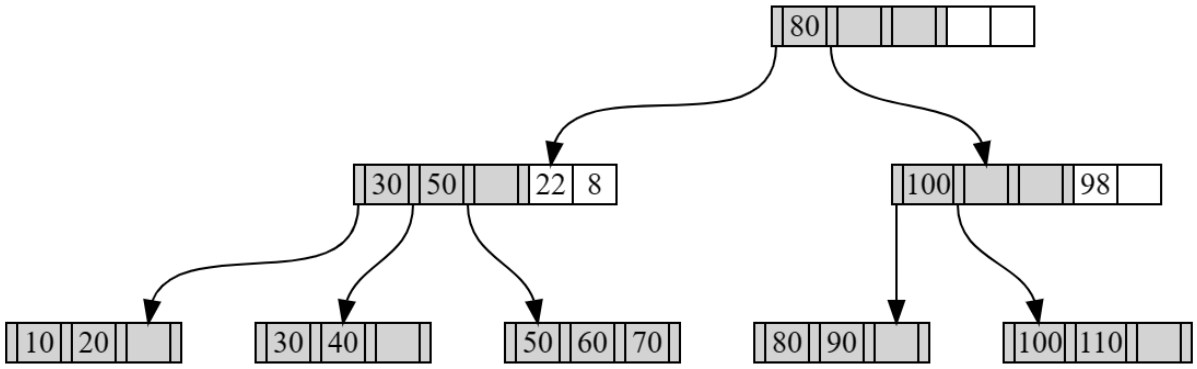
Buffer Tree 是一种写优化的索引结构。即在 B+树的每个内节点出增加一块缓冲区。

当进行插入操作的时候，我们直接将要插入的键值写入 Buffer Tree 根节点的缓冲区中。

当某一节点的缓冲区溢出的时候：

- 若该节点的子节点为内节点，则将该节点的缓冲区中的所有元素写入其子节点的缓冲区中。
- 若该节点的子节点为叶子，则将该节点的所有元素插入其子节点，当其子节点溢出的时候正常进行分裂。

本题中我们讨论缓冲区的大小为 2 且度为 4 的 Buffer Tree。其初始状态如下图：



（1）现对该树范围查询不大于 30、不小于 50 的键值，该过程中有多少个节点被访问?

（2）向该树中依次插入三个键值：38、92、28。请画出三次插入操作之后的 Buffer Tree。

（3）在第（2）问的插入操作中，每一次插入有多少节点被修改?

注：此处根节点始终被缓存在内存之中，所以计算时忽略根节点；分裂时新增加的节点视同被修改的节点。

1. Multiple Choice Questions (single answer, 12*3 points)

(1)

For the table **student(<u>id</u>, name, age, gender, dept_name)**, and the following query：

```
select  s1.id, s2.id
from student s1, student s2
where s1.dept_name=s2.dept_name and s1.age<18 and
        s2.age>28,
```

What algebra expression is ***not equivalent*** to above query?

(A) $\prod_{s1.id,\ s2.id}(\sigma_{s1.dept\_name=s2.dept\_name \wedge s1.age<18 \wedge s2.age>28}(\rho_{s1}(student) \times \rho_{s2}(student)))$

(B) $\prod_{s1.id,\ s2.id}(\sigma_{s1.age<18 \wedge s2.age>28}(\rho_{s1}(student) \bowtie \rho_{s2}(student)))$

(C) $\prod_{s1.id,\ s2.id}(\sigma_{s1.dept\_name=s2.dept\_name}(\sigma_{age<18}(\rho_{s1}(student)) \times \sigma_{age>28}(\rho_{s2}(student))))$

(D) $\prod_{s1.id,\ s2.id}(\sigma_{s1.dept\_name=s2.dept\_name}((\sigma_{age<18}(\rho_{s1}(student))) \times \prod_{id,\ dept\_name}(\sigma_{age>28}(\rho_{s2}(student)))))$

(2)

The table **r (<u>pid</u>, qid)** is defined as following:

```
create table r
 (      pid   char(2)   primary key,
        qid   char(2),
        foreign key (qid)  references r
                on delete cascade
                on update cascade );
```

An instance of ***r*** is as following：

pid	qid
11	
22	11
33	22
44	22
55	11

Executing the following SQL statements one by one:

a) **insert into r values ('11','55');**
b) **delete from r where pid='22';**
c) **update r set pid= '22' where pid ='11';**
d) **select count(*) from r where qid='22';**

What is the result of the statement **d**) above ?

(A) 1 （B）2 （C）3 （D）4

(3)

The functional dependency set **F={A→B, B→C, A→CD, BD→C }** holds on the relation **R(A, B, C, D, E)**.

What is the Canonical Cover of F?

(A) F={A→BCD , B→C, BD→C }

(B) F={A→BCD , B→C}

(C) F={A→B , A→D, B→C}

(D) F={A→BD , B→C}

(4)

The functional dependency set **F={A→B, B→CD}** holds on the relation **R(A, B, C, D, E)**.

What decomposition of R is ***not*** dependency preserving?

(A) R1(B,C,D) ,R2(A,B), R3(A,E)

(B) R1(A,B), R2(A,E), R3(B,C,D)

(C) R1(B,C), R2(B,D), R3(A,B),R4(A,E)

(D) R1(A,B), R2(A,C,D), R3(A,E)

(5) Which statement about data storage is ***incorrect***?

(A) **LRU** is the most suitable replacement strategy for buffer manager in any cases.

(B) For **heap file organization**, records can be placed anywhere in the file where there is free space.

(C) If the needed block is not in the buffer, the **buffer manager** will replace some other block, if required, to make space for the new block.

(D) The **system catalog** of a database stores metadata, such as Information about relations.

(6) What is ***not*** the benefit of **column-oriented storage**?

(A) **Reduced IO** if only some attributes of a relation are accessed

(B) **Improved CPU cache** performance when a query processor fetches the contents of a particular attribute

(C) **Reduced cost** of tuple deletion and update

(D) **Improved compression** of data of the same attribute

(7) Which statement about **index** is ***incorrect***?

(A) **Indexing mechanisms** are used to speed up access to desired data.

(B) **Range query** returns records with an attribute value falling in a specified range of values.

(C) In a **dense index**, index record appears for every search-key value in the file.

(D) **Secondary index** is an index whose search key specifies an order same as the sequential order of the file.

