Page No.
Date

DBMS

— 8 to 10 marks

(25-30) marks

1) SQL & Relational algebra
2) Tuple calculus & Domain calculus
3) Key concept, functional dependency and normalization
4) Transaction & Serializability
5) Hashing & Indexing (file structure)
6) E-R Diagram

1) Key Concept FD & Normelization

(Relation: Student, Attribute)

	Rollno	Name	Phone no	Subject	marks
Primary key	01	xyz	987	DBMS	90%
	02	xyz	258	OS	87%

Name → Phone no. ✗

Functional dependency:
- Rollno → name ✓
- 1 value (L.H.S) → Unique value R.H.S

Rollno	Name	CNo.	CName	grade
01	xyz	01	DBMS	A
01	xyz	102	OS	A+
02	ABC	101	DBMS	B
03	wxy	102	OS	C
03	wxy	103	TOC	A

Rollno → Name ✓ e.g. FD

C.NO → C.Name ✓

Rollno → CNO ✗

Rollno CNO → grade ✓

Rollno Cname → grade

Primary key : { Rollno Cno }
{ Rollno Cname }

they both are candidate for that relation

Note: all Primary key are Candidate key, but all Candidate key is not Primary key.

* R(A, B, C, D)

A → B
B → C
C → D

A	B	C	D
1	2	7	1
1	2	7	1
2	2	3	1
3	4	7	1

* Rules of functional depedency :-

(I) Reflexive

$x \rightarrow x$

is always true

for every attribute is reflexive.

Note:

$x \rightarrow y$ ~~then~~ $y \rightarrow x$

(II) Transitive :-

$x \rightarrow y$ and $y \rightarrow z$ then $x \rightarrow z$

iii) Projectivity (decomposition)

$x \rightarrow yz$ then $x \rightarrow y$, $x \rightarrow z$

iv) Union (additive)

$x \rightarrow y$, $x \rightarrow z$ then $x \rightarrow yz$

v) Augementation

$x \rightarrow y$ then $xz \rightarrow yz$

vi) Pseudo transitive:-

$x \rightarrow y$ and $yw \rightarrow z$ then $xw \rightarrow z$

★ Closure of A. $(\{A\}^+)$

$\{A\}^+ \rightarrow \{A, B, C, D\} \rightarrow$ C, K

$\{B\}^+ \rightarrow \{B, C, D\}$

$\{C\}^+ \rightarrow \{C, D\}$

$\{D\}^+ \rightarrow \{D\}$

$A \rightarrow B$
$B \rightarrow C$
$C \rightarrow D$

The key that derive all other things (attributes) is the candidate key here A is the candidate key.

2) R(A, B, C, D, E)

$A \to B$

$B \to C$

$D \to E$

$\{A\}^+ \to \{A, B, C\}$

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

$\{C\}^+ = \{C\}$

$\{D\}^+ = \{D, E\}$

$\{E\}^+ = \{E\}$

$\{A\ D\}^+ \to \{A, B, C, D, E\}$

↓

C.K

3) R(A, B, C, D, E, F, G)

$A \to D$

$D \to C$

$D \to E$

$BE \to F$

$F \to G$

$\{A\}^+ \Rightarrow \{A, B, C\}$

$\{B\}^+ \Rightarrow \{B, E\}$

$\{D\}^+ \to \{D, E\}$

$\{BE\}^+ \to \{BE, F\}$

$\{F\}^+ \to F, G$

A, D shows be in left side, so, it must be in candidate key. //

$\{AD\}^+ \to \{A, B, C, D, E, F, G\}$

So, AD is candidate key.

$\{ADF\}^+ = \{$

↓

Any extra attribute with candidate key is super key

Ans we can super ke

4) R(A, B,

{

$A \to B$

$B \to C$

$C \to D$

$D \to A$

$D \to E$

}

5) R(A, B, C

{ A →

A → B

DC →

D →

}

we can say that, Candidate key is a minimal super key.

4) R{A, B, C, D, E}?

{
A → B
B → C
C → D
D → A
D → E
}

Find C.K.

$\{A\}^+ \rightarrow \{A, B, C, D, A, E\}$ → C.K

$\{B\}^+ \rightarrow \{B, C, D, A, E\}$ → C.K

$\{C\}^+ \rightarrow \{C, D, A, E, B\}$ → C.K

$\{D\}^+ \rightarrow \{D, A, B, C, E\}$ → C.K

$\{E\}^+ \rightarrow \{E\}$

To Here, 4 candidate key, out of which any one can act as Primary key.

5) R(A, B, C, D, E)

{
A → B
AB → C
BC → D
D → E
}

How many S.K are there

Ans $\{A\}^+ \rightarrow \{A, C, B, D, E\}$ → C.K

A	ABC	ABCD
AB	ABD	ABCE
AC	ABE	ABDE
AD	ACD	ACDE
AE	ACE	ABCDE
	ADE	

There are 16 S.K.

6) R(A, B, C, D)
{
A → B
D → A
C → D
}
How many C.K and S.K?

Sol:
$\{A\}^+ \Rightarrow \{A, B\}$
$\{D\}^+ \Rightarrow \{D, A\}$
$\{C\} \Rightarrow \{C, D\}$
$\{D\} \Rightarrow \{D\}$

$\{AC\}^+ = \{A, B, C, D\}$ — C.K
$\{BC\}^+ \Rightarrow \{A, B, C, D\}$ — C.K
} 2 CK

AC
ACB
ACD
ACD
} 6 S.K

BC
BCD

7) R(A, B, C, D, F, G)
{
A → B
B → C
DE → G, F → G
}

Sol:
$\{A\}^+ \Rightarrow \{A, B, C\}$
$\{B\}^+ \Rightarrow \{B, C\}$
$\{D\}^+ \Rightarrow \{D\}$
$\{DE\}^+ \Rightarrow \{D, E, G\}$

E does not in relation
so discard this relation.

Discard this

8) R(A, B, C, D, F, G)
{
A → B
BC → D
DF → G
}

Sol:
{ADF}
ADFB
ADFC
ADFG

$\{AD\}^+ \Rightarrow \{A, B, C, D, F, G\}$

* Minimal or Canonical

1.) Simplify Functional should

{ADF}

ADFB
ADFC
ADFG

ADFBC
ADFBG
ADFCG

ADFBCG

8) R(A, B, C, D, E, F, G)?

{
A → B
BC → D
DF → G
}

Sol:

{A}+ → {A, B,

{A C G F}+ → {A, B, D, C, D, E, F, G}

or

{ACBF}+ = {A, B, C, D, E, F, G}?

* Minimal Cover of Functional dependency:-
(canonical cover)

1.) - Simplify the Functional dependency set, with the help of Functional dependency rule (R.H.S of every F.D should contain single attribute.

R (A, B, C, D, E, F, G)

{
A → BCD
B → C
A → D
A → E
DE → A

DE → B
DE → C
DE → F
DE → G

Find minimal cover of following F.D ?

Sol^n

$\{$

$A \to B$

$A \to C$

$A \to D$

$B \to C$

$A \to D$ ✗

$A \to E$

$DE \to A$

$DE \to D$

$DE \to C$

$DE \to F$

$DE \to G$

$\}$

II.) Find non-redundant set of Functional dependency by removing redundant functional dependency.

