$\boxed{\#1}$ $R(A\ B\ C\ D\ E\ F)$

$AC \longrightarrow B$

$F \longrightarrow C$

$BD \longrightarrow F$.

$\{AC\}^+ = \{ABC\} \neq \{ABCDEF\}$

$\{F\}^+ = \{CF\} \neq \{ABCDEF\}$

$\{BD\}^+ = \{BDF\}^+ = \{BCDF\} \neq \{ABCDEF\}$

Decomposition:

using $AC \rightarrow B$: $R(A\,\underset{x}{B}\,CDEF) \Rightarrow R1(ABC)$ $R2(ACDEF)$

using $F \rightarrow C$: $R2(A\underset{x}{C}DEF) \Rightarrow R3(CF)$ $R4(ADEF)$

using $BD \rightarrow F$: Nothing to Decompose.

Thus, the final BCNF tables are:

$$\boxed{R1(ABC) \quad R2(CF) \quad R4(ADEF)}$$

$\boxed{\#2\text{-}1}$ R(ABCDEFGH)

A → B How many subset are closed?

B → CD

$\{A\}^+ = \{A\}^+$

$= \{AB\}^+$

$= \{ABC\}^+$

$= \{ABD\}^+$

$= \{ABCD\}$ closed

$\{B\}^+ = \{B\}^+$

$= \{BC\}^+$

$= \{BD\}^+$

$= \{BCD\}$ closed

subset open whe B present.

and CD does not present.

subset open when A present; and

B C D does not present

⇒ Whether the subset is closed depends on how A B C D are present in the subset.

present.	A	B.	C	D	closed?
	0	0	0	0	√
	0	0	0	1	√
	0	0	1	0	√
	0	0	1	1	√
	0	1	0	0	X
	0	1	0	1	X
	0	1	1	0	X
	0	1	1	1	√
	1	0	0	0	X
	1	0	0	1	X
	1	0	1	0	X
	1	0	1	1	X
	1	1	0	0	X
	1	1	0	1	X
	1	1	1	0	X
	1	1	1	1	√

of "√" = 6 $\quad (6)(1^4)(2^4) = 96$ — closed

of "x" = 10 $\quad (10)(1^4)(2^4) = 160$ — open

$\boxed{\text{Thus 96 of them are closed.}}$

$\boxed{\#2\text{-}2}$ Relation $R(A, B, C)$

$F \subseteq G$.

$G = \{\underline{A \to B} \quad B \to A. \quad \underline{A \to C} \quad C \to A. \quad \underline{B \to C}. \quad C \to B\}$

Direct imply : $A \to B \Rightarrow \{A \to B\}^+$

indirect imply : $A \to C \to B \Rightarrow \{A \to C, C \to B\}^+$

$\Rightarrow$ Whether the subset imply $A \to B$ depends on how $A \to B$. $A \to C$, and $C \to B$ are present in F.

present $A \to B$	$A \to C$	$C \to B$	Imply?
0	0	0	X
0	0	1	X
0	1	0	X
0	1	1	V
1	0	0	V
1	0	1	V
1	1	0	V
1	1	1	V

It implies $A \to B$ when $A \to B$ present of both $A \to C$ and $C \to B$ present.

