Quick Review:
Principles of Database Systems

June 24， 2021

Assessment

Course Grading Policy:

- Homework: 10%
- Quiz: 10%
- Lab & Project: 30%
- Final exam: 50% (open two-page note, handwriting, with student ID & name)

The chapters we have studied before mid-term

Chapter 1: Introduction

Chapter 2: Introduction to the Relational Model

Chapter 3: Introduction to SQL

Chapter 4: Intermediate SQL

Chapter 5: Advanced SQL

Chapter 6: Formal Relational Query Languages(6.1)

Chapter 7: Database Design and the E-R Model

Chapter 8: Relational Database Design

The chapters we have studied after mid-term

Chapter 10: Storage and File Structure

Chapter 11: Indexing and Hashing

Chapter 12: Querying Processing

Chapter 13: Query Optimization

Chapter 14: Transactions

Chapter 15: Concurrency Control

Chapter 16: Recovery System

Chapter 22: Object-based Databases

Chapter 23: XML

Relational Algebra

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".

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

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

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)
```

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 )
```

Relational Algebra

1. Relational Algebra(14 points, 7 points for each)

Consider the following Hotel Booking System Database (Primary keys are underlined):

Hotel (hotelNo, name, city, address)

Room (roomNo, hotelNo, type, price)

Booking (guestNo, hotelNo, roomNo, startdate, enddate)

Guest (guestNo, name, address, phone)

Please write relational algebra expressions for following queries.

(1) Find all the hotel names that have rooms with price below 200.

(2) Find how many rooms for each hotel in "Hangzhou" city.

(1) $\Pi_{name}(\sigma_{price<200}(Hotel \bowtie Room))$

(2) $_{hotelNo}\,G_{\,count(*)}\,(\sigma_{city=”Hangzhou”}\,(Hotel \bowtie Room))$

2. SQL (30 points, 6 points for each)

For schemas in question 1, please write the following SQL statements:

(1) Create the table Booking, with all the necessary constraints.

(2) Find all the guest names who had booked hotel rooms in "Hangzhou" city on date "2016-10-20".

(3) Find the hotels with the highest average room price.

(4) Rewrite the where clause in the following statement, without using the unique construct.

```
Select name from Guest A
Where unique ( select hotelNo from Hotel natural join Booking
                where Hotel.city='Hangzhou'
                and Booking.guestNo=A.guestNo )
```

(5) Suppose that the hotel (hotelNo is 12345) will have a new room (roomNo is 202) available from date "2016-10-20" on, and its another room (roomNo is 101) will need repair from date "2016-10-20" and not be available. So, we need shift all the bookings on room 101 to room 202 from "2016-10-20" on. Please write the SQL statements to carry out the task (Suppose the data for room 101 has been inserted into table Room. Please remember that one record in table Booking may correspond to two records after modification).

(1) Create table Booking

```
(    guestNo int,
     hotelNo int,
     roomNo int,
     startdate date,
     enddate date,
     primary key (hotelNo, roomNo, startdate),
     foreign key (guestNo) reference Guest,
     foreign key(hotelNo, roomNo) reference Room,
     check (startdate<=enddate))
```

(2)

```
Select guest.name
from Hotel, Booking, Guest
Where Hotel.hotelNo=Booking.hotelNo
        and Guest.guestNo=Booking.guestNo
        and city='Hangzhou'
        and date '2016-10-20' between startdate and enddate
```

(3)

```
Select hotelNo from Room
Group by hotelNo
having avg(price) >= (select avg(price)
                      from Room
                      group by hotelNo)
```

(4)

```
Where ((select count(hotelNo) from Hotel natural join Booking
        Where Hotel.city='Hangzhou'
        And Booking.guestNo=A.guestNo) =
       (select count(distinct hotelNo) from Hotel natural join
        Booking
        Where Hotel.city='Hangzhou'
        And Booking.guestNo=A.guestNo))
```

```
Insert into Booking
        ( select guestNo, hotelNo, date '2016-10-20', enddate, roomNo
        From Booking
        Where hotelNo=12345
        And roomNo=101
        And  startdate<date '2016-10-20'
        And enddate>=date '2016-10-20')
Update Booking set enddate='2016-10-19'
        Where hotelNo=12345
        And roomNo=101
        And  startdate<date '2016-10-20'
        And enddate>=date '2016-10-20')
Update Booking set roomNo=202
        Where hotelNo=12345
        And roomNo=101
        And  startdate=date '2016-10-20'
```

Relational Database Design

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.

