DATA MANAGEMENT

Homework # 5

A. Data Structures (1 + 1 pt)

1.

(a) (b) Deleting takes between 1 and N operations

If the heap needs to be "compacted" it will take allways N operations (first to reach the record, then to move the "tail" to close the gap)

Also if the "null" value can be put instead of a "real" value, then it will cause the heap to grow unnecessarily.

If the heap doesn't need to be compacted, then it would take 1 operation.

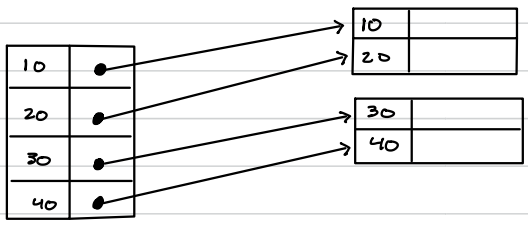
2. For sorted sequence, it usually take between logN and log N+N operations. However, it would take log N+N operations depending on the variant. For instance, if a "null" value can be put instead of the real value, then it would cause the sequence to grow unnecessarily.

B. Index and File (1 + 1 + 1 pt)

1.

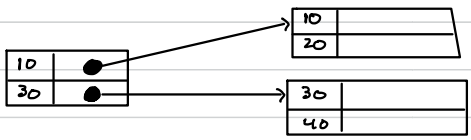
(a) An index is dense when it has index entries for every search key value in the database file. This helps to search faster, however it needs more space to store index records.

Example:



In the above indexing, the method records contains search key value and then points to the real record on the disk.

(b) As compare to Dense Index, Sparse Index appears for only some of the values in the file. Hence, due to this it is much slower to locate the records. However, as compare to dense indexing is that it has small size of the index, and also maintaing the index would be decreased.



C. B+ trees

Disk containing 2^{35} bytes.

Disk is organized into blocks each containing 2^{9} bytes

ID: 14 bytes

Salary: 7 bytes

DateOfBirth: 9 bytes

Comment: 8 bytes

2^{35} bytes on the disk, each holding $2^{9} = 512$ bytes

$\therefore$ there are 2^{38} blocks, hence a block can be specified in 38 bits

Allocating 4 bytes to block address

$(m) \times (\text{size of pointer}) + (m-1) \times (\text{size of key}) \leq \text{size of the block}$

$(m) \times (4) + (m-1) \times (14) \leq 512$

$4m + (m-1)14 \leq 512$

$4m + 14m - 14 \leq 512$

$19m - 14 \leq 512$

$19m \leq 526$

$m \leq 27.684$

$m = 27$

$\therefore$ the root will have between 2 and 27 children

$\therefore$ the internal node will have between 14 and 27 children

$(m) \times (\text{size of pointer}) + (m-1) \times (\text{size of key}) \leq \text{size of the block}$

$(m) \times (4) + (m-1)(14) \leq 512$

$4m + 14m - 14 \leq 512$

$18m - 14 \leq 512$

$18m \leq 526$

$m \leq 29.22$

$m = 29$

$\therefore$ a block address can be between 2 and 29

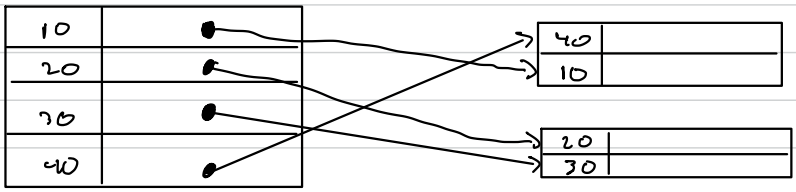
$\therefore$ an internal node will have between 15 and 29 children.

Between 2 and 29 pointers come out of the root

Between 15 and 29 pointers come out of a non-root.