$\{$

$A \to D$ ✗

$A \to C$ ✗

$A \to D$ ✓

$D \to C$ ✓

$A \to E$ ✓

$DE \to A$ ✓

$DE \to D$ ✓

$DE \to C$ ✗

$DE \to F$ ✓

$DE \to G$ ✓

$\}$

$\{A\}^+ = \{A, C, D, E\}$

$\{A\}^+ = \{A, C, D, E\}$

$\{DE\}^+ \to \{D, E, A\}$

$\{$

Date

Page No.
Date

{

$A \to D$

$A \to E$

$B \to C$

$DE \to A$ ⟶ $AE \to A$

$DE \to B$

$DE \to F$

$DE \to G$

}

This is minimal cover

Note:

If we change the order, then

{

$DE \to A$

$DE \to BX$

$DE \to CX$

$DE \to E$ ✓

$DE \to G$ ✓

$A \to D$ ✓

$A \to C$ ✗

$A \to D$ ✓

$D \to C$ ✓

$A \to D$ ✗

$A \to E$ ✓

}

This is also minimal one

means, if we change the order, then the minimal cover may be changed.

a) Consider the following dependency set, and find minimal cover,

T1

Page No.
Date

$A \rightarrow B$

$ABCD \rightarrow E$

$EF \rightarrow G$

$EF \rightarrow H$

$ACDF \rightarrow E$

$ACDF \rightarrow G$

Sol'n

Pseudo rule

$A \rightarrow B$

$ABCD \rightarrow E$

$EF \rightarrow G$

$EF \rightarrow H$

$ACDF \rightarrow E$ ✗

$ACDF \rightarrow G$ ✗

11 c) Consider the following relation

$R(A, B, C, D, E, G, H)$

$AB \rightarrow C$

$A \rightarrow BD$

$AD \rightarrow E$

$D \rightarrow DE$

$E \rightarrow G$

which of the following is minimal cover

Pseudo transi

a) $\{A \rightarrow B, AB \rightarrow C, B \rightarrow D, BD \rightarrow E, E \rightarrow G\}$

b) $\{A \rightarrow B, A \rightarrow C, B \rightarrow D, B \rightarrow E, E \rightarrow G\}$

c) $\{A \rightarrow D, B \rightarrow D, D \rightarrow E, E \rightarrow G\}$

d) $\{A \rightarrow B, A \rightarrow D, A \rightarrow C, D \rightarrow D, D \rightarrow E, E \rightarrow G\}$

Sol'n

Page No.
Date

a) Given the functional depedency

$AB \rightarrow C$

$AC \rightarrow D$

Which of the following also holds for the relation

i) $A \rightarrow B$

ii) $AB \rightarrow D$

iii) $AB \rightarrow D$

iv) $AC \rightarrow B$

v) $ABC \rightarrow D$

a) i only

b) II & III only

c) I & III & IV

d) II & III & V

Soln

$(AB)^+ = \{A, B, C, D\}$

$(Ac)^+ = \{A, C, D\}$

$(ABC)^+ = \{A, B, C, D\}$

b) Consider the following set of depedency,

$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H\}$

$G = \{A \rightarrow CD, E \rightarrow AH\}$

a) $F \equiv G$

b) $F \geq G$

c) $F \leq G$

d) F & G are not equivalent

Soln

$F \geq \{A \rightarrow C, AC \rightarrow D\}$

$A \rightarrow C$

$A \rightarrow D$

$E \rightarrow A$

$E \rightarrow D$

$E \rightarrow AD$, $E \rightarrow H$

$E \rightarrow A$

$E \rightarrow D$

$A \to C$
$A \to D$
$A \to CD$

$E \to A$
$E \to H$
$E \to AH$

★ Normalization!

- Problem:-
  1) Insertion
  2) Deletion
  3) Updation
  4) Data Redudancy

-
  1) 1NF
  2) 2NF
  3) 3NF
  4) BCNF (Boyce-code normal form)
  5) 4NF
  6) 5NF (Not in gate)

1) 1NF

Rollno	Name	Phon no (mobile no, class no. office no
01	xyz	987 237 478

2) 2NF:

a) A relation is said to be in 2NF if all non-prime attribute are fully functionally depends on the primary key of relation or

R(A, B, C, D, E) → Non Prime attribute

→ [AB]+ = C, K

A & B are Prime attribute

Composite relation

b) A rel is said to be in 2NF, if there does not exist any partial functional dependency.

R (A, B, C, D, E)

Prime A. → non P. A (PFD)

2NF ✗

Not in 2NF

Primary key → Non-Prime attribute (FFd)

↓

fully functional dependency ✓

Yes, 2NF ✓

Page No.
Date

Prime attribute → Prime attribute (Not Partial dependency) 2NF ✓

NPA → NPA (PFd) 2NF ✓

P.A → P.A & NPA (PFd) Not in 2NF ✗

P.A
↓
Primary attribute

1) * Student (Rollno, name, Cno, Cname, grade)
{
Roll no → name
C.No → Cname
RollnoCNo → grade
}

soln

Roll_no, name, Cno, Cname, grade

{Roll no, CNo}

P.A → N.P.A
Roll no → name (PFA)
2NF ✗

CNo → Cname
P.A → N.P.A
Not in 2NF.

2) Student (Rollno, name, dob, age)
{
Rollno → name ✓
name → Rollno ✓
dob → age ✗
}

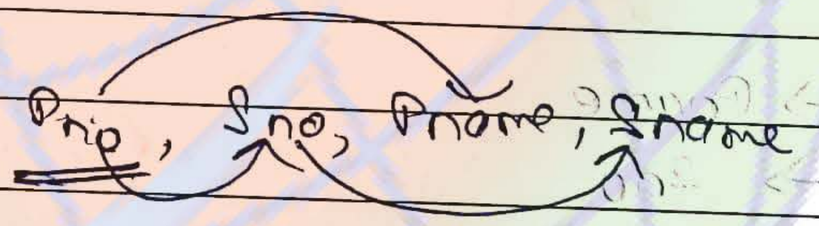
Page No.
Date

Rollno, name, dob, age

C.K : {Rollno, dob}
{name, dob}

Not in 2NF

3) R(Pno, Sno, Pname, Sname)
{
Pno → Pname
Pno → Sno
Sno → Sname
}



✓ 2NF

Note: • If relation contain zero partial attribute, then that relation is always in primary key.

• If key is not composite, then relation is always in 2NF.

☆ 3NF

A relation is said to be in 3NF, if all non-prime attribute, non-transitively depends on the Primary key (candidate key, or super key) of the relation

Date ____

(A → B, B → C) ✗
A → C

parameter to check in 3NF or not

- If α → β is in 3NF, then either (i) α is super key or (ii) β is prime attribute.

α → β
(α: Primary / Cand. key / super key)

* R (Pno, Pname, Sno, Sname)
{
Pno → Pname
Pno → Sno
Sno → Sname
}

Pno, Pname, Sno, Sname

(Sno → Not primary key; Sname → It should be in prime attribute ✗)

Is in 2NF but not in 3NF.

- Note
If R → is in 3NF then → it will also be 2NF and 1NF

Note:
If relation is in 3NF then insertion, deletion, updation anamalies does not exist but data redudency exist in the relation.
So, all upper normal forms are used minimize data redudency problem.

★ BCNF:

BCNF is a stronger form of 3NF, here determinant should be super key.

If $\alpha \rightarrow \beta$

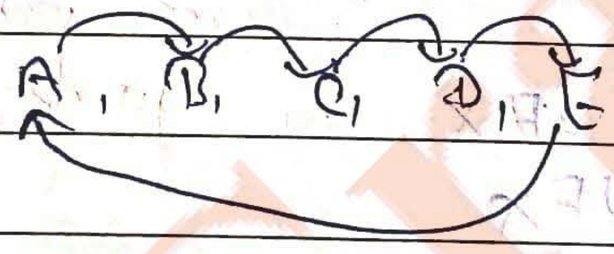
then, it should be "super key"

Q) Consider,

R(A, B, C, D, E)

{
A → B
B → C
C → D
D → E
E → A
}

find highest normal form of the relation



So, relation is in BCNF

Note:
If relation contain n-attribute and all are key then relation is in BCNF.

2) R(A,B)

Sol^n

Possible relation	R(A,B)	R(A,B)	R(A,B)
R{A,B}	}	{	{
{	D → A	A → B	AB
A → B	}	B → A	{
}		}	}

So

A relation with two attribute, will always be in BCNF.

3) R(A, B, C, D, E, F, G)

{

AB → C

DA → E

F → G

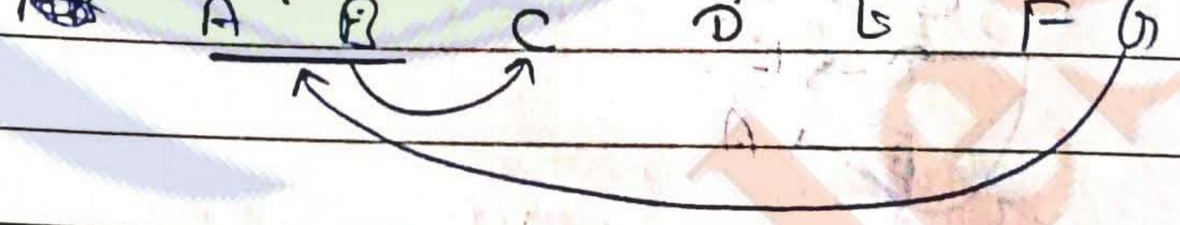
G → AB

}

Not in BCNF

Find highest normal form

Sol^n



{DF}+ = {D, E, F, G, A, B}

(1) BCNF ✗

3NF ✗

2NF ✗

1NF ✓

4) R(A, B, C,

{

AB

B

{ D

Sol^n

5) R(A

{

6) R(

Sol^n

4) R(A, B, C, D, E)

{

AB → DE

B → C

{ PA N.P.A

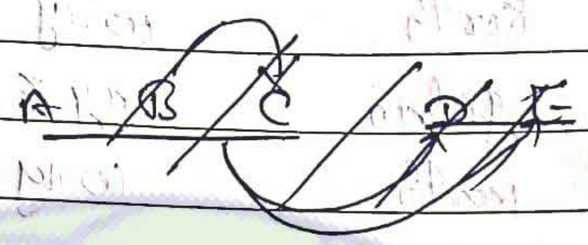
Find highest normal form.