1) {C E}
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)
3) The decomposition is dependency preserving.

Relational Database Design

Relational Database Design (16 points, 4 points for each)

Consider the following relation schema and functional dependencies:

R=(A, B, C, D), F= {B→A, C→A, AC→D, AD→C }.

(1) Compute the closure of attribute C.

(2) Explain why schema R isn't BCNF.

(3) Decompose R into a collection of BCNF relations, please show each step in detail.

(4) Tell whether your decomposition of (3) is dependency preserved or not, why?

(1) {ACD}

(2) candidates are {BC} and {BD}

①B→A, 分解为{BA}和{BCD}；对{BCD}, C→D, 分解为{BC}{CD}*（不保持依赖）*

②C→A, 分解为{CA}和{BCD}；对{BCD}, C→D, 分解为{BC}{CD}*（不保持依赖）*

③AC→D, 分解为{ACD}和{ABC}；对{ABC}, B→A, 分解为{BA}{BC}*（保持依赖）*, 或C→A, 分解为{CA}{BC}*（不保持依赖）*

④AD→C, 分解为{ACD}和{ABD}；对{ABD}，B→A，分解为{BA}{BD}*（保持依赖）*

⑤C→D，分解为{CD}和{ABC}；对{ABC}, B→A, 分解为{BA}{BC}, 或C→A, 分解为{CA}{BC})*（不保持依赖）*

⑥BC→A, 分解为{ABC}和{BCD}；对{ABC}, B→A, 分解为{BA}{BC}*（保持依赖）*, 或C→A, 分解为{CA}{BC}；对{BCD}, C→D, 分解为{BC}{CD}*（不保持依赖）*

⑦C→AD，分解为{ACD}和{BC}*（保持依赖）*

(4)见(3)

Entity-Relationship Diagram

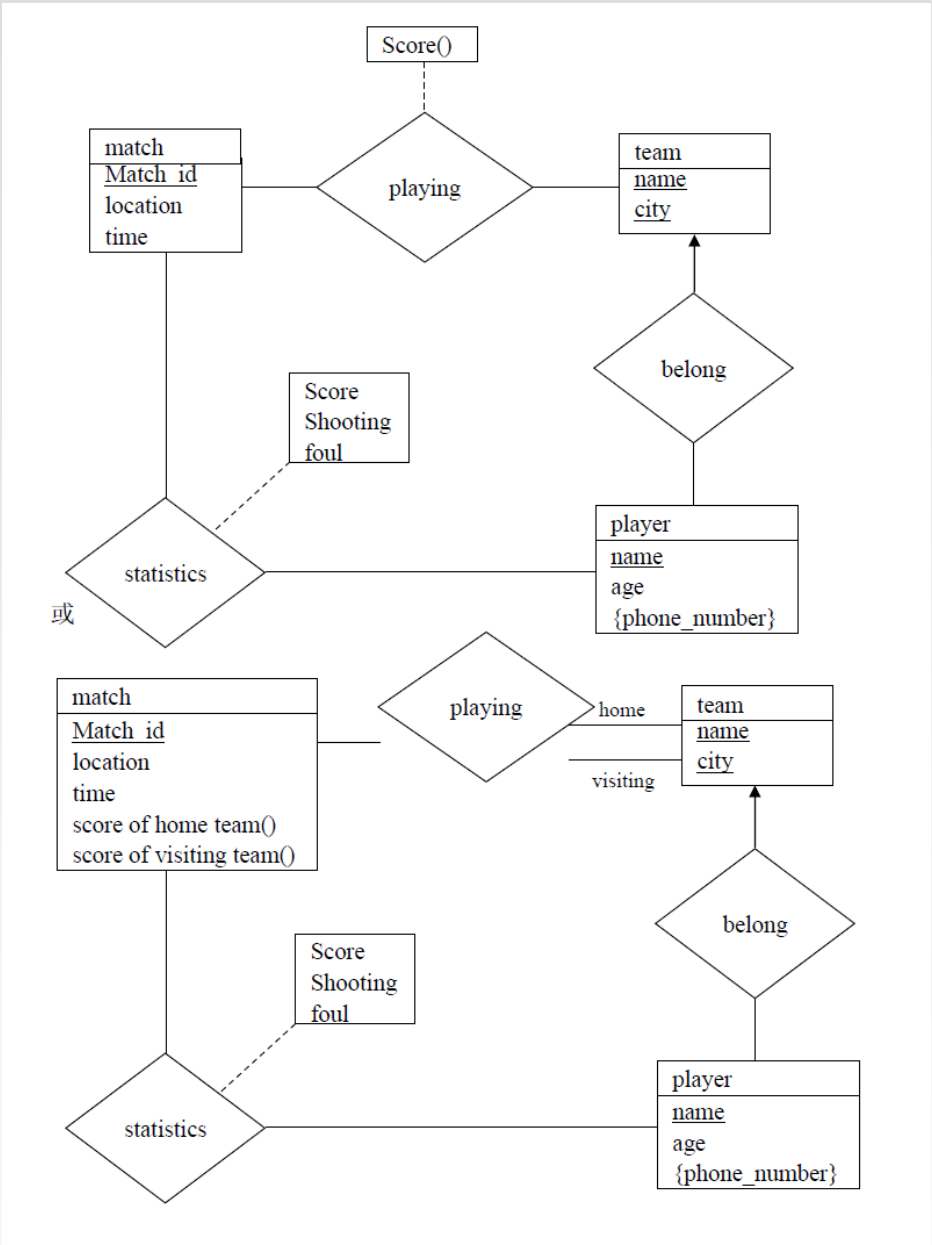
Problem 2: E-R Model (9 points)