(8)

Assuming the height of the B+-tree index on the table **student(ID)** is 4. Considering the query: **select * from student where ID='2020160008',** the estimated cost for evaluating above query using the **Index Scan algorithm** is:

(A) 4 block transfers + 4 seeks.
(B) 5 block transfers + 5 seeks.
(C) 3 block transfers + 3 seeks.
(D) 5 block transfers + 4 seeks.

(9)

Assuming the table **r** has **160 blocks,** and buffer memory size is **10 blocks**. In the process of sorting **r** with the **External Merge Sort** algorithm, **2 buffer blocks** are allocated to each input run and to the output run, and t**he sorted result of the final pass is *written back* to disk.** The estimated cost for sorting **r** is:

(A) 800 block transfers + 272 seeks.
(B) 800 block transfers +512 seeks.
(C) 960 block transfers + 352 seeks.
(D) 320 block transfers +2 seeks.

(10)

Assuming the table **r** and **s** has **256** and **1024** blocks respectively. Each block has **4K bytes**. To do natural join **r** and **s** with the **Hash Join** algorithm, what is the **approximate minimum memory** needed to avoid recursive partitioning?

(A) 64K bytes （B）128K bytes (C) 1M bytes (D) 2M bytes

(11)

What equivalence rule of the relational algebra is ***incorrect***?

(A) $\sigma_{\theta 1}(\sigma_{\theta 2}(E)) \equiv \sigma_{\theta 2}(\sigma_{\theta 1}(E))$
(B) $\sigma_{\theta}(E_1 \cup E_2) \equiv \sigma_{\theta}(E_1) \cup E_2$
(C) $(E_1 \bowtie E_2) \bowtie E_3 \equiv E_1 \bowtie (E_2 \bowtie E_3)$
(D) $(E1 \cup E2) \bowtie E3 \equiv (E1 \bowtie E3) \cup (E2 \bowtie E3)$

(12)

There are following assumptions about the table **instructor** and **teaches**:

- The number of records in **instructor** is **4000**.
- The number of records in **teaches** is **8000**
- The number of distinct values of **dept_name** in **instructor** is **20**
- The value of **salary** in **instructor** is between **10000** and **90000**.

Please estimate the size returned by the following query:

select * from instructor natural join teaches on ID

where dept_name='CS and salary >=70000;

(A) 25 （B）50 （C）100 （D）200

2. SQL query(16 points，4 points per part)

Consider the following relational schema in a university:

course (course-id, title, department, credits)

prereq(course-id, prereq-id)

Write SQL statements to answer following queries:

(1) What course in CS department has the most credits?

(2) What department offers the most total credits of courses?

(3) Are there any courses with the same title?

(4) Calculate the number of times of each course being the prerequisites for other courses.

3. Database Design(16 points)

Every year scientist submit papers to various academic conferences. After peer review, a part of papers could be accepted and posted at the conferences. A database for a simplified Conference Management System (CMS) must be able to keep track of conferences, paper submissions and paper reviews.

- CMS maintains profile for each **user**: ID, password, name, and email.
- Users have multiple roles: author and/or reviewer, etc.
- An **author** includes a set of research subjects. A r**eviewer** includes an experience level.
- A **conference** includes conference ID, title, date, and city.
- A paper could be submitted to only one conference. Information of **paper submission** includes paper ID, title, abstract, submit date，status (i.e. accept, reject), and a file name of paper full text.
- Every author of a paper submission must contain rank, affiliation, and email.
- A paper submission could be reviewed by several reviewers. Information of **review** includes rating and comments.

(1) Please draw the E-R diagram for CMS database.

(2) Transform the E-R diagram into relational schemas.

4. Two-phase locking Protocol(8 points)

Please give a schedule with three transactions and at least a pair of conflict operations, to illustrate that the two-phase locking protocol is not a necessary condition for conflict serializability. The schedule should include **read**, **write**, **lock**, and **unlock** operations.

5. Aries Recovery Method (12 points, 3 points per part)

A DBMS uses Aries algorithm for system recovery. Following is a figure of a log file after system crashes. The log file consists of 15 log records with LSN from 2001 to 2015. Assuming that the last completed checkpoint is the log record with LSN 2011.

(1) When the Analysis Pass is complete, what is the contents of the DirtyPageTable?

(2) Which transactions should be undone?

(3) After recovery, what values are the data items identified by "5001.2" and "5002.2"?

(4) After recovery, what additional log records appended to the log fie?

2001： <T1 begin>

2002： <T2 begin>

2003： <T1 , 5001.1, 11, 111>

2004： <T2 , 5001.2, 22, 222>

2005： <T3 begin>

2006： <T3, 5002.1, 33, 333 >

2007： <T4 begin>

2008： <T2 commit>

2009： <T4, 5001.2, 222, 444 >

2010： <T1, 5002.2, 55, 555>

2011：

Txn	LastLSN
T1	2010
T3	2006
T4	2009

PageID	PageLSN	RecLSN
5001	2009	2003
5002	2010	2006

2012： <T1 commit>

2013： <T3, 5002.2, 555, 666>

2014： <T4, 5003.1, 77, 777>

2015： <T3 commit>

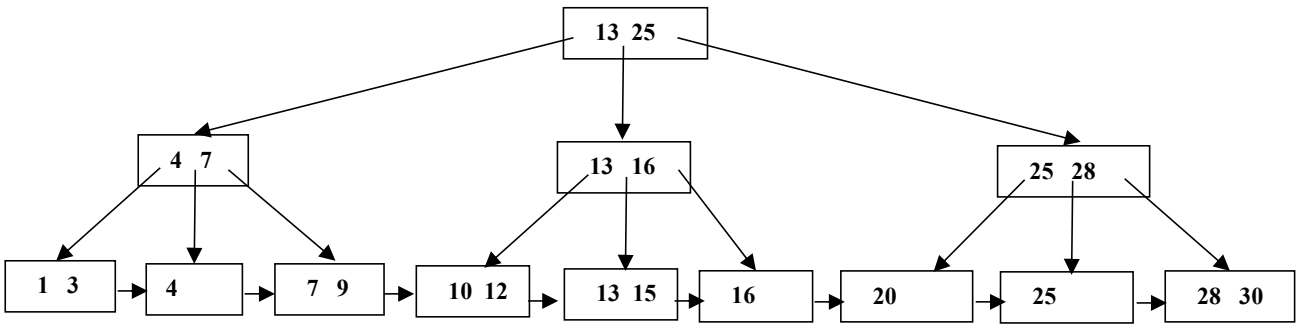
6. LSM-Tree index (12 points, 4 points per part)