BCNF ✓

2NF ✗

1NF ✓

Soln



C.K = {AB}

{AB}+ = {AB, C, D, E}

2NF ✗

1NF ✓

5) R(A, B, C, D, E)

{

ABC → DE BCNF

B → D Not in 2NF

P.A N.P.A

1NF ✓

6) R(P, C, h, q, d, t)

PC → hadt BCNF ✓

h → q 3NF ✗

NPA NPA

Find highest normal form

Soln

PC hadt

{PC}+ → {P, C, h, q, d, t}

so, the relation is in 2NF.

Page No.
Date

a)

Cname	Auth	Inst
DBMS	Korth	xyz
DBMS	Navath	xyz
DBMS	Korth	cony
OS	galvin	xyz
OS	Korth	cony

Find F.D for that?

Sol^n

Cname → Auth ✗

Cname → Inst ✗

Auth → Inst ✗

Cname Auth → Inst ✓

Cname Ins → Auth ✓

No FD exist but exist multivalue dependency (→→)

- If all the attribut

Note:

- If relation is zero functinal depedency, then it is always in BCNF

Note multivalue depedency

Cname →→ Auth

↓ ↓

1 value more then one in R.H.S

L.H.S

Cname →→ Inst

Auth →→ Inst .

Page No.
Date

4NF:

A relation is in 4NF, if multivalued dependency
i. either
(i) trivial dependency
or
(ii) If non-trivial ($\alpha \rightarrow\rightarrow \beta$) then α must be super key.

* (i) trivial dependency :-

$\alpha \rightarrow\rightarrow \beta$ is trivial if (i) $\beta \subseteq \alpha$

or

(ii) $\alpha \cup \beta = R$ (gives all the attributes of relation.)

$AB \rightarrow\rightarrow B$ ✓
$R(A,B,C)$
$A \rightarrow\rightarrow BC$ ✓

#		
	$AB \rightarrow\rightarrow B$	$R(A,B,C)$
	$AB \rightarrow\rightarrow B$ ✓	$A \rightarrow\rightarrow BC$

Functional dependencies are generated at the time of creating structure of the table, while multivalue dependency generated at the time of insertion of the data.
that means multivalu dependency are tuple generated

Decomposition of relation:-

Decomposition in 2NF:-

Q1) R(A, B, C, D, F)

{
AB → C
B → D.
C → D.
D → F
}

i) Find all candidate keys of relation
ii) make new relation with the attributes of the partial func. dependency

R(A, B, C, D, F)

R(A, B, C, D)

R(B, D, F)

Q2) R(A, B, C, D, E)

Σ

B → C R(A, B, C)

D → D R1(D, C, E)

D → E

Q3) R(A, B, C, D, E, F, G, H, I, J)

{

AB → C {AB}+ {AD}+ = {A, B, C, D, E, F, G, H, I, J}

A → DE

B → F Not in 2NF

F → GH

D → IJ

}

Decompose

R(A, B, C)

R1(A, D, E)

A → DE

B → F

R(A, B, C)

R1(A, D, G, I, J)

R2(B, F, G, H

Decomposition :

* (1) Loss-less decomposition :- R → n tuple
  R1 → x
  R2 → y
) join → n tuple

2) Lossy decomposition
  R1 → x
  R2 → y
) join → > n

Pno	Aname	Sno	Sname
1	A	10	xyz
1	A	20	xoxy
2	B	10	xyz
3	C	23	ABC
2	B	20	xoxy

Pno Sno

R1 (Pno, Pname, Sno) } loss-less
R2 (Sno, Sname)

Sno	Sname
10	xyz
20	xoxy
23	ABC

Pno	Pname
1	A
2	B
3	C

Sno	Sname
10	xyz
20	xoxy
23	ABC

Lossy decomposition
- info is lost

Pno → Pname
Sno → Sname

dependency preserving
↓
same ही relation में होना चाहिए

* R (A, B, C, D, E)
{
A → B
A → C
B → C
D → E
}

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

Not dependency preserve.

	A	B	C	D	E
R_1	a	a	a	a	a
R_2			a	a	a

Page No.
Date

Group complete FD
→ loss-less

पर Pad करने के पहले
ये देखे कि वहाँ पर
पहले से एक "a" होना
होना चाहिये।।

4)

2) $R(A, B, C, D, E)$

$\{$

$A \to B$

$A \to C$

$D \to C$

$D \to E$

$\}$

Soln

$R_1(A, B, C)$

$R_2(C, D, E)$

Dependency Preserving

	A	B	C	D	E
a_1	A	B	C		
a_2			C	D	E

Soln

5)

↳ lossy decomposition

3) $R(A, B, C, D, E)$

$\{$

$A \to B$

$A \to C$

$D \to C$

$D \to E$

$\}$

$R_1(A, B, D)$

$R_2(A, C, E)$

Not dependency preserving

	A	B	C	D	E
R_1	✓	✓	✓	✓	
R_2	✓	✓	✓		✓

↳ Lossy decomposition

4) $R(A, B, C, D)$

Σ

$A \rightarrow BC$

$BC \rightarrow A$

$B \rightarrow CD$

]

$R_1(A, B, C)$

$R_2(B, C, D)$

Sol^n

dependency preserving

	A	B	C	D	
R_1	✓	✓	✓	✓	
R_2	✓	✓	✓	✓	

↳ Non-les

5.) $R(A, B, C, D, E)$

{

$A \rightarrow DE$

$D \rightarrow B$

$E \rightarrow C$

{

$R_1(A, D)$

$R_2(B, C, D)$

Not dependency preserving

A	B	C	D	E
✓	✓			
	✓	✓	✓	

↳ Lossy - decomposition.

(Directly E तो हमें मिल ही नहीं सकते)

6.) R (A, B, C, D, E, G, H)

{

A → B

A → C

B → D

D → E

E → G

}

R_1 (A, B, C)

R_2 (B, D, E)

R_3 (E, G)

R_4 (A, H)

dependency preserving

	A	B	C	D	E	G	H
R_1	α	α	α		α	α	
R_2		α		α	α	α	
R_3					α	α	
R_4	α	α					α

7) R(a, b, c, d, e)

{

a → c

d → e

}

R_1 (a, c)

R_2 (d, e)

R_3 (a, b, d)

dependency preserving

	a	b	c	d	e
R_1	α		α		
R_2				α	α
R_3	α	α	α	α	α

↳ loss-less ✓

Decomposition in 3NF:

I. Find out all candidate keys of relation
II. Find minimal cover of functional depedency set.
III. make relation with functional depedencies having common left-hand size

$$\Sigma \begin{cases} A \to B \\ A \to C \\ D \to E \end{cases} \Rightarrow \begin{matrix} R_1(A, B, C) \\ R_2(D, E) \end{matrix}$$

IV. If candidate key is not a part of any relation, then make new relation with candidate key.

$$\begin{cases} A \to B \\ A \to C \\ D \to E \end{cases} \Rightarrow \begin{matrix} R_1(A, B, C) \\ R_2(D, E) \\ R_3(A, D) \end{matrix}$$

(1) $R(A, B, C, D, E)$

$$\begin{cases} A \to B \\ A \to C \\ D \to E \end{cases} \quad \begin{matrix} R_1(A, B, C) \\ R_2(D, E) \\ R_3(A, D) \end{matrix}$$

Depded Preserving

	A	B	C	D	E
R_1	✓	✓	✓		
R_2				✓	✓
R_3	✓	✓	✓	✓	✓

Lossless.

2) R(Pno, Sno, Sname, Pname)
{
Pno → Pname
Pno → Sno
Sno → Sname
}
(Not in 3NF or Not)

Pno, Sno, Sname Pname

$\sum Pno^+$ ↓ C.K

Not in 3NF,
So, decompose it

R_1 (Pno, Pname, Sno)
R_2 (Sno, Sname)

	Pno	Pname	Sno	Sname
R_1	∝	∝	∝	∝
R_2			∝	∝

loss less and depedency Presern

Note:
3NF decomposition is always loss-less and dependency preserving.