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)



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.

Entity-Relationship Diagram

A university database needs to store information about all its professors (identified by prof_id, with name, salary and several phones as attributes), students (identified by s_no, with name and grade as attributes), research projects (identified by name, with budget, start date, and end date as attributes), and office (identified by o_no, with building and description as attributes).

Suppose the data has the following properties:

A. Each Professor has an individual office, students share offices.

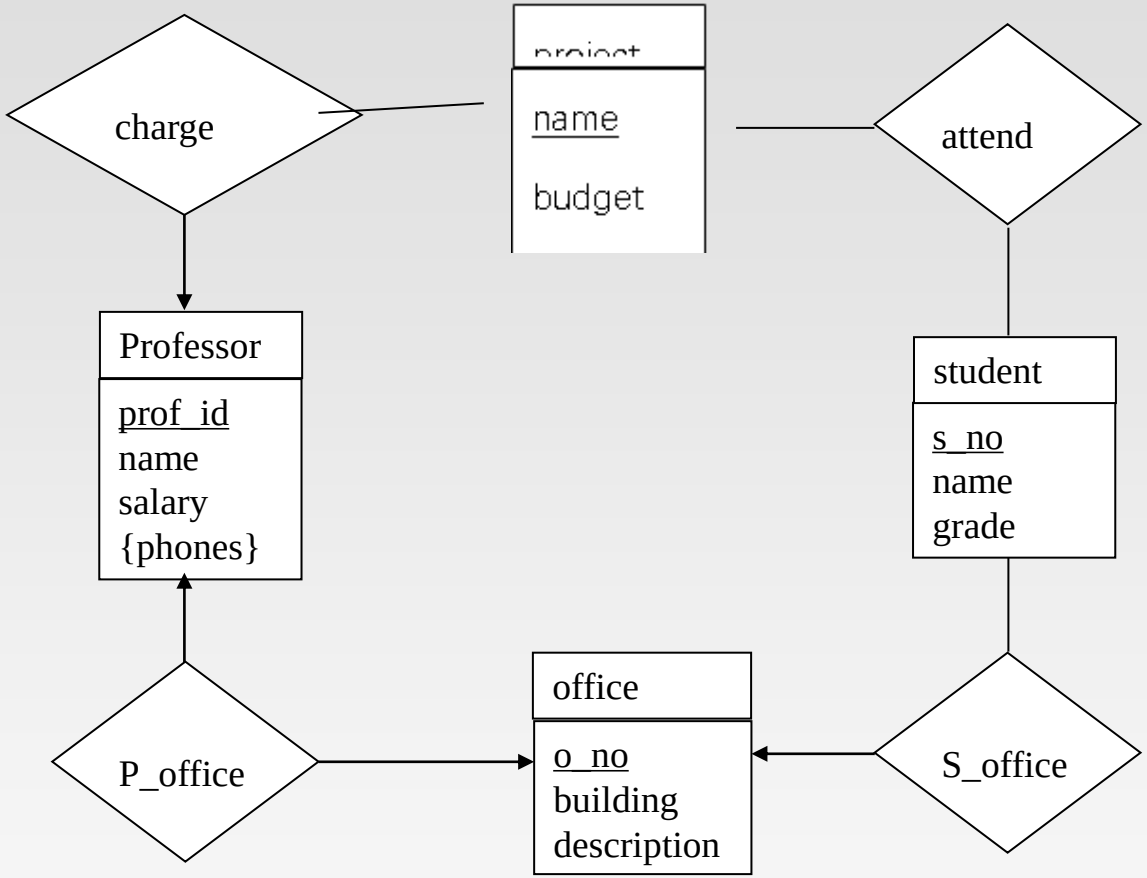
B. Each Student can join several projects.