Following is an example of a variant of the LSM tree which has multiple trees at each level. L0 is a B+-tree index at main memory. When L0 tree reaches its maximum size (i.e.13 blocks), it is written to disk as L_0^1 at level 0 using only sequential I/O operations. L_0^2 is another B+-tree index that has been written to disk at level 0 using only sequential I/O operations. When the number of B+-tree indexes at level 0 reaches its upper limit (i.e. 2), all the trees L_0^1 and L_0^2 are merged together into one combined new B+-tree at disk as L_1^1 at next level 1: the leaves of L_0^1 and L_0^2 are read sequentially, and the keys merged in sorted order, and the B+-tree L_1^1 at level 1 is constructed using standard techniques for bottom-up construction of B+-trees. For the sake of the exam, we assume the followings: (a) fan-out (n) of B+-tree is 3, (b) node size of B+-tree is the same as block size, (c) buffer space is big enough to load all leaf nodes of L_0^1 and L_0^2 simultaneously .

(1) Please estimate block transfers and seeks that are needed to write $\mathbf{L_0^1}$.

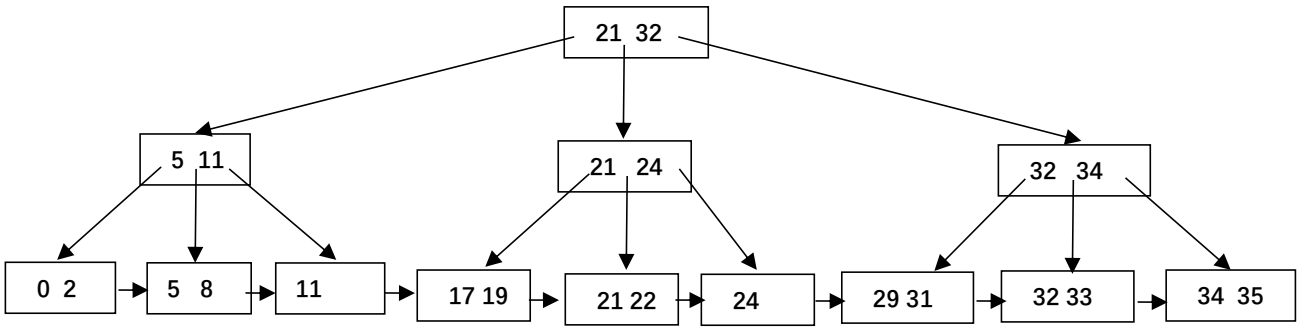
(2) Please estimate block transfers and seeks that are needed to lookup an index entry at leaf nodes by first searching the B+-tree at main memory and then searching the B+- trees at the level 0(i.e. $\mathbf{L_0^1}$ and $\mathbf{L_0^2}$) .

(3) Please estimate the block transfers and seeks that are needed to construct L_1^1 by merging $\mathbf{L_0^1}$ and $\mathbf{L_0^2}$。



L0 : B+-tree at main memory

$\mathbf{L_0^1}$: B+-tree written to disk (level 0)



$\boldsymbol{L_0^2}$**:** B+-tree written to disk (level 0)

Answers of problem 1：

Please fill the right choice into the blank for each question.

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)

Answers of problem 2：

Answers of problem 3：

Answers of problem 4：

Answers of problem 5：

Answers of problem 6：

Answers of Problem 1(每小题 3 分)

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)
B	A	D	D	A	C	D	B	C	A	B	C

第（9）小题答 A 给 2 分

Answers of problem 2(每小题 4 分)

（每错一处语法或语义，扣 1 分）

1)

```
select * from course
where department='CS' and
        credits =( select max(credits) from course where departmet='CS');
```

或：

```
select * from course
where department='CS' and
        credits >=all ( select credits from course where departmet='CS');
```

或：

```
select * from course
where department='CS'
order by credits desc
limit 1;
```

2)

```
select department
from course
group by department
having sum(credits)>=all( select sum(credits) from course    group by department);
```

或：

```
select department
from course
group by department
order by sum(credits) desc
limit 1;
```

3)

```
select title
from course C1
where exists (select * from course C2 where C2.title=C1.title and C2.course-id !=
                    C1.course-id)
```

或:

```
select C1.title
from course C1, course C2
where C1.title=C2.title and C1.course-id!= C2.course-id;
```

或

```
select C1.course-id, C2.course-id, C1.title
from course C1，course C2
where C1.title=C2.title and C1.course-id!= C2.course-id;
```

或:

```
select title
from course
group by title
having count(*)>1;
```

4)

```
select course.course-id, count(*)
from course, prereq
where course.course-id= prereq. prereq-id or course.course-id not in
                        (select prereq-id from prerequisite )
group by course.course-id
```

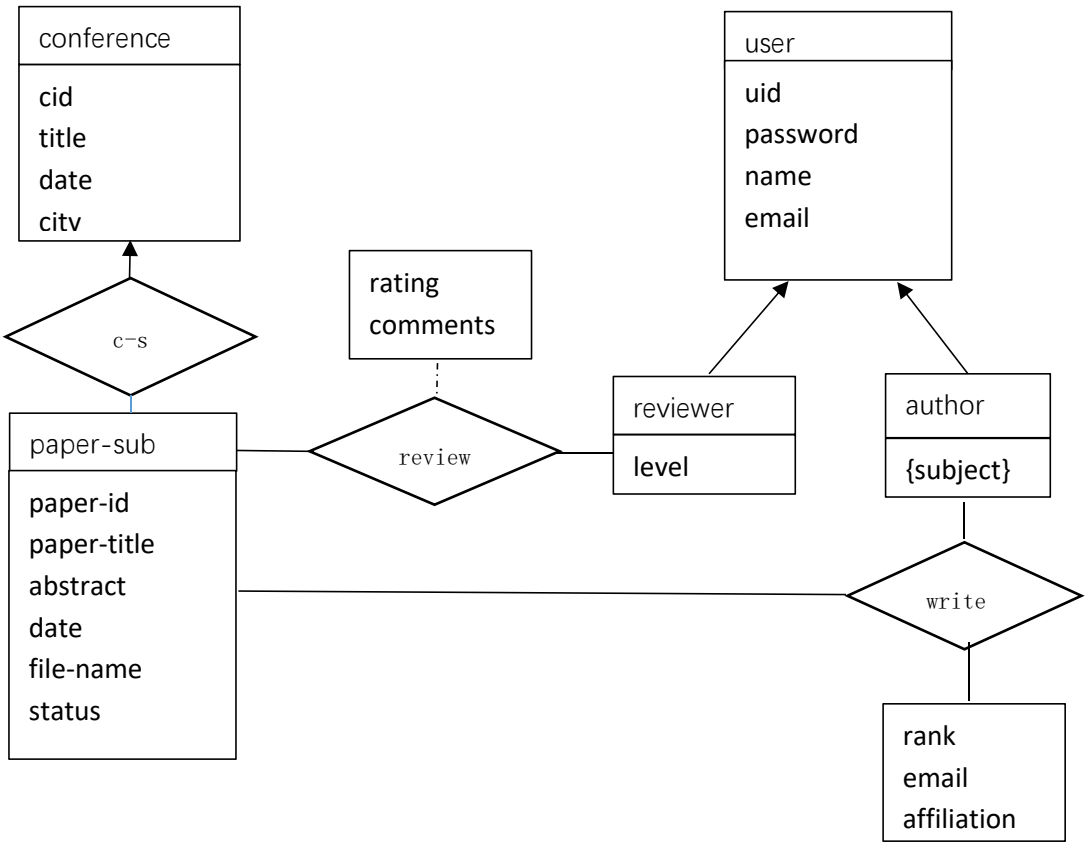
或：

```
select course.course-id, count(prereq-id)   // 或 count（prereq.course-id）
from course left outer join prereq on (course.course-id= prereq. prereq-id)
group by course.course-id
```