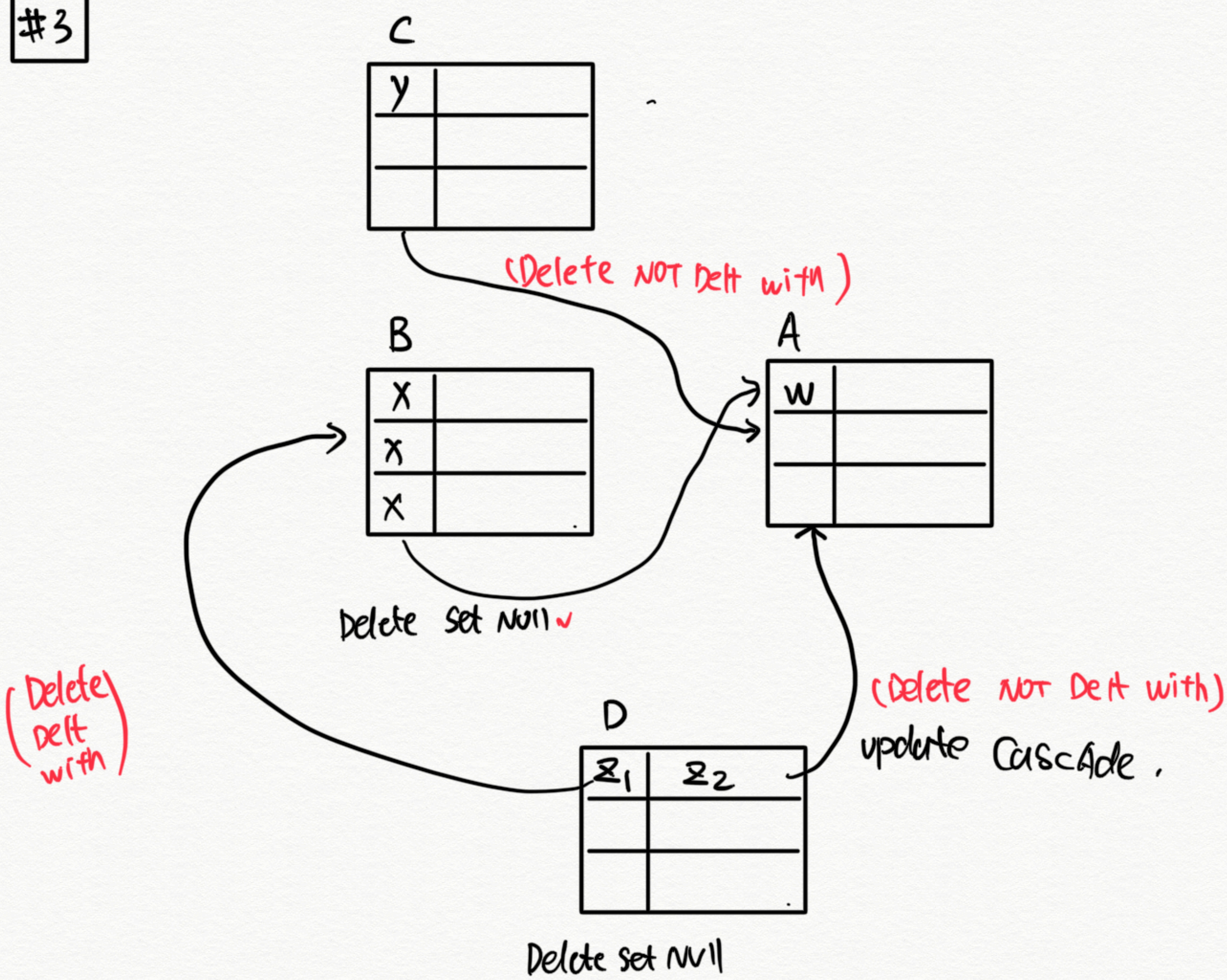
$\#$ of "V" $= 5 \; (10)(1^3)(2^3) = 40$

$\#$ of "X" $= 3 \; (6)(1^3)(2^3) = 24$

$\boxed{\text{Thus } 40 \text{ subsets of } G \text{ logically imply } A \to B.}$

Final. Q3.