C. Professors can be responsible for multiple projects, and each project is taken charge of by one professor.

Please answer the following questions:

(1) Draw the ER diagram for the application.

(2) Convert the ER diagram into a minimal number of BCNF relational database schemas, primary keys should be underlined.


charge
project
name
budget
attend
Professor
prof_id
name
salary
{phones}
student
s_no
name
grade
office
o_no
building
description
P_office
S_office

project(name, budget, start_date, end_date, prof_id)

Professor(prof_id, name, salary, o_no)

Phones(prof_id, phone)

Student(s_no, name, grade, o_no)

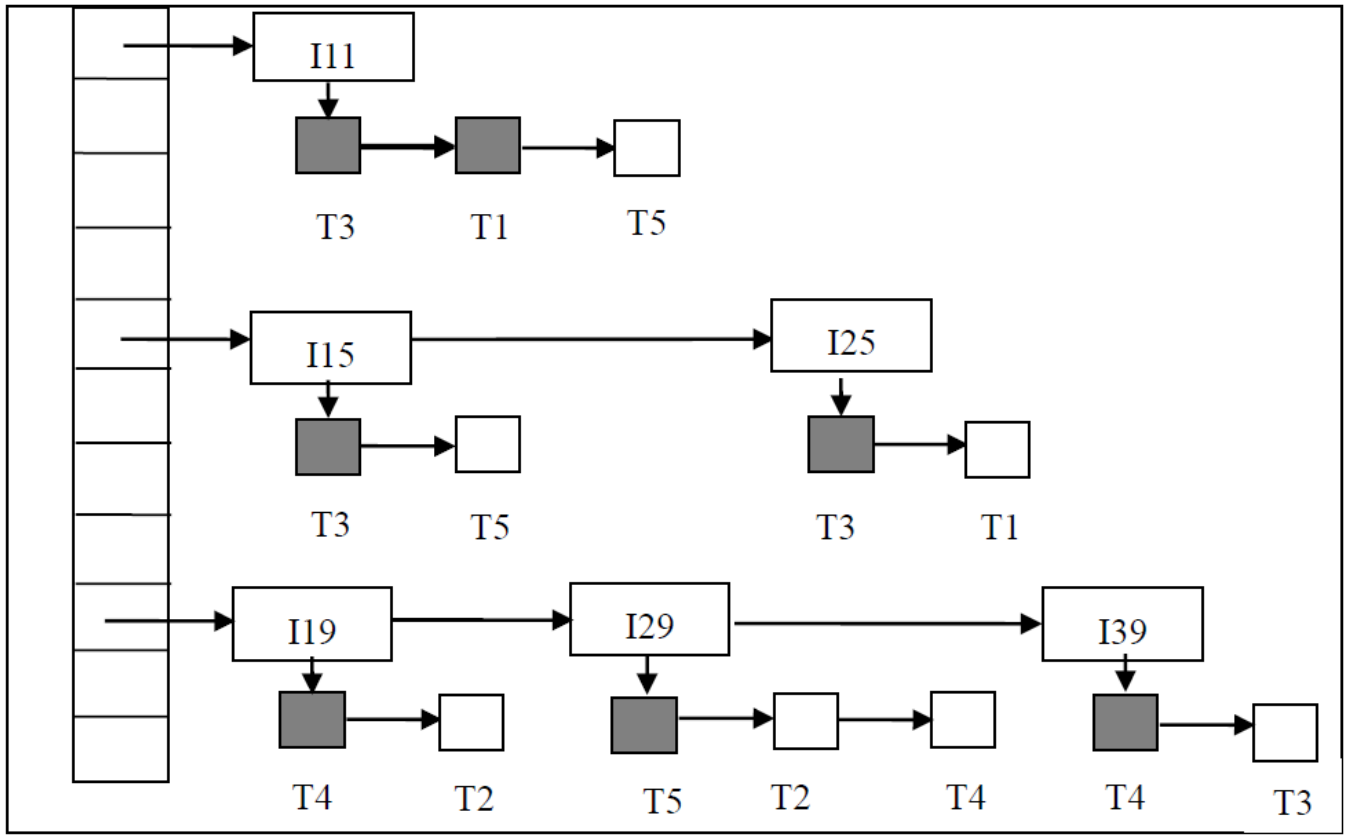
Attend(proj_name, s_no)