3) Conside the following relation
R (A, B, C, D, E, G, H)
{
AB → C
A → BD
AD → E
D → DE
E → G
}

Decompose relation in 3NF?

Sol^n

$R_1(A, B, C, D, E)$

$R_2(E, G)$

$AD \rightarrow G$ ✗

A, B, C, D, E, G, H

$\{AB \rightarrow C, A \rightarrow D, AD \rightarrow E, B \rightarrow D, D \rightarrow E, E \rightarrow G\}^+$

$R_1(AB)$

$A \rightarrow C$ ✓
$A \rightarrow D$ ✓
$A \rightarrow D$ ✗
$A \rightarrow E$ ✗
$B \rightarrow D$ ✓
$D \rightarrow E$ ✓
$E \rightarrow G$ ✓

&

Minimal cover is

$\{A \rightarrow C, A \rightarrow B, B \rightarrow D, D \rightarrow E, E \rightarrow G\}$

Scale -

$R_1(A, B, C)$

$R_2(B, D, E)$

$R_3(E, G)$

$R_4(A, H)$

$\Leftarrow$

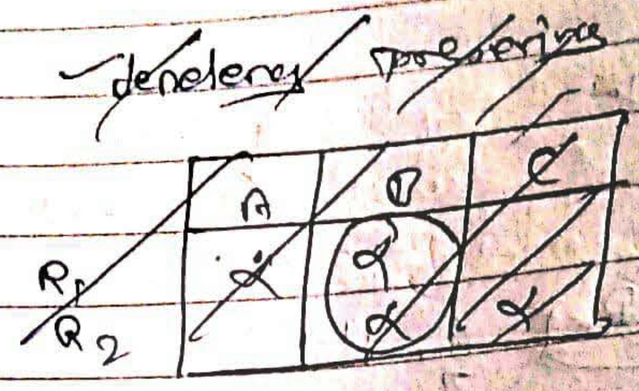
lossless and dependency preserving.

4) $R(A, B, C)$

A	B	C
a_1	b_1	c_1
a_2	b_2	c_2
a_3	b_3	c_2
a_4	b_4	c_1

$R_1(A, B)$

$R_2(B, C)$



R(A, B, C)

{

A → B

B → C

A → C ✗

}

{A, B}

{A, C}

{A → B
B → C
}

R1(A, B)

R2(B, C)

★ Decomposition in BCNF :-

R(A, B, C, D, E)

{

A → B ✗

A → C ✗

D → C ✗

D → E ✗

}

This is not in BCNF, so to decompose

I.) Find out all candidate keys of relation

II.) Find minimal cover

III.) R(A, B, C, D, E)

Right side वाला remove कर दो

R1(A, B)

R2(A, C)

R3(D, E)

Page No. _____
Date _____

Now, $R(A, D)$
$R_1(A, B)$
$R_2(A, C)$
$R_3(D, E)$
→ loss less ✓
→ This is not dependency preserving.

Note: BCNF decomposition is always loss-less but dependency preserving is not guaranteed.

original and new relation both exist but we are performing some updation in original relation while making new relation.
we are designing new relation only for the functional dependency that does not satisfy BCNF rule
suppose
$A \rightarrow B$ does not satisfy BCNF rule,
then make new relation
$R_1(A, B)$
and remove B from original relation

If B exist in the remaining Functional dependency sets, that does not satisfy BCNF then such type of functional dependency is directly discarded without making new relation.

Q) check the following decomposition is in BCNF or not.
$R(A, B, C, D, E)$
$\{$
$A \rightarrow C$
$D \rightarrow CE$
$\}$

Sol:
$R(A, D, C, D, E)$
$\{ A \rightarrow C$ ✓
$D \rightarrow C$ ✗
$D \rightarrow E$ ✗
$\}$

$\{ R_1(A, C)$
$R_2(D, E)$
$R_3(A, B, D)$
$\{$
$R(A, D, C, D, E)$
$R(A, C)$
$R_1(D, E)$

4NF

Note

If x →→ y is multi value dependency
then
x →→ R−y is also multi value attribute

Cname	Aeth	Inst

Cname →→ Aeth then Cname →→ Inst

** 4NF
↳ trivial
↳ (x →→ y) then y ⊆ x
or x ∪ y = R
↓
Non-trivial
then x must be super key.

(P.A), R

Q1) NPA → P.A

Q1(A) r → w
rw → x
y → rx
y → z

⇒ r → w
r → x
y → v
(y → x) ✗
y → z

Q2. (B) r → w

Q3. (B) r y

Q4) (B) R(V, N, S, E, T, Y, P)
(VNSE)
VNSE → T
VN → Y (N.D.A)
VNSE → P

} 1NF

5)(C) x
6(A)
7(D)
08(B)
9(1
10(B)
11(B)

Page No.
Date

R(V, N, S, E, T, P) → is in BCNF → 3NF → 2NF → 1NF

R2(V, N, V)

BCNF → 3NF → 2NF → 1NF

Ans 2NF (D)

5(C) X(P, Q, R, S, T, U)

F = {
{P, R} → {S, T}
{P, S, U} → {Q, R}
∅
}

R = {E, F, G, H, I, J}

6(A)

7(D)

8(B)

R(E, F, G, H)

↳ K ⇒ key

E EFG
EF EFH
EG EGH
EH EFGH

Ans "8"

9(19) → '*' में 19 डालो पै यो data save ही जाएगा।

10(B) यहाँ Composite key नहीं होगा तो, Prime attribute नहीं होगा।

11(B)

CH → G
A → BC
B → CFH
E → A
F → EG

CH → G
A → B
A → C
B → C
B → F
B → H
E → A
F → E
E → G

A B C D E F G H

{ AD
BD
ED
FD } ⇒ 4 candidate key,

(12)(A) A → BC
PA

Not in 2NF
so 1NF

13(C)

14(C) Book(T, A, C, P, Y, Pr)
Collection(T, A, C)
{ TA → C → BCNF → 2NF
C → TAPY → 2NF
PTY → Pr → 2NF
NPA   NPA
{T, A}⁺

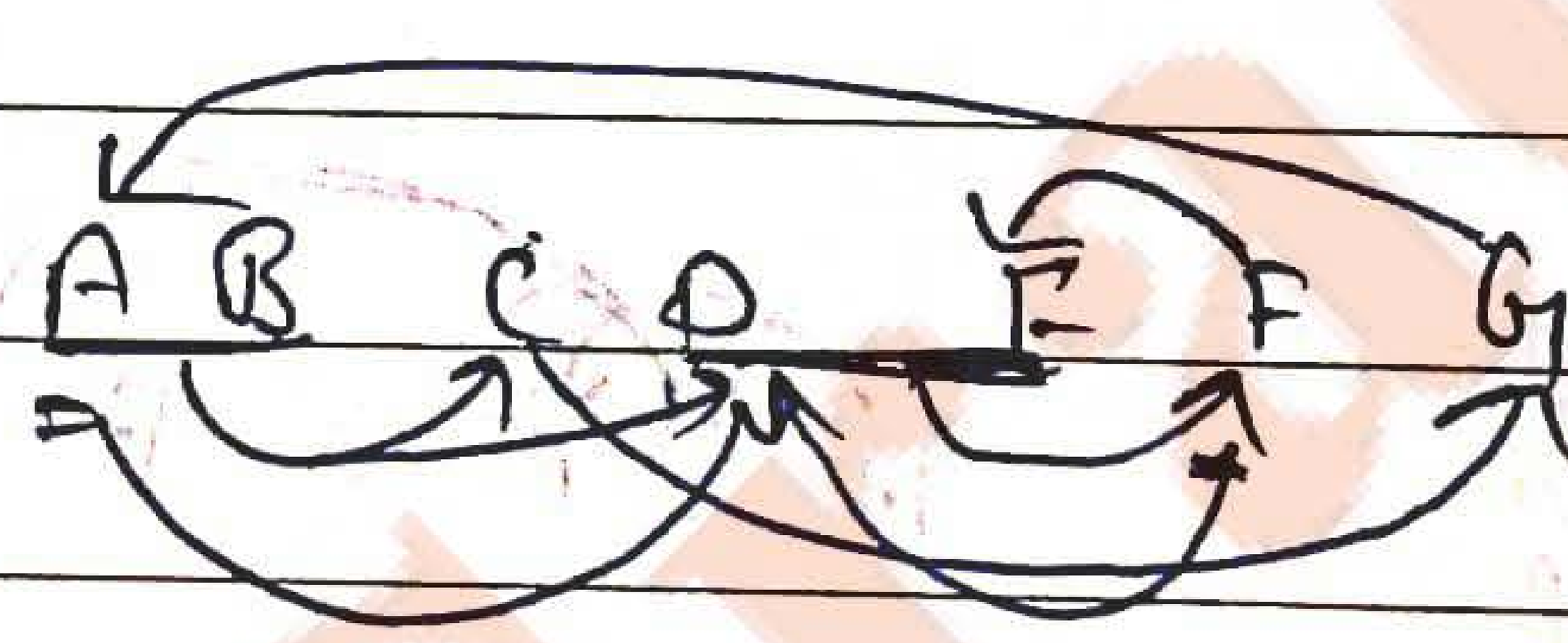
2NF ✓

So,
book is in 2NF

Now
collection
{ TA → C — BCNF
C → T — 3NF
C → A → 3NF
}
3NF

15(D) 3NF α → β
Prime
depends
on

16(GD)