Answers of problem 3.

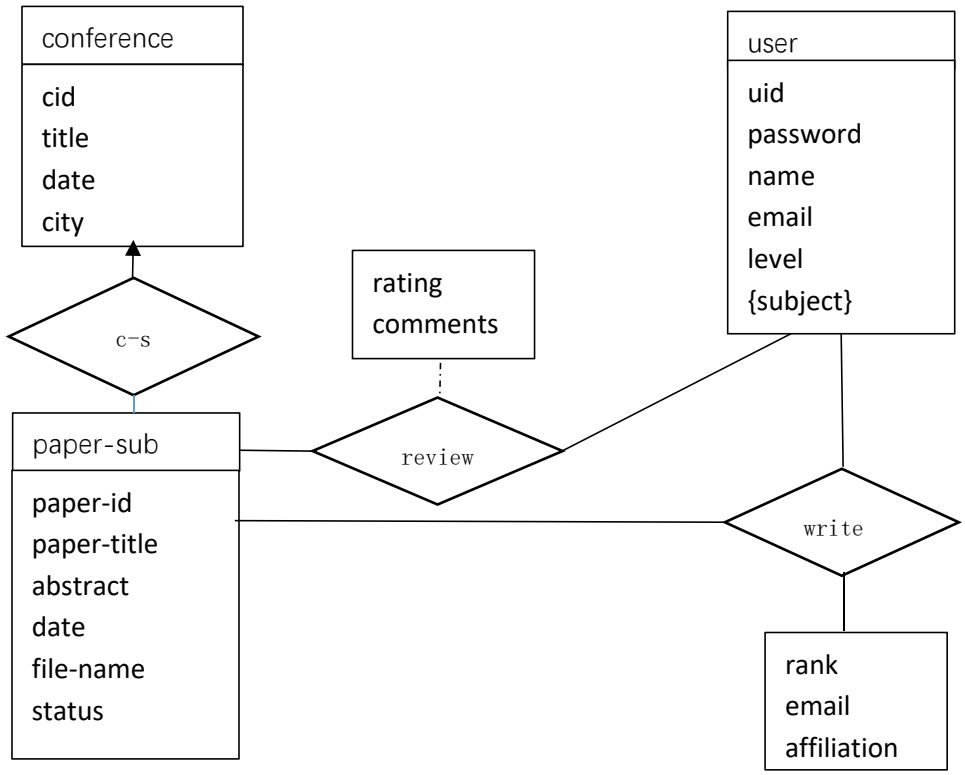
（1）8 分

（每个 Entity 或 Relationship 或 Mapping Cardinality 错扣一分）



或：

reviewer 和 author 不单列，它们的属性 level 和 {subject}合并到 user 中，也对。



（2）8 分，

（每个 relation 错扣 1 分）

user(uid, password, name , email)

*author(uid)

author-subject(uid, subject)

reviewer(uid, level)

conference(cid, title, date ,city)

paper-submission(pid, title, abstract ,file-name, status, date, cid)

write(uid, pid, rank, affiliation, email)

review(pid,uid, rating, comments)

→

user(uid, password, name , email, level)

author-subject(uid, subject)

conference(cid, title, date ,city)

paper-submission(pid, title, abstract ,file-name, status, date, cid)

write(uid, pid, rank, affiliation, email)

review(pid,uid, rating, comments)

Answers of problem 4:

给出正确的调度得 4 分，

说明非 2PL(如画出 precedence graph 、转化成串行调度)得 2 分

说明串行调度（如给出一个等价的串行调度），得 2 分。

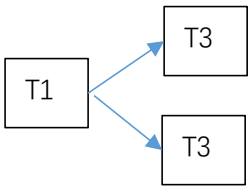
用树形协议的调度例子来说明也对。

下面是调度的一个例子。

T1	T2	T3
lock-x(A)		
write(A)		
unlock(A)		
	lock-x(A)	
	write(A)	
	unlock(A)	
lock-s(B)		
read（B）		
Unlock(B)		
		lock-x(B)
		write(b)
		unlock(B)

T1 doesn't follow 2PL.

The precedence graph is following



T1 doesn't follow 2PL. But the schedule is conflict serializable.

The serial order is T1→T2→T3 , or T1→T3→T2

Answers of problem 5:（每小题 3 分）

1)

PageID	PageLSN	RecLSN
5001	2009	2003
5002	2013	2006
5003	2014	2014

（错一行扣 1 分）

1) T4

2) 5001.2： 222

5002.2： 666

（错一个扣 1 分，错两个扣 3 分）

3) 2016： <T4, 5003.1, 77>

2017： <T4, 5001.2, 222 >

2018： <T4, abort >

（错一个扣 1 分）

Answers of problem 6（每小题 4 分）

(1) 13 block transfers + 1 seek

（transfers 、seek 错各扣 2 分）

(2) (0+3+6)/3=3, 3 block transfers + 3 seeks

（transfers 、seeks 错各扣 2 分）

分别说明三种情况的代价估计也全对。

但是三种情况的代价简单求和扣 1 分

(3) The size (number of nodes) of L_1^1 is (30/2)+15/3+5/3+1 = 15+5+2+1=23*（ 1 分）*

9 block reads + 1 seek for reading L_0^1. *（1 分）*

9 block reads + 1 seek for reading L_0^2 *（1 分）*

23 block writes +1 seek for writifg L_1^1 *（1 分）*

Total cost to construct L_1^1 is 41 block transfers and 3 seeks.