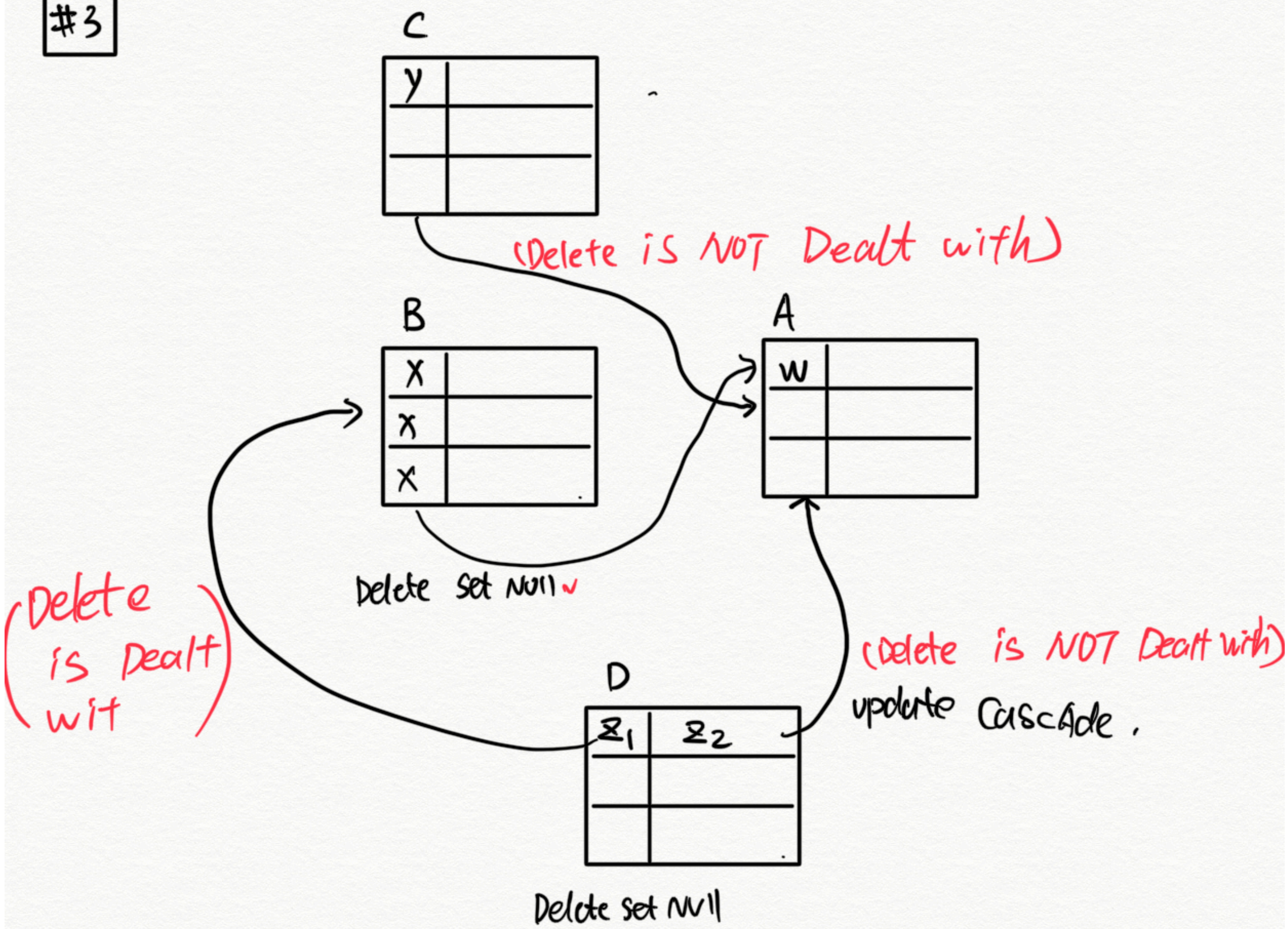
$\boxed{\#3}$



	Referencing	Referenced	
	Insert	~~Insert~~	× No violate
No violate ×	~~Delete~~	Delete	
	Update	Update	

Final. Q3.

$\boxed{\#3}$



	Referencing	Referenced	
	Insert	~~Insert~~	× No violate
No violate ×	~~Delete~~	Delete	
	Update	Update	

#q3:

Since all modifications we are dealing with are deletions, RI violation will occur only when we delete from a referenced table. A and B are referenced table. Let's analyze them separately.

Deleting from B won't cause violate since the dangling tuples in B have been dealt with by the "ON DELETE SET NULL" statement.

Table B, C, and D are all referencing to table A. Table B's reference to Table A won't cause a RI violation since it's been dealt with by the "REFERENCES A(w) ON DELETE SET NULL" statement. However. The reference from table C to table A and from table D to table A might cause RI violation since the dangling tuples in C and D are not dealt with after deleting from A.

Thus, "DELETE FROM C" must happen before "DELETE FROM A", and "DELETE FROM D" must happen before "DELETE FROM A".

Thus, II and III works, I doesn't work.