{CF}⁺ → {C, F, E, G, A, D}

{BG}⁺ → {B, G, A, C, D}

{AF}⁺ → {A, F, D, E}

{AD}⁺ → {A, D, C, D, G}

17)(D)

18)(D)

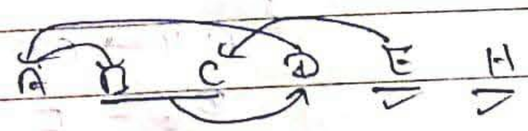
R(A, B, C, D, E, H)

A → B

BC → D

E → C

D → A



(E H)

19)(B) { name CNo } → B

{ Roll no CNo } → grade

name CNo → grade } BCNF

rollno CNo → grade

name → rollno } 3NF

rollno → name

} 3NF

20)(D)

R(rollno, Name, Dob, Age)

Σ

Dob → Age ✓

✗ Age → eligibility → does not exist

Name → Rollno ✓

Rollno → name ✓

CNo → Cname ✗

Cno → Inst ✗

Rollno CNo → grade ✗

{

⇓

PP { P.A NP.A

Dob → Age → not in 2NF so in 1NF

name → Rollno

Rollno → name

}

21)(B) R(A, B, C, D)?

A → B

B → C

C → D

D → B

	A	B	C	D
R_1	a	a	a	a
R_2		a	a	a
R_3		a	a	a

loss less ans dependency not preserve

$\{1,2,3\}^+ \rightarrow \{1, x, y, 8w, z\}$

$(23)(\textcircled{3})\ R\ (1, w, x, y, z)$

$11 \rightarrow \textcircled{3}$

$wx \rightarrow z$

$zy \rightarrow 1$

$\{1, x, y, 8w, z\} \rightarrow \{1, 2, 3\}^+$

$\textcircled{3}\ E \rightarrow$

$\beta \rightarrow \alpha$

$E \rightarrow C$

$C \rightarrow E$

$[23)(\textcircled{3})\ R\ (A, B, C, D, E)$

$\{A \rightarrow C\textcircled{3}$

$\beta E \rightarrow E$

Page No.

Date

Transaction & serializibility

Transaction:

(1) ACID properties:

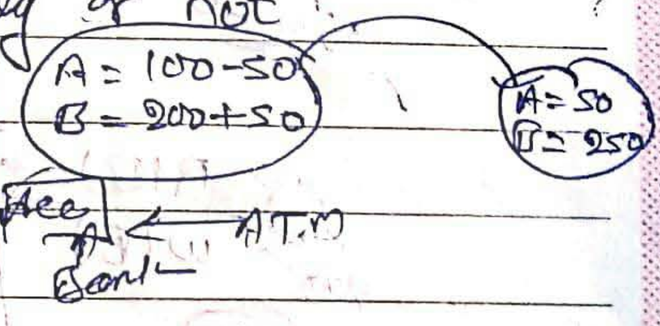
(I) Atomic → either committed fully or not.

(II) Consistency → one → other

(III) Isolation →

(IV) Durability →

↳ If succes, then shud should not be fail.



(2)

T_1	T_2
R(A)	R(A)
A=A+50	A=A-30
W(A)	W(A)
R(B)	R(B)
B=B-50	B=B+50
W(B)	W(B)

(serial)

$T_1 \rightarrow T_2$

o/p A = 120
B = 200

A=100 ~~150~~ 120
B=200 ~~150~~ 200

(3)
1. To improve response time
2. to reduce waiting time
3. to improve performance of the system
4. to improve resource utilization.

} Concurrent execution (1)

T_1	T_2
R(A)	
A=A+50	
150	
	R(A) 100
	A=A-30,
	W(A) 70
W(A) 70	
R(B)	
B=B-50	
150	
	R(B)
	B=(B+50), 250
250 W(B)	W(B) 250

A=100 ~~150~~ 120 ✓
B=200 ~~150~~ 200 ✓

A=100
B=200

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
R(B)	
W(B)	
	R(B)
	W(B)

Note: If concurrent schedule is equivalent to serial schedule then such type of concurrent schedule is called serializable schedule.

- To test concurrent schedule is serializable or not, we use following two testing method.
  (1) Conflict serializable
  (2) view serializable

(1) Conflict serializable

- ✱ To check schedule is conflict serializable or not firstly design Presedence graph of concurrent schedule.
- ✱ No. of nodes in presedence graph is equal to no. of transaction in schedule.

Page No.
Date

Note:

T_1 T_2

- $R(A)$ → then → $R(A)$ ✗ (No edge)
- $R(A)$ → $W(A)$

- $W(A)$ → $W(A)$

- $W(A)$ → $R(A)$

(*) Precedence graph never contain self loop.

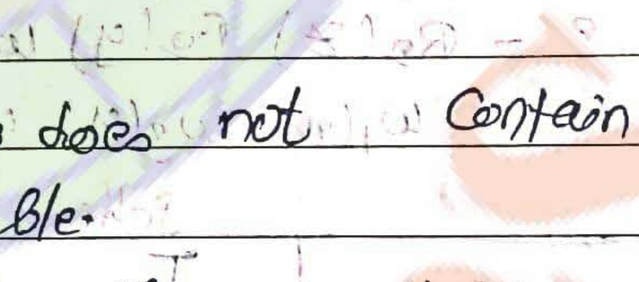
(*) If precedence graph does not contain cycle, then it is conflict serializable.

* If it contain cycle, then it is conflict serializable.

(*) If conflict serializ
ie.
Let s serializable,

1)a)

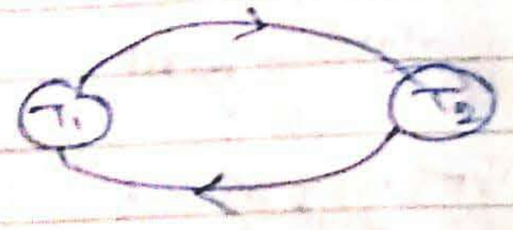
S	
T_1	T_2
$R(A)$	
	$R(A)$
	$W(A)$
$W(A)$	
$R(B)$	
	$R(B)$
	$W(B)$
$W(B)$	



Contain cycle
Not conflict serializable

(2.)