1. Single-choice Questions (24 points, 3 points*8)

1.1. The table course(id, credit, pre_course_id) is defined by the following SQL statement:

```
create table course(
    id char(2) primary key,
    credit int,
    pre_course_id char(2),
    foreign key(pre_course_id) references course(id)
    on delete cascade
    on update cascade
);
```

There is an instance of the course table below.

id	credit	pre_course_id
11	4	
22	2	11
33	3	
44	1	22
55	4	22
66	3	33

Executing the following SQL statements one by one:

1) insert into course values ('77', 4, '22');
2) delete from course where id = '11';
3) update course set id = '11' and credit = 2 where id = '33';
4) select count(*) from course where credit > 2;

What is the result of the statement 4) above?

(A) 1 (B) 2 (C) 3 (D) 4

1.2. Consider the table course (id, credit, pre_course_id) and the following query:

select c1.id, c2.id

from course as c1, course as c2

where c1.pre_course_id = c2.id and c1.credit=4 and c2.credit>2

Which algebra expression is ***not equivalent*** to above query?

(A) $\Pi_{c1.id,c2.id}\left(\sigma_{c1.credit=4\ \Lambda\ c1.credit>2}\left(\rho_{c1}(course) \bowtie \rho_{c2}(course)\right)\right)$

(B) $\Pi_{c1.id,c2.id}\left(\sigma_{c1.pre\_course\_id=c2.id\ \Lambda\ c1.credit=4\ \Lambda\ c1.credit>2}\left(\rho_{c1}(course) \times \rho_{c2}(course)\right)\right)$

(C) $\Pi_{c1.id,c2.id}\left(\sigma_{c1.pre\_course\_id=c2.id}\ \left(\sigma_{c1.credit=4}\left(\rho_{c1}(course)\right) \times \sigma_{c2.credit>2}\left(\rho_{c2}(course)\right)\right)\right)$

(D) $\Pi_{c1.id,c2.id}\left(\sigma_{c1.pre\_course\_id=c2.id}\left(\left(\sigma_{c1.credit=4}\left(\rho_{c1}(course)\right)\right) \times \Pi_{id,pre\_course\_id}\left(\sigma_{c1.credit>2}\left(\rho_{c2}(course)\right)\right)\right)\right)$

1.3. For relation schema R (A, B, C, D, E, F) with functional dependencies set F = {A->B, B->C, C->B, D->E, D->F}. Answer the following questions:

Which of the following ones is the correct **candidate key**?

(A) AC

(B) AD

(C) AF

(D) AE

1.4. The function dependency set $F = \{A \rightarrow B, A \rightarrow D,\ BC \rightarrow E,\ AC \rightarrow D\}$.

What is the **canonical cover** of *F*?

(A) $F = \{A \rightarrow BD, C \rightarrow E\}$

(B) $F = \{A \rightarrow BD, AC \rightarrow DE\}$

(C) $F = \{A \rightarrow B, ABC \rightarrow DE\}$

(D) $F = \{A \rightarrow BD, BC \rightarrow E\}$

1.5. Suppose there is a relation *r* with 20000 records on which a clustering B^+-tree index is constructed on a non-candidate key. Each B^+-tree node has a maximum size of 2048 bytes. Each search key occupies 32 bytes, and each pointer occupies 8 bytes. **A selection operation** is performed by scanning the B^+-tree index, where the selection condition specifies an equality comparison on the search key. 5 blocks contain records that match the specified search key. In the worst case, how many seeks and block transfers are required for the selection operation?

(A) 4 seeks and 8 block transfers

(B) 4 seeks and 9 block transfers

(C) 5 seeks and 8 block transfers

(D) 5 seeks and 9 block transfers

1.6. There are two relations ***r*** and ***s*** to perform **hash join operation**. The relation ***r*** has 20,000 records, and each block can store 50 records of ***r***. The relation ***s*** has 5,000 records, and each block can store 20 records of ***s***. Assume ***s*** is the build input. There is no recursive partitioning. A hash function is used to partition ***r*** and ***s*** into 10 partitions, respectively. Here, two buffer pages are available for each partition during the hash partitioning phase. Please estimate the seeks and blocks for the hash join operation?

(A) 1950 transfers, 650 seeks

(B) 1950 transfers, 1320 seeks

(C) 3300 transfers, 650 seeks

(D) 3300 transfers, 1320 seeks

1.7. Suppose that a table with 30000 records is stored in a file, where each file block holds 200 records. A selection operation is performed, where the selection condition specifies an equality comparison on a candidate key. What is the estimated size of the result?

(A) 1 (B) 150 (C) 200 (D) 30000

1.8. The following table shows some lock requests made by three transactions, T1, T2,

and T3. Lock-S and Lock-X stand for “shared lock” and “exclusive lock”, respectively. At this time, no granted lock is released.

T1	T2	T3
Lock-X(A)		
	Lock-S(B)	
Lock-X(B)		
		Lock-X(C)
	Lock-S(C)	
		Lock-X(B)

Which of the following statements is correct?

(A) There is no deadlock.

(B) There is deadlock, because T1 is waiting for T3, and T3 is waiting for T1.

(C) There is deadlock, because T2 is waiting for T3, and T3 is waiting for T2.

(D) There is deadlock, because T1 is waiting for T2, T2 is waiting for T3, and T3 is waiting for T1.

2. True or False Questions (12 points, 2 points *6)

2.1. It is always possible to losslessly decompose a relation into relations in Third Normal Form and the dependence is preserved. (True or False)

2.2. Column storage performs better than raw storage in transaction scenarios where data needs to be updated frequently. (True or False)

2.3. When creating a secondary index, if we want to reduce space overhead, we can use a sparse index with one index entry per data block. (True or False)

2.4. Index scan algorithms always have a lower time complexity than full table scans. (True or False)

2.5. Performing the projection as early as possible can reduce the size of the temporary relation that are generated by joining two relations. (True or False)

2.6. Schedules, which are impossible under two-phase locking, can be possible under the tree protocol. (True or False)

3. SQL query (16 points)

Consider the following relational schema in the social network:

user(<u>user_id</u>, name, gender, age, group_name)
followship(<u>user_id</u>, <u>follower_id</u>)

The underlined attributes are primary keys, and foreign keys are listed as follows:

The “user_id” and “follower_id” of followship both references the “user_id” in the user table

Write **SQL statements** to answer the following queries.