In the narrow tree we would like to contain only 15 pointers, therefore we will have

Level	Nodes in a Narrow tree $\left\lceil \frac{29}{2} \right\rceil$		Nodes in a wide tree
1	1		1
2	2	2·15	29
3	30	30·15	841
4	450		24389



Homework 06

A history is recoverable if for every transmission T that commits, the Commit of T follows the commit of every transaction which T read.

T1 R x ⟶ T1 reads item x

Money Transfer (Example from slides)

xa	xb	a	b	log
?	?	8	1	
?	?	8	1	[T s]
8	?	8	1	[T s]
3	?	8	1	[T s]
3	?	8	1	[T s] [T a 8 3]
3	?	3	1	[T s] [T a 8 3]
3	1	3	1	[T s] [T a 8 3]
3	6	3	1	[T s] [T a 8 3]
3	6	3	1	[T s] [T a 8 3] [T b 1 6]
3	6	3	6	[T s] [T a 8 3] [T b 1 6]
3	6	3	6	[T s] [T a 8 3] [T b 1 6] [T c]

Recovery with Checkpointing

- For each transaction for which you have a commit record, add it to the redo list.
- For each transaction for which you have a start record but not a commit record, add it to the undo list.
- For each transaction that is listed in the checkpoint record for which there is no commit record, add it to the undo list.

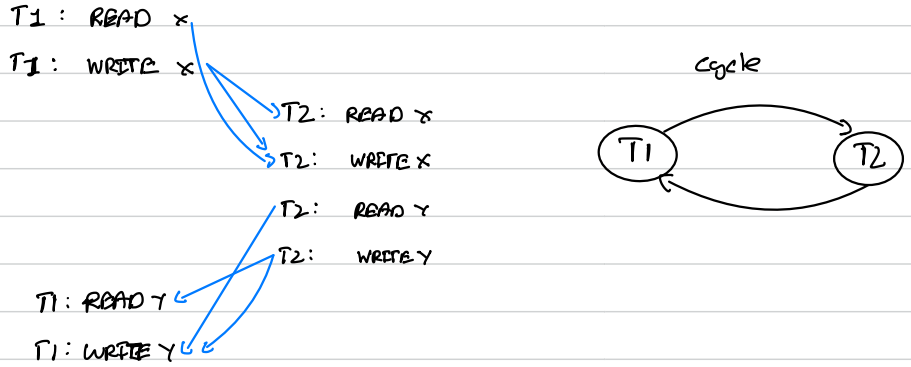
Serializable Histories

- Each serializable history describes a correct execution.
- So we really have algorithms that partition histories into two classes.
  - Serializable
  - Perphaps not serializable
- We will partition histories into two classes:
  - Conflict serializable (guaranteed to be serializable)
  - Not conflict serializable (we do not know whether they are serializable or not)

Conflict Graphs

- Conflict Graph is used to decide whether a history:
  - Is conflict serializable, or
  - Is not conflict serializable.

- Example :



- The conflict graph for a serial history does not have cycles !

1.

UNDO LIST : [T5]

REDO LIST : [T6, T3]

NOT AFFECTED LIST : [T1, T2, T4]

2.

(a) T5 : READ x

T1 : READ x

T2 : WRITE x

T1 : WRITE Y

T3 : WRITE Y

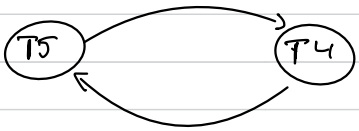
T4 : WRITE x

T4 : WRITE Y

T4 : WRITE Z

T5 : READ Z

T5 : WRITE Z



Cycles, hence not a serialized

(b) T2 : READ A

T1 : READ A

T2 : WRITE B

T1 : WRITE B

T1 : WRITE C

T2 : WRITE C

T3 : READ A

T4 : WRITE B

T4 : WRITE C

T4 : WRITE D

T5 : WRITE E

This is serializable Histories, since there is no cycles in it. Hence, this will be equal to the following serial history:

[T1, T2, T3, T4, T5]

4.