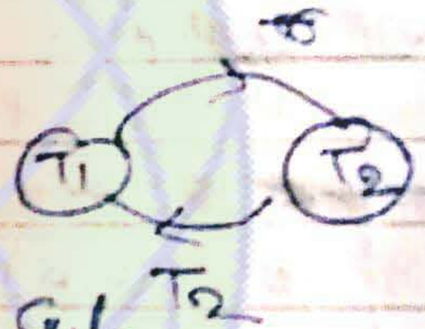
T_1	T_2
R(A)	
	R(A)
	W(B)
R(B)	
W(B)	
	W(A)



cycle,
Not conflict serializable

(3)

T_1	T_2
r(x)	
r(y)	
	w(y)
	w(x)
w(x)	
w(y)	



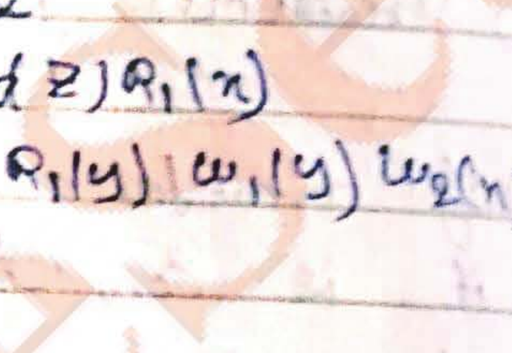
Cycle
Not conflict serializable

4) Consider the following schedule

$S = R_2(z)\ R_2(y)\ W_2(y)\ R_3(y)\ R_1(z)\ R_1(x)\ W_1(x)\ W_3(y)\ W_2(z)\ R_2(x)\ R_1(y)\ W_1(y)\ W_2(x)$

Schedule (S)

T_1	T_2	T_3
	$R_2(z)$	
	$R_2(y)$	
	$W_2(y)$	
		$R_3(y)$
		$R_3(z)$
$R_1(x)$		
$W_1(x)$		
		$W_3(y)$
		$W_3(z)$
$R_1(y)$		
$W_1(y)$		
	$R_2(x)$	
	$W_2(x)$	



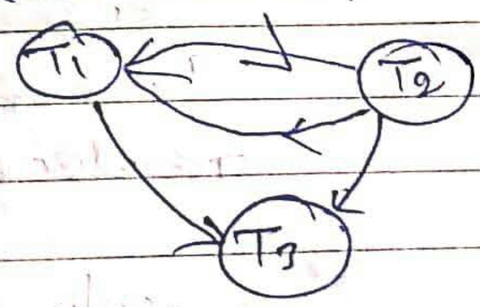
cycle
Not conflict serializable

R1

5)

T_1	T_2	T_3
R(Q)		
	W(Q)	
W(Q)		
		W(Q)

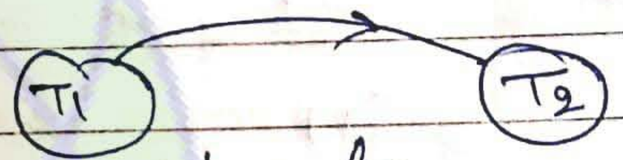
Precedence graph:



cycle
Not conflict serial

6) S: $r_1(x)$; $w_1(x)$; $r_2(x)$; $r_1(y)$; $w_2(x)$; $w_1(y)$

T_1	T_2
$r_1(x)$	
$w_1(x)$	
	$r_2(x)$
$r_1(y)$	
	$w_2(x)$
$w_1(y)$	



Not cycle
conflict serializable.

(*) Every conflict serializable schedule is serializable but vice versa is not true.

R1 (*) Every conflict serializable is view serializable but vice versa is not true.

Conflict equivalent:

S_1

T_1	T_2

S_2

T_1	T_2

two schedule S_1 & S_2

(i) No. of transaction in both schedule should be equal.

(ii) Precedence graph of both schedule should be same.

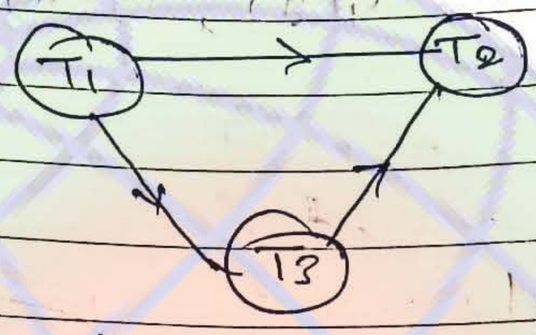
(iii) both schedule should be conflict serializable,

Page No.
Date

Topological sort - (when graph is acyclic)

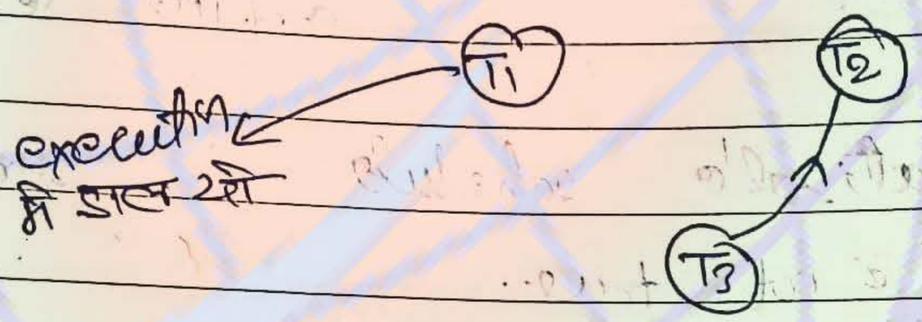
To check a concurrent schedule is equivalent to which serial schedule. for that we are performing topological sort on presidence graph

⇒ only for the conflict serializable schedule.