Office(o_no, building, description)

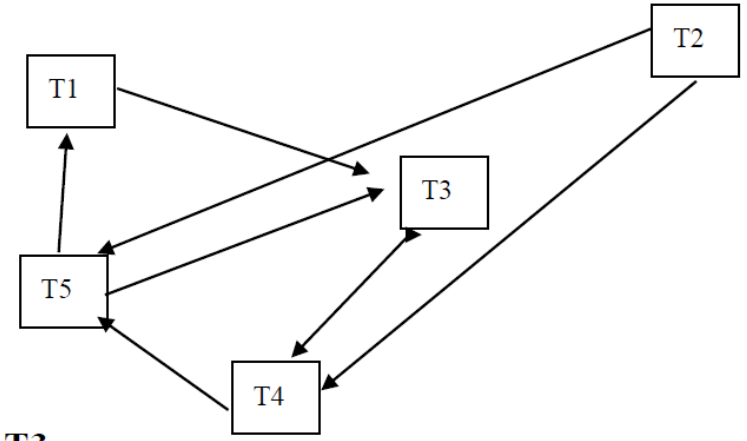
Problem 5: Deadlock Handling (12 points, 4 points per part)

The following figure shows an instance of a lock table. There are 5 transactions (T1-T5) and 6 data items (I11, I15, I19, I25, I29, I39). Granted locks are filled (black) rectangles, while waiting requests are empty rectangles.

1) Which transactions are involved in deadlock?
2) In order to break the deadlock and release most lock resources, which transaction (victim) should be rolled back?
3) Please draw the lock table after the victim transaction of 2) is rolled back.

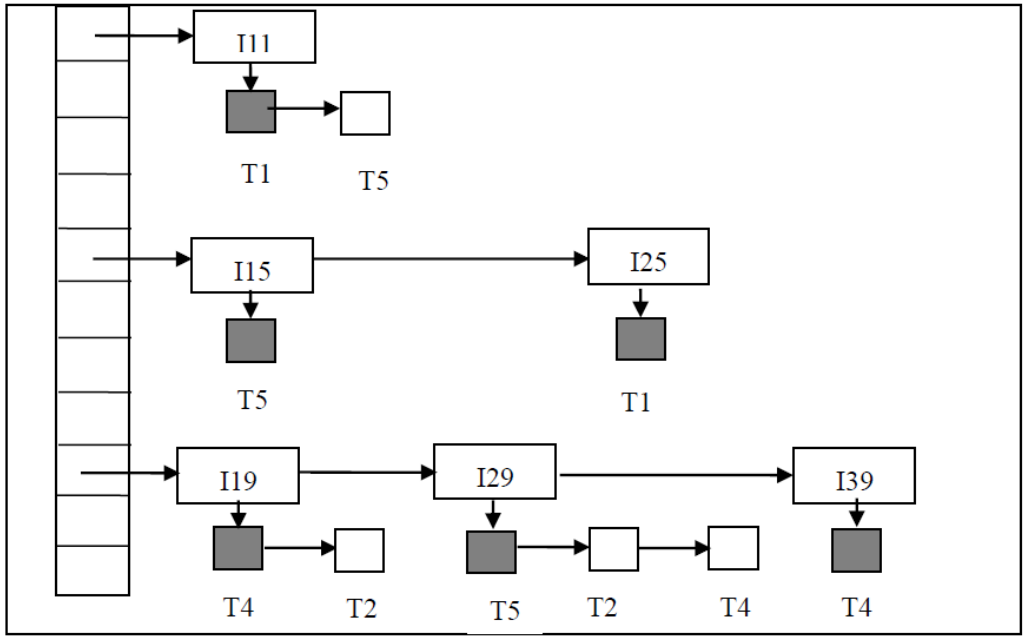


1） **T1，T3 ，T4， T5**



2） **T3**

3）



XML

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.