(1) Write the SQL statement to find the groups (without redundancy) that have male users.

(2) Find the people that are the followers of the user with id "1001" and is older than 18. Output their information including user_id and name.

(3) Output the group(s) that have the most female users. Note that, if there are multiple groups satisfying the condition, output all of them.

(4) In the "game" group, identify users whose number of followers is greater than the average number of followers of users in the "game" group. Output their information including user_id, name, and the number of followers

4. Database design (16 points)

This year, the tourism industry is recovering. One popular way to plan trips is to use online travel websites like Tuniu and Xiecheng. To do this, the database of travel websites needs to keep track of information about travel agencies, travel plans, and users.

- The database stores basic information about each **user**, such as user's ID, password, name, and telephone number.
- For each **travel agency** that wants to sell the travel plans, it must upload its name, the legal person's ID number (法人身份证号), the bank card number of the corporate account (对公账户银行卡号), and its location. The travel database would also allocate each agency an ID for identification.
- Each travel agency can issue different **travel plans** and specify the information of the title, the total hour, the description of the travel, and the price.
- Each user can **order** the required travel plan according to his preference. The order information, such as the order ID, the user ID, the travel plan ID, the amount purchased, the total price, and the order time.
- After traveling, the user can **rate** the travel agency, and the travel database should record these rating scores.

(1) Please draw the E-R diagram of the travel database.

(2) Transform the E-R diagram into the relational schemas and specify the primary keys and foreign keys of these relations

5. Concurrency Control (8 points)

Consider following schedule ***S*** with five transactions T1, T2, T3, T4, and T5:

S: r1(A) w2(A) r2(B) w3(B) w3(C) r1(C) w4(B) r4(D) w5(D) r3(D)

where: ***ri(X)*** means transaction ***Ti*** read data ***X***.

wi(X) means transaction ***Ti*** write data ***X***.

(1) Draw the precedence graph of ***S***.

(2) Explain whether ***S*** is serializable.

(3) Explain whether ***S*** could be generated by the two–phase locking protocol.

6. ARIES Recovery. (12 points)

Suppose a DBMS uses the ARIES method for crash recovery. You are given part of the log file at the time of a system crash, where each log record may contain the transaction ID, the page ID and slot number, the original value, and the updated value.

8000: <T1, 6421.1, 10, 20>

8001: <T2 start>

8002: <T3 start>

8003: <T2, 3462.1, 30, 40>

8004: <T3, 7923.1, 50, 60>

8005: <T2, 3462.1, 40, 70>

8006: <T1, 6421.1, 20, 80>

8007: <T3 commit>

8008: checkpoint

Active Transaction Table

TXN	LastLSN
T1	8006
T2	8005

DirtyPageTable

PageID	PageLSN	RecLSN
3462	8005	8003
6421	8006	8006
7923	8004	8004

8009: <T4 start>

8010: <T4, 5235.1, 60, 90>

8011: <T1 commit>

End of log at crash

Suppose that the PageLSN of each page on disk is less than 8000. Given this information, answer the following questions:

(1) During the analysis pass, what record is added to the DirtyPageTable?

(2) Which log record is the first to be redone in the redo pass?

(3) What log records should be appended to the log file during recovery? For CLR (compensation log records), you should provide the transaction ID, the page ID and slot number, and the original value (i.e., the value to be recovered).

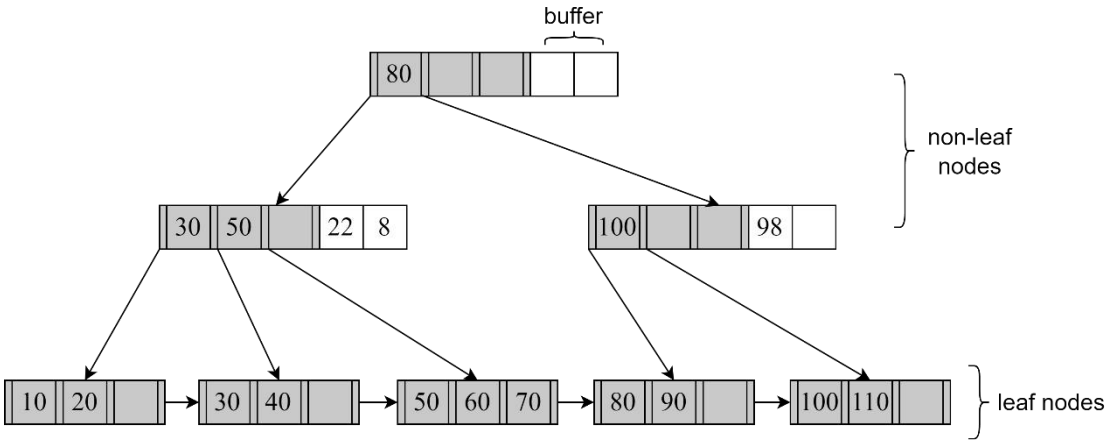
(4) After recovery, what are the values of locations 3462.1 and 6421.1, respectively?

7. Buffer Tree. (12 points)

A buffer tree is an index structure that performs updates lazily to save I/O operations. In this question, you are given a simplified version of buffer tree with the following properties:

i. Each non-leaf node consists of a B^+-tree node with a max fanout of 4, and a buffer that can hold up to 2 items (i.e., index entries). Meanwhile, each leaf node is a regular B^+-tree leaf node that can hold up to 3 items.
ii. When an item is inserted, it is first inserted into the buffer of the root node. (We assume that the root node is not a leaf node.)
iii. If a node's buffer is overfull (i.e., contains more than 2 index entries), the following procedure is performed:
    a. If the node is an internal node, the items are pushed down to the appropriate buffers of child nodes one level below. After all the items are pushed down, if any of the children has an overfull buffer, apply this procedure recursively.
    b. If the node has leaf nodes as children, move the smallest item from the buffer to the appropriate leaf node. If the leaf is overfull, split it in the usual B^+-tree manner. Then repeat this step until no item is left in the buffer.