REDO LIST = (T2)

UNDO LIST = [T4, T3]

(a) a = 1, 2

b = 1, 2, 3

c = 0, 1

(b) c = 0 ∴ a = 2

b = 2 b = 2

a = 2 c = 0

Final Exam Prep

UNIT 4 EXAMPLES:

1. Produce the relation Answer(A) consisting of all ages of people.

```
SELECT A
FROM Person
```

2. Produce the relation Answer(N) consisting of all women who are less or equal than 32 years old.

```
SELECT N
FROM Person
WHERE S='F' AND A <= 32
```

3. Produce a relation Answer(P, Daughter) with the obvious meaning

```
SELECT P, C AS Daughter
FROM Person, Birth
WHERE C=N AND S='F'
```

4 Produce a relation Answer(Father, Daughter) with the obvious meaning.

```
SELECT N AS Father, C AS Daughter
FROM Person, Birth, Person AS Person1
WHERE P = Person.N AND C = Person1.N
      AND Person.S='M' AND Person.S='F'
```

5. Produce a relation. Answer(Father_in_law, Son_in_law)

```
SELECT N AS Father_in_law, C AS DAUGHTER
FROM Person, Birth, Person AS Person1
WHERE P = Person.N AND C = Person1.N
      AND Person.S='M' AND Person.S='F'
```

HOMEWORK #4

1.

(a)

i $AB^+ = ABC$

ii $AEG^+ = AEGHDF$

iii $EG^+ = EGDF$

iv $ABCD^+ = ABCDEGDFH$

v $AE^+ = AEH$

(b) ABD , ABG

2.

(a)

Us → AlTi

Ti → Ye Da

Al → Le Re

Us Re → Pr

Da → Ye

(b)

Us → AlTi Pr

Ti → Ye Da

Al → LeRe

Da → Ye

→

Us → AlTi Pr

Ti → Da

Al → LeRe

Da → Ye

(c) i Us AlTi Pr

Ti Da

Al Le Re

Da Ye

ii No you can't

iii Trying to find :

Us AlTi Le Re Pr Ye Da

Us^+ ↓

3.

(a)

CF → E

AE → C

F → D

E → F

B → F

Both sides : E , C, F

Left side : B , A

Right side : D

Every key must contain : A, B

$AB^+ = ABFD$

ABC

(b)

CF → E = key

AE → C = Partial

F → D = transitive

E → F = Partial

B → F = Partial

4.

ABFH → DE

E → D

H → F

A → G

G → FH

→

AB → E

E → D

H → F

A → G

G → H

SQL

1.
```
SELECT  CName
FROM    Customer
WHERE   Ccity = 'Boston'
```

2.
```
SELECT *
FROM   Customer
WHERE  Ccity = 'Boston'
```

3.
```
SELECT  Pname
FROM    Plant
WHERE   Pcity = 'Boston'
```

4.
```
SELECT  C
FROM    Customer , Plant
WHERE   Customer.P = Plant.P  AND  Ccity = Pcity
AND     Profit >= 50 000
```

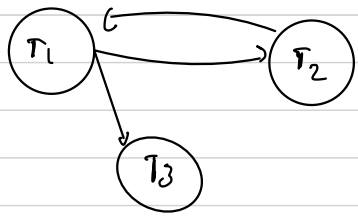
5.
```
SELECT  First.P AS Bigger , Second.P AS Smaller
FROM    Plant AS First , Plant AS Second
```

6.

(a)

No it isn't serializable

(b)



7.

(b) $\frac{6000}{3} \times \frac{8000}{4} = 2000 \times 2000 = 4,000,000$

1 Read of R + 3000 Read of S $= 4,000,000 + 6000$
$= 4,006,000$

1 Read of S + 4000 read of R $= 4,000,000 + 8000$
$= 4,008,000$

The most efficient way is $4,006,000$

(c) $\frac{400}{6} \times \frac{160}{4} = 66.6 \cdot 40 = 2666$

12000 Read of R is 200 read of S = 26