It is Conflict serializable

a) a node with in-degree is ~~zero~~ zero ⇒ T1



b) Now again in-degree zero = T3

T2

execution में डाल दो
T3

c) then T2 will execute

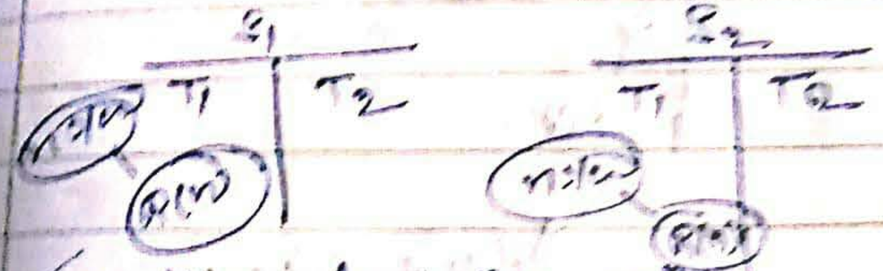
So,

So

$T_1 \rightarrow T_3 \rightarrow T_2$ Ans

D. View serializable

view equivalent

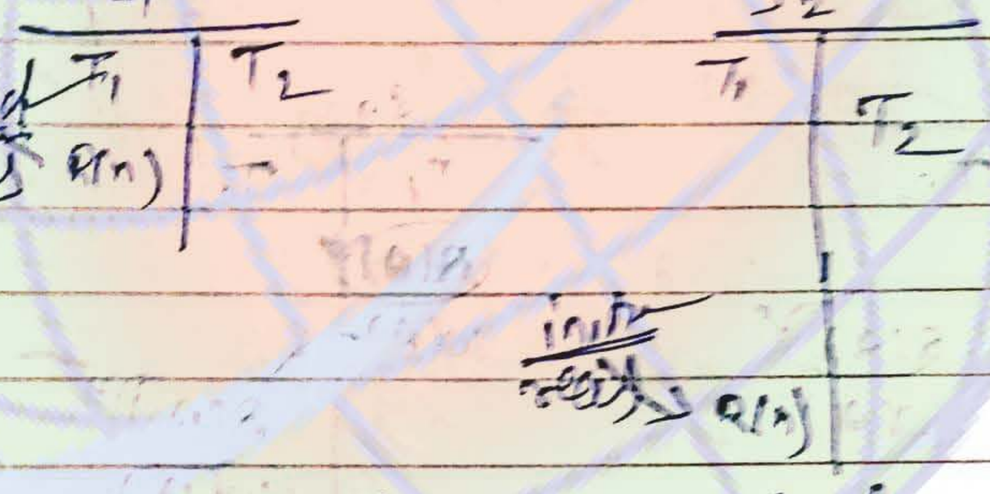


Condition for view eq :-

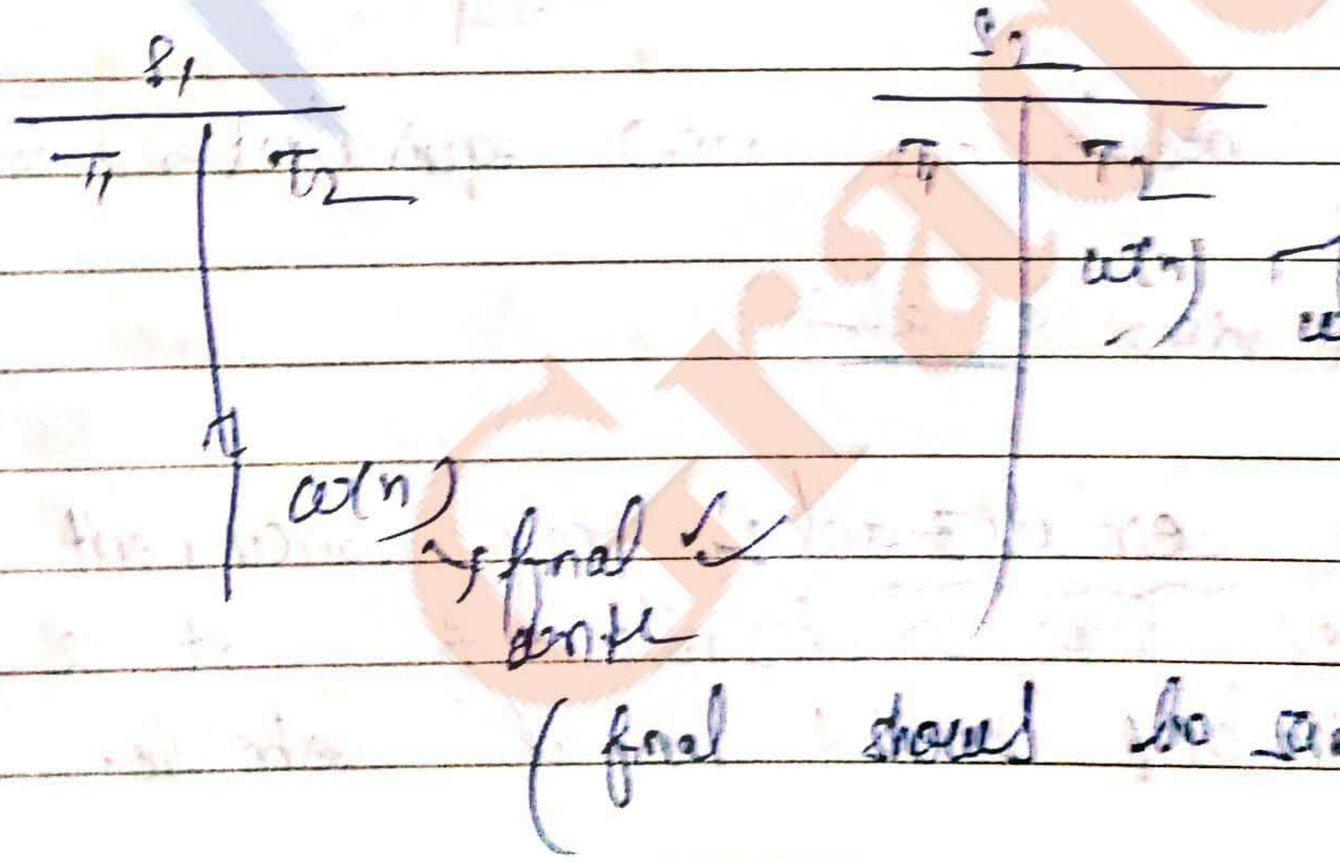
(i) If in transaction in both transaction schedule should be same

(ii) If transaction T_1 perform initial read operation on same data item in schedule S1, then same transaction must perform initial read on the same data item in schedule S2.

Eg.

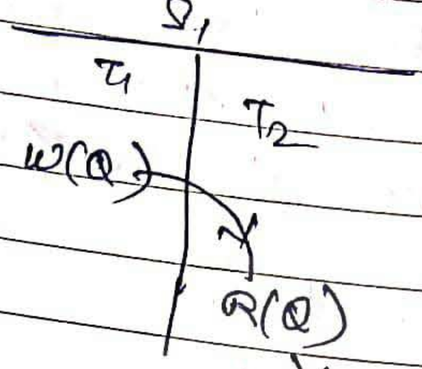


iii) If transaction T_2 perform final write operation on same data item, in schedule S1, then transaction T_2 must perform final write operation on same data item in schedule S2.



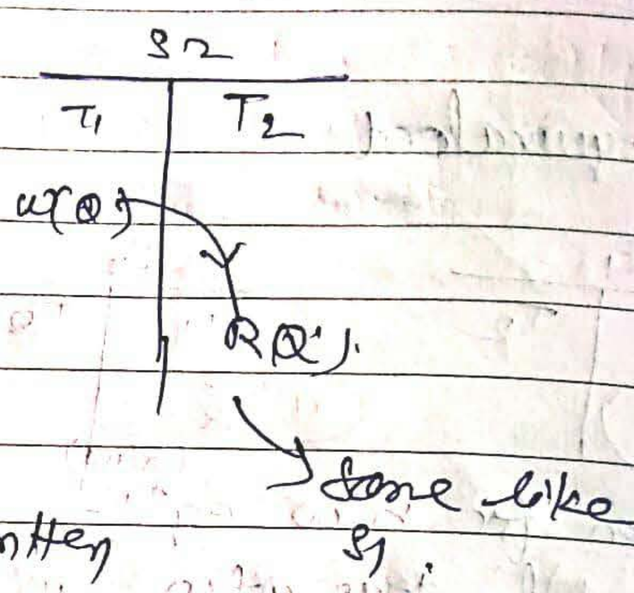
(final should be same)

(iv)



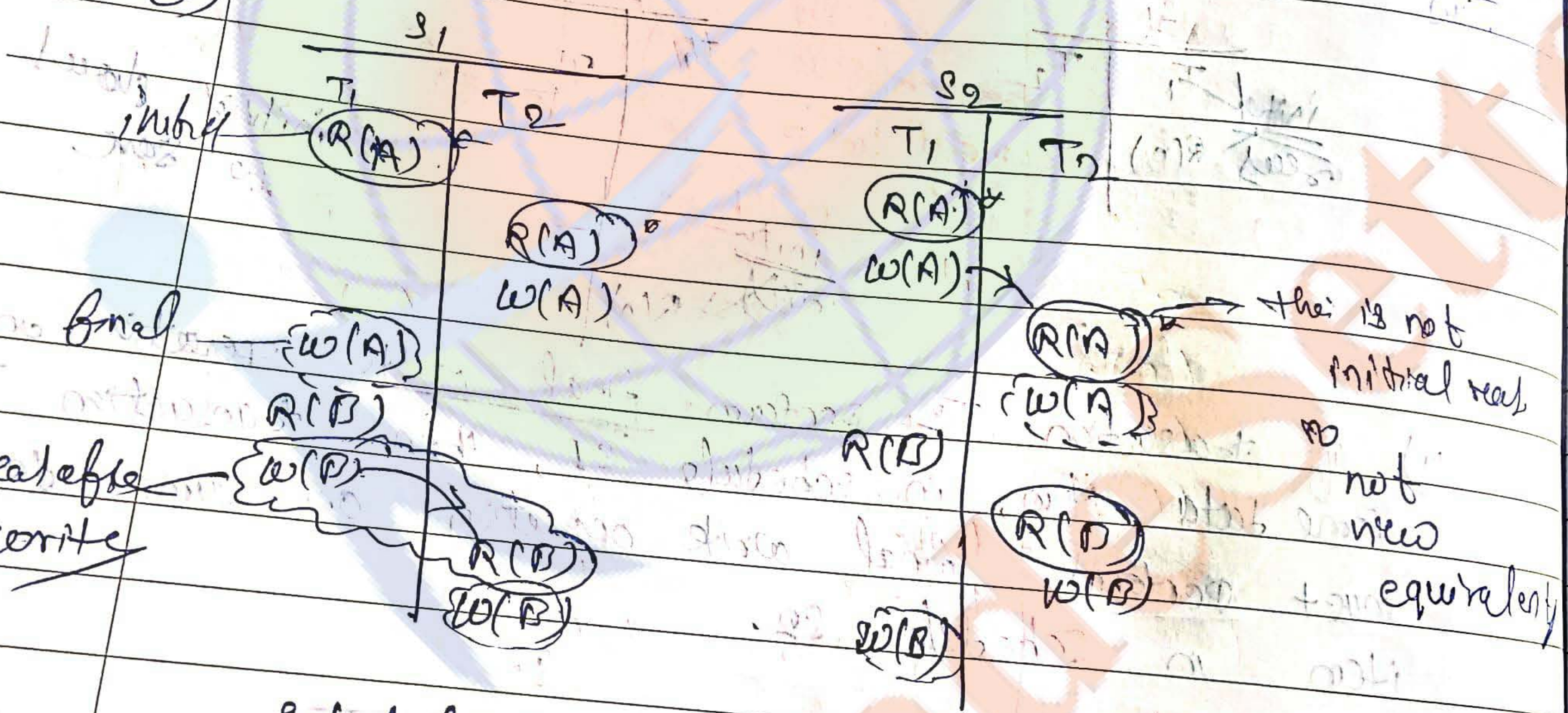
reads after write

reads some data which was previously written by some other transaction then same will be in S2



If a transaction T_2, reads some data item, that is previously written by transaction T_1, in schedule S_1, then same concept must be in S_2.