Below is the content of the buffer tree.



Answer the questions below.

(1) For a range query from 30 to 50 (both inclusive), how many index nodes are accessed? Explain the procedure.

(2) Provide a diagram of the buffer tree after inserting 38, 92, and 28, in that order, into the buffer tree.

(3) Assume that each buffer tree node (with the optional buffer) is stored in a disk block. The root node is cached in memory, so it is excluded from the following calculation. How many blocks are modified for each insert operation in question (2)? If a block is created, count it as a modified block.

1. Single-choice Questions (24 points, 3 points*8)

Answer

A A B D D A A C

2. True or False Questions (12 points, 2 points *6)

Answer

T F F F T T

3. SQL query (16 points)

(1) **select distinct group_name**

from user

where user.gender = "male";

语法/语义错写、漏写每个扣一分，如没写male条件、distinct没写

(2) **select user.user_id, name**

from user, followship

where user.user_id = followship.follower_id and followship.user_id = "1001" and user.age > 18;

4分，where中每个查询条件一分，select和from子句写对一分，

select中user_id没指出表名扣一分；

where中join条件写错扣一分；

(3) 语法/语义错写、漏写每个扣一分，如没写female条件、group by条件写错、直接写max函数（而不是写在select里面）、limit 1扣一分

select group_name

from user

where user.gender = 'female'

```
group by group_name
having count(user_id) >= all (
    select count(user_id)
    from user
    where user.gender = 'female'
    group by group_name);
```

或

```
select group_name
from user
where user.gender = 'female'
group by group_name
order by count(user_id) desc
limit 1;
```

(这种答案算部分对，因为只输出了一个结果，实际可能有多个，去年这种算对的)

或使用with 子句

```
with T(group_name, female_count) as (select group_name, count(user_id)
    from user
    where user.gender = 'female'
    group by group_name)
select group_name
from T
where T.female_count = (select max(female_count) from T);
```

(4)

语法/语义错写、漏写每个扣一分，如没写game条件、group by条件写错、join条件写错、直接写avg函数（而不是写在select里面）、limit 1扣一分

```
select user.user_id, name, follower_count
from user, (select user.user_id, count(follower_id) as follower_count
            from user, followship
```

```
        where user.user_id = followship.user_id and user.group_name = 'game'
        group by user.user_id) as T
where user.user_id = T.user_id and T.follower_count > (select avg(follower_count)
from T);
```

with 子句类型

```
with T(user_id, follower_count) as (select user.user_id, count(follower_id)
        from user, followship
        where user.user_id = followship.user_id and user.group_name = 'game'
        group by user.user_id)
with R(avg_follower_count) as (select avg(follwer_count) from T)
select user.user_id, name, follower_count
from user, T, R
where user.user_id = T.user_id and T.follower_count > R. avg_follower_count;
```

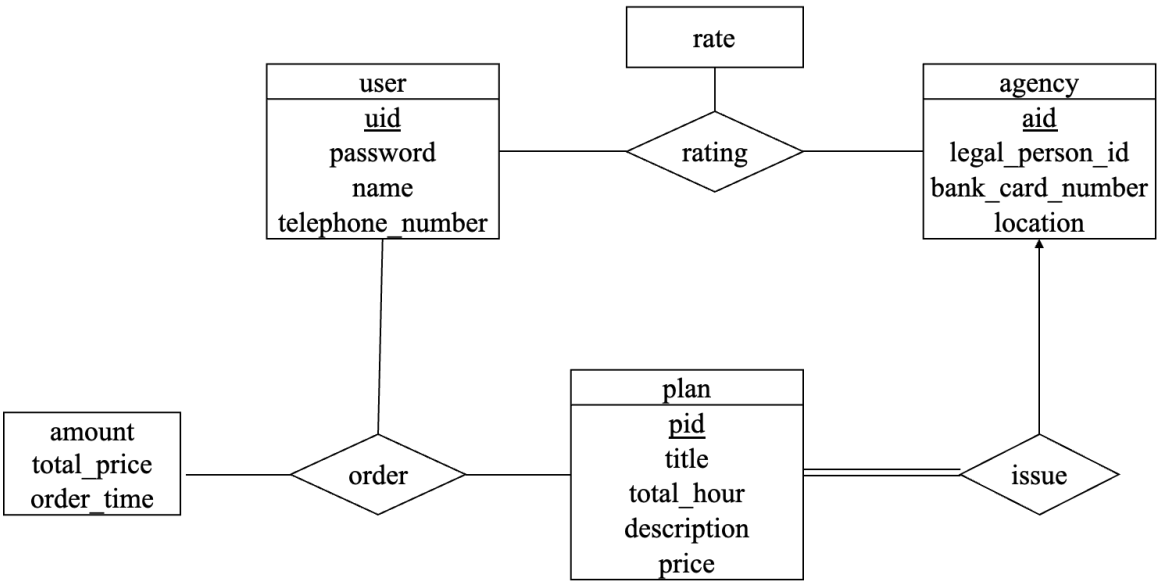
另有一解：

```
select user_id
from user natural join followship
where group_name = 'game'
group by user_id
having count(follower_id) >= ((
    select count(follower-id)
    from user natural join followship
    where user.group_name = 'game')   /
(select count(*)
 from user
 where group_name ='game'));
```

4. Database design (16 points)

Answer

(1)



评分细则：
每个实体（user、agency、plan）、关系（rating 的多对多关系、order 的多对多关系、issue 的多对多关系）、rating 和 order 关系衍生出的属性各一分，共 8 分；
每少写/错写减一分。
关系的名称随意，但是注意关系的类型和箭头；
注意 order 和 rating 每个多对多关系可以写成两个一对多关系，两个一对多关系需要注意箭头方向正确（指向一的实体）。

(2)

The relational schemas are as follows.
user(uid, password, name, telephone_number)
agency(aid, legal_person_ID, bank_card_number, location)
travel_plan(pid, title, total_hour, description, price, agency_ID, issue_time)
order(oid, user_ID, plan_ID, amount, total_price, order_time)
rating(user_ID, agency_ID, rate)

The primary key of each table is underlined. (rating 表如果有单独主键也给分，外键写明白即可)
The foreign keys are as follows:

- the agency_ID of the travel plan table references the aid of the agency table
- the user_ID of the order table references the uid of the user table
- the plan_ID of the order table references the pid of the travel plan table
- the user_ID of the user table references the uid of the user table
- the agency_ID of the rating table references the aid of agency table