Final q4

$\boxed{\#4\text{-}1}$

	Insert	Deletion	Update
Executive	Yes		Yes
Employee	Yes	Yes	Yes

```sql
-- q4-2

CREATE TABLE Executive(
    ...
    CHECK(salary <= 10 * (SELECT MIN(e.salary)
                          FROM Employee e)
                          WHERE e.division = div_in_charge),
    ...
);
```

```sql
-- q5

SELECT * FROM Books e where e.publisher IN (
    SELECT publisher FROM Books
    WHERE birthYear > 1950

    GROUP BY publisher
    HAVING SUM(price) > 10000
)
```

Assume a disk of the following characteristics for this problem:

1TB total disk capacity
10 surfaces
1000 tracks per surface
6000 RPM   6000 rotation / minute => 10ms/rotation
10ms average seek time
4KB block size (1KB is 1024 bytes)

1. Suppose that we are reading a file F that occupies exactly one entire track, and we want to estimate how long it takes to read the whole file sequentially. Are there any key parameters that are missing for doing this? Select one from the following choices:

(a) the diameter of the disk surface — d

(b) the average rotational latency — r

(c) the transfer rate of the disk — t

(d) none

Choose (d) none.

Time = avg seek time + avg rotational delay + transfer time
= 10ms + .5 *10ms + 1 rotation * 10ms/rotation
= 15ms + 10 ms
= 25ms

All parameterers are given, choose d

Based on your choice above, how long does it take to read the file on average? You may use the symbol (after each choice) to represent the parameter that you think is useful in the estimation result. Assume that the disk head may not initially be on top of the track where the file is located. (5 points)

Time = avg seek time + avg rotational delay + transfer time

= 10ms + .5 *10ms + 1 rotation * 10ms/rotation

= 15ms + 10 ms

= **25ms**

It take **25ms** on average to read the file.

1TB total disk capacity
10 surfaces
1000 tracks per surface
6000 RPM 6000 rotation / minute => 10ms/rotation
10ms average seek time
4KB block size (1KB is 1024 bytes) 4094 bytes/block

2. You need to store the following table in the provided disk:

Student(id LONG, name CHAR(32))
(id: 8bytes, name: 32 bytes) —> total 40 bytes for one tuple

LONGs are 8 bytes and CHAR(32)'s are 32 bytes. Assume that tuples are not spanned across blocks. The table has 1 million tuples and is sequenced by the id attribute. What is the minimum number of blocks needed to store this table? (5 points)

4094 bytes/block / 40bytes/tuples = 102.4 tuple/block = 102 tuple/block

1M tuples / 102tuple/block = 9803.9 block = **9804 block**

3. We decide to create a B+tree index on the attribute id of the Student table. The pointers in the B+tree are of size 8 bytes. One node of the B+tree corresponds to one disk block. What is the minimum number of nodes that are needed for the B+tree? (10 points)

Block size = 4094 bytes
Assume we use n-degree B+ tree, and we have m level of the tree.

4094B node, 8B key, 8B ptr
(8)(n) + (8)(n-1) <= 4094
n <= 256.3
n = 256 degree

\# of key in leaf node = 255
\# of ptr in node = 256

In index B+ tree:
Level m: 1M key / 255 = 3921.5 = 3922 nodes
Level m-1: 3922 / 256 = 15.3 = 16 nodes
Level m-2: This is root, and only one node is needed to point to 16 children.
Total # of nodes = 3922 + 16 + 1 = **3939 nodes**

Thus , we need 3939 nodes.

4. Using the constructed B+tree, how many disk IOs are needed to execute the following

query?

SELECT * FROM Student WHERE id = 334234
Note that the student id is a key of the table. Assume that the root node of the B+tree is always cached in main memory and does not require a disk IO to read it. No other B+Tree nodes or the Student table blocks are cached in main memory. (5 points)

Index of 1M records are stored in the 3939-nodes B+ tree.

Since the B+ tree has three levels according to q3, and root is always in the cache, we need two I/O to find out the index entry where the key, id, equals to 334234. And we need one more I/O to read the complete content of the tuple.

Thus, we need 3 I/O to execute this query.