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)>
]>
```

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

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

B+ tree index

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

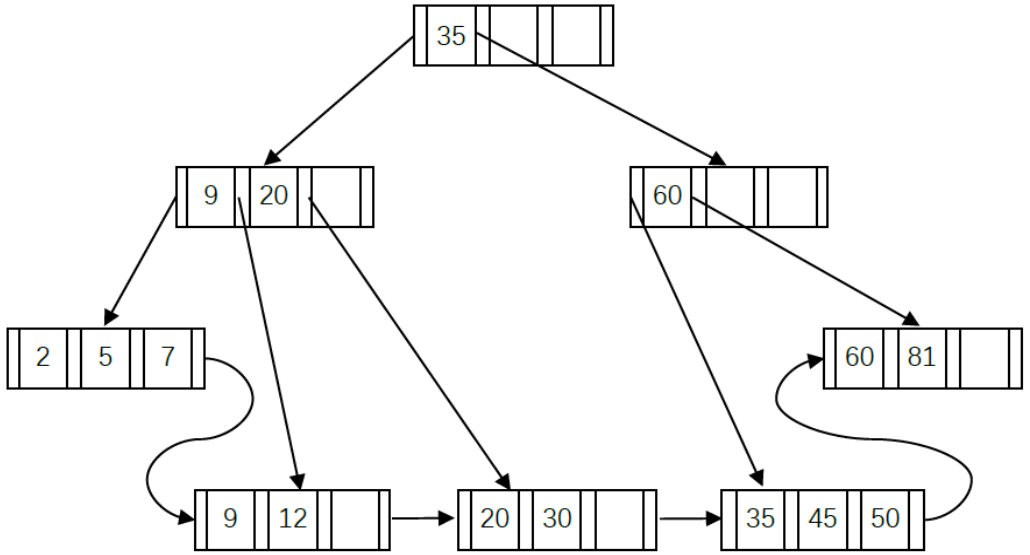
For the following B^+-tree (n = 4), answer following questions:

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.

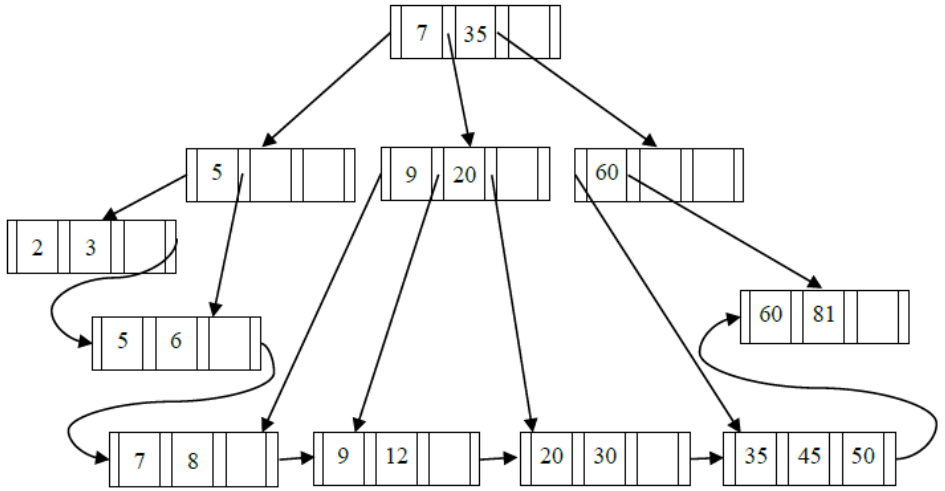
3) If the height of tree is 5, please tell the minimal and maximal numbers of key values in the tree.

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.



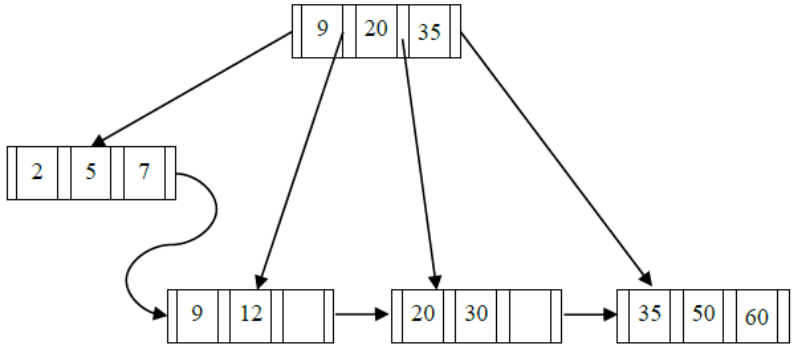
1)

After inserting 8, 6 and 3:



2)

After deleting 81 and 45:



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

Minimal number of key values: 2*2*2*2*2=32

4) (3 + 1) + 1 = 5 或 (3 + 1) + 2 = 6

Query Processing

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 ⋈ 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) ⋈ 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).

1) 5,000/500/5 = 2

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

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

Query Processing

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))

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.)

1) (10000000*10000000) / (100 * 365)) *3/12 = 684.93M
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
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
5) Cost for evaluating **σ operation** **（2分，各1分）**
   block transfer = 62 t_T
   seek time =1 t_S

Query Processing

Consider the following relational schema and SQL query:

account(account_no: char(10), customer_name: char(10),
branch: char(20), balance: integer)

access (serial char(10), account_no: char(10), access_date: date, amount: integer)

note: account_no of table ***access*** *references to table* ***account****.*