评分细则：
一共 8 分

- user 和 agency 表正确，各一分
- travel_plan 表正确，主键和属性一分，外键一分；

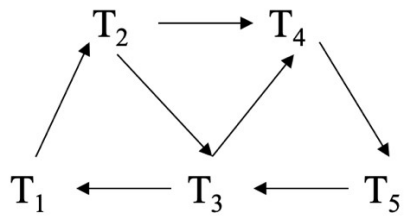
- order 表正确，主键和属性一分，外键一分；
- rating 表正确，主键和属性一分，外键一分；

没有指出外键属性-3 分；直接写 create 语句而不是 schema 扣 1 分。

5. Concurrency Control (8 points)

Answer

(1)



评分细则：图中的边多/少一条扣 1 分，全对给 3 分

(2) S is not conflict serializable，because there are cycles in the graph :

$T_1 \rightarrow T_2 \rightarrow T_3 \rightarrow T_1$

$T_4 \rightarrow T_5 \rightarrow T_3 \rightarrow T_4$

$T_1 \rightarrow T_2 \rightarrow T_4 \rightarrow T_5 \rightarrow T_3 \rightarrow T_1$

评分细则：答案错误扣 3 分，指出一个环算全对，全对给 3 分；没指出环扣 1 分。

(3) No，because every schedule generated by 2PL is serializable. :
评分细则：答案错误扣 2 分，理由错误扣一分，全对给 2 分

6. ARIES Recovery. (12 points)

Answer:

(1) **(5235, 8010, 8010)**. Log 8010 modifies page 5235, which should be added to the DirtyPageTable.

评分标准：2 分。

(2) **8003**, which is the smallest RecLSN in the DirtyPageTable.

评分标准：2 分。

(3) **<T4, 5235.1, 60>, <T2, 3462.1, 40>, <T2, 3462.1, 30>.**

The active transaction table after the analysis pass contains T2 and T4. The related update records should be redone.

评分标准：4 分，写出 3 个各得 1 分；<T2, 3462.1, 40>在<T2, 3462.1, 30>前面得 1

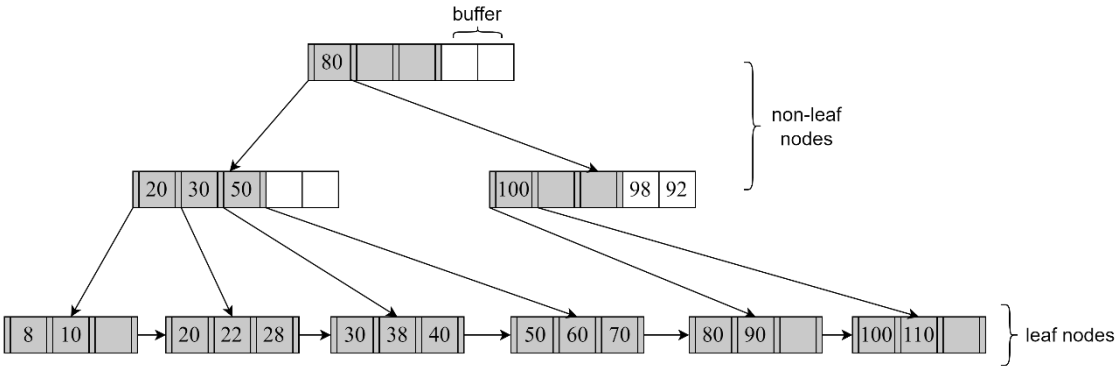
分；每多写一个扣 1 分。

(4) **30, 80**. Location 3462.1 is modified by T2, which is undone. Location 6421.1 is modified by T1, which is committed.
评分标准：4 分，1 个 2 分。

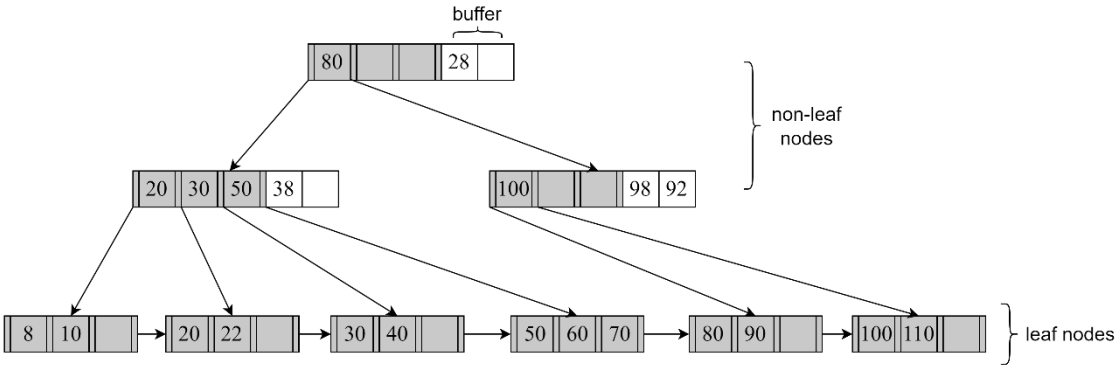
7. Buffer Tree. (12 points)

Answer:

(1) **4.** First, an equality search on 30 requires accessing the root node, the leftmost node on the second level, and the second leaf node on the left. Then, one additional leaf node needs to be searched to collect all items from 30 to 50.
评分标准：数量答对得 1 分，解释合理给 2 分（只要体现出 range query 的查询思路，就可以得分）。
(2)
答案一：按照题目要求，如果 buffer 溢出，将所有元素下推。



答案二：也可以只下推前两个元素，将剩下的元素放进原来的 buffer 里。



评分标准：6 分。根节点错扣 1 分，第二层两个节点错一个扣 1 分，第三层左边三个节点错一个扣 1 分，右边三个节点错任何一个扣 1 分，扣完为止。第二层右边的节点 buffer 如果写成 92 98，算对。两种答案，以令本大题得分较高的为准。

(3) 若(2)为答案一，则为 **0, 0, 5**；若(2)为答案二，则为 **0, 0, 4**。

解析：

插入	描述
38	只插到根节点 buffer
92	只插到根节点 buffer
28	需要修改第二层两个 buffer、左边三个（或者两个）叶子节点，第二层左边节点的指针，一共是 5 个（或者 4 个）节点

评分标准：3 分，1 个 1 分。