#9.7

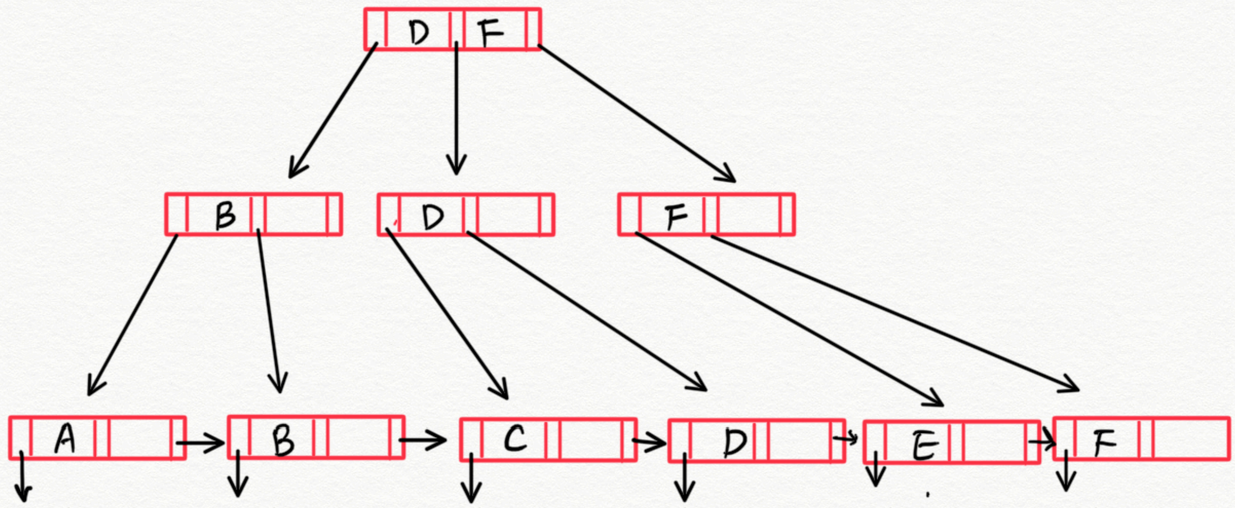
Space gurantee:

Non leaf Node: $\lceil 3/2 \rceil = 2$ ptr

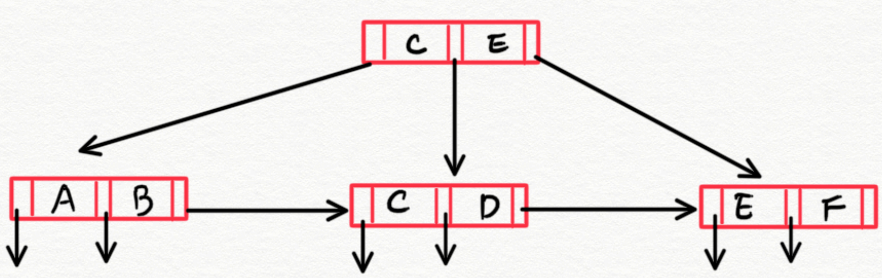
$\lceil (3+1)/2 \rceil = 2$ ptr.

at least

Height-3 Tree:



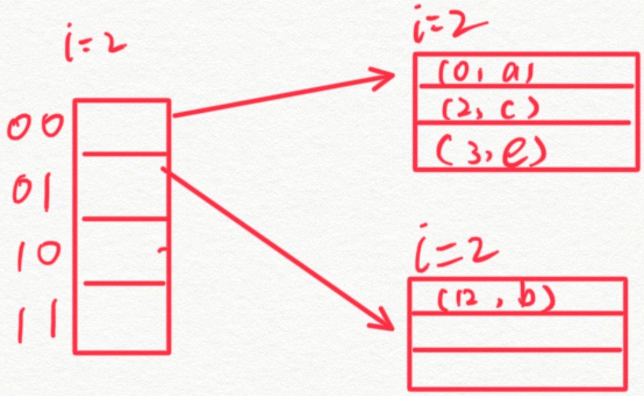
Height-2 Tree:



```
(0, a)  ==> h(0) = 0000
(12,b)  ==> h(12)= 0100
(2, c)  ==> h(2) = 0000
(3, e)  ==> h(3) = 0001
(4, f)  ==> h(4) = 0000
```

#8

After inserting (3, e):



After inserting (3, e), (4, f)