```
select  account.account_no, account.customer_name
from    account , access
where   account.account_no = access.account_no and year(access_date)=2012
        and branch='Hangzhou' and amount between 10000 and 89999.
```

1) Estimate the size (i.e. number of records) returned by above SQL statement.
2) Estimate the cost, in the best case, for evaluating **account** ⋈ **access** with Block Nested-Loop Join method.

Assumptions:

- account ***table*** has 10,000 records; ***access*** table has 100,000 records
- ***branch*** attribute has 50 distinct values.
- The value of a***ccess_date*** attribute is between '2011-01-01' and '2014-12-31'
- The value of ***amount*** attribute is between 1 and 1,000,000.
- The values of all attributes are uniformly distributed.
- The integer type needs 4 bytes.
- The date type needs 4 bytes.
- The file system supports 4K byte blocks.
- There are 100 buffer blocks available in memory for evaluating the join operation.

1) 100000*1/4*1/50*80000/1000000=100000*1/4*1/50*8/100=40

2) Size of account record=44

Number of account record per block=4096/44 = 93

Number of blocks of account= 10000/93=108

Size of access record=28

Number of account record per block=4096/28 = 146

Number of blocks of account= 100000/146=685

br/M-2*bs+br = 108/98*685+ 108 = 2*685+108 =1478 block transfers

2*br/(M-2) seeks = 2*108/98 = 4 seeks

Concurrency Control

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.

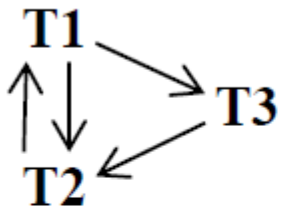
T1	T2	T3
Read(A)		
Read(B)		
		Read(A)
		Write(A)
	Read(B)	
	Read(A)	
Write(B)		
	Write(A)	

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).

1)



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

2) The schedule is not cascadeless.

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

Recovery

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?

1) 1002

2) 1010

3) T4

4) “102.1” = 62, “102.2” = 73

5)

1015: <T4, 102.1, 62>

1016: <T4, abort>

Aries recovery method is widely used in industrial DBMSs. Please answer following questions about ***Aries***.

1) What contents are in the checkpoint log of ***Aries***?
2) Why checkpoint operation of ***Aries*** puts less side effects on normal transaction processing of DBMS?

Aries recovery method is widely used in industrial DBMSs. Please answer following questions about ***Aries***.

1) What contents are in the checkpoint log of ***Aries***?
   Dirty Page Table
   Active Transaction List
2) Why checkpoint operation of ***Aries*** puts less side effects on normal transaction processing of DBMS?
   Doesn't output dirty pages in buffer to disk during checkpointing.

Good Luck!