(v)



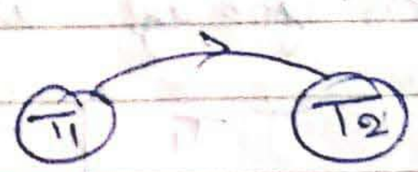
Schedules are view equivalent or not.

Not view equal

view serializable: Any concurrent schedule is called view serializable, if it is view equivalent to its serial schedule.

(1)

T_1	T_2
*R(A)	
W(A)	
	R(A)
	W(A)
*R(B)	
W(B)	
	R(B)
	W(B)



conflict serializable.

(2)

T_1	T_2
*R(A)	
W(A)	
*R(B)	
W(B)	
	R(A)
	W(A)
	R(B)
	W(B)

Note: जब तक पहले T_2 के साथ का check कर लिया तो अगला गलत है तो

* ① ✓ initial
② ✓ final
③ ✓ reads after write.

so, it is view serializable

(2:)

T_1	T_2
*R(x)	
	*R(x)
W(x)	*R(y)
W(y)	
	W(y)

final

Not conflict serializable
so not serializable

T_1	T_2
R(x)	
W(x)	
W(y)	
	R(x)
	R(y)
	W(y)

Not view serializable

T_1	T_2
	*R(x)
	R(y)
	W(y)
$R_2(x)$	
W(x)	
W(y)	

Not view serializable.

Blind write: If any transaction directly write any data item without reading then such type of write is blind write.

T_1	T_2
R(n)	
W(n)	
Blind write → W(y)	
	R(n)
	R(y)

2 #

T_1	T_2
*R(x)	
	*R(n)
	*R(y)
W(n)	
Blind write → W(y)	W(y)

check view serialized or not

Ex/n

T_1	T_2
*R(n)	
**W(n)	
**W(y)	
	*R(n)
	*R(y)
	W(y)**

Not view ser

T_1	T_2
	*R(n)
	**W(y)
*R(n)	
*W(n)	
**W(y)	

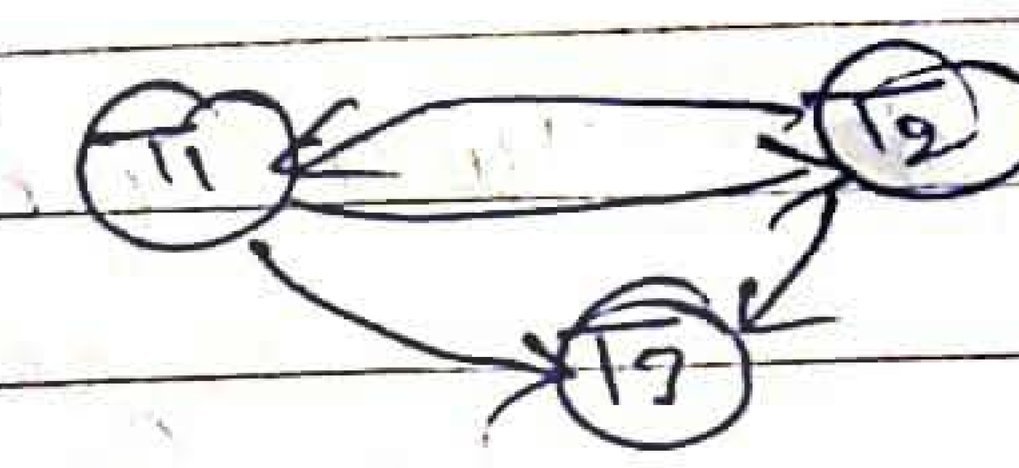
view serial

so, it is view serializable with $T_2 \rightarrow T_1$

3)

T1	T2	T3
R(Q)		
	W(Q)	
W(Q)		
		W(Q)

initial read — blind write



Not conflict serializable

T1	T2	T3
R(Q)		
W(Q)		
	W(Q)	
		W(Q)

final write — view serializ

1 2 3
1 3 2

Note: this is not conflict serializable but it is view serializable

Not ~~view serializ~~

Note in-degree zero वाला method से पता करना होता उसको युगल करो

T1	T2	T3
~~R(Q)~~		
~~W(Q)~~		
		~~W(Q)~~
	~~W(Q)~~	

4)

★ Locking Protocol

There are two type of lock:
① Shared lock — only read
② exclusive lock — write

	shared lock (A) read S(A)	exclusive lock (A) write (X(A))
Shared lock (A) (read) S(A)	Yes	NO
X(A)	NO	NO

only share - share is possible

R1

Page No. ____
Date ____

Two phase locking protocol:-

I. Growing phase. – lock acquire
II. shrinking phase. – lock release

Note: ~~Any transaction~~
- Firstly growing then shrinking ~~and~~ but
- After shrinking growing is not possible.

T1	T2
X(A)	
S(B)	
	S(B)

S

T1	T2
X(A)	
R(A)	
W(A)	
X(B)	
R(B)	
W(B)	
Unlock	
R(B)	
W(B)	
Unlock(B)	

growing phase: जो lock ~~लिये~~ उन्हीं को use कर सकते हैं

shrinking phase

Here transaction is in 2PL (two phase lock)

R1 Every 2PL is conflict serializable but reverse is not true.

Any transaction only in growing phase, still it is in 2PL.

Problem in 2PL :-

① Cascading rollback

②

T_1	T_2	T_3
X(A)		
R(A)		
W(A)		
Unlock(A)		
	X(A)	
	R(A)	
	W(A)	
	Unlock(A)	
		S(A)
		R(A)

rollback

rollback

rollback

fail

because of any failure if multiple transaction rollback then this is known as cascading rollback

② Deadlock :-

T_1	T_2
X(A)	X(B)
R(A)	R(B)
W(A)	W(B)

wait

wait

Deadlock

Note (*) Irrecoverable schedule :-

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
	C;
fail;	

rollback

→ Irrecoverable schedule :

because commit before fail.

To remove irrecoverable, we uses :-

① Recoverable schedule

② cascadless schedule.

① Recoverable schedule-

T_1	T_2	T_3
R(A)		
W(A)		
	R(A)	
	W(A)	
		R(A)
		W(A)
C		
	C_2	
		C_3

→ Here there is a problem of cascade rollback.

To remove cascade rollbacking, we use Cascadeless schedule

② Cascadless schedule.

T_1	T_2	T_3
R(A)		
W(A)		
C_1		
	R(A)	
	W(A)	
	C_2	
		R(A)
		W(A)
		C_3

Now,

Note Every cascadeless schedule is always recoverable schedule but vice-versa is not true.

Page No.
Date

★ How to overcome problem of two phase locking -

① Cascade rollback

↳ a) strict 2PL:

T_1	T_2
X(A)	
R(A)	
W(A)	
C	
Unlock(A)	
	X(A);
	R(A);
	W(A);

- All exclusive lock should commit before unlocking.
- Deadlock is possible in Strict 2PL.

b) Rigorous 2PL:-
Both exclusive and shares lock should commit before unlocking.

Note:
* All 2PL are recoverable? T/F, also all 2PL are not cascadeless
* All strict 2PL are recoverable T/F
* All strict 2PL are cascadeless T/F

② Deadlock:-

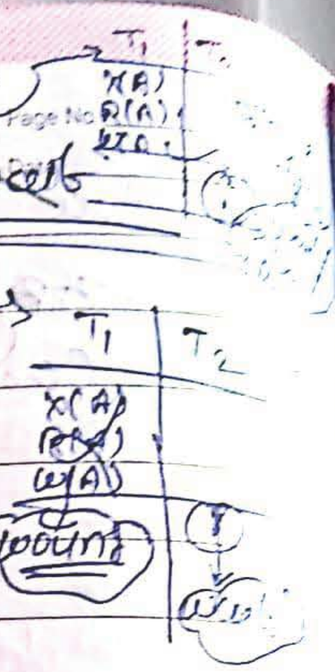
↳ a) Conservative 2PL:-

T_1	T_2
X(A)	
X(B)	

- It will acquire all lock initially.

Not: This is not practically in use.

Deadlock → time stamp
a) wait-die
b) wound-wait



b) Time stamp protocol :-

★ R(A,B,C,D)

a) A → B
b) A → B , D → C, C → D
c) AD → C , C → AD
d) A → BCD

★ Time-stamp Protocol :-

R-time stamp (x) :
W-time stamp (x) :-
time stamp (Ti) :-

① Real-time stamp (x) :
It is largest time stamp of any transaction that perform read(x) operation successfully.

② Write-time stamp (x) :-
" "
that perform write(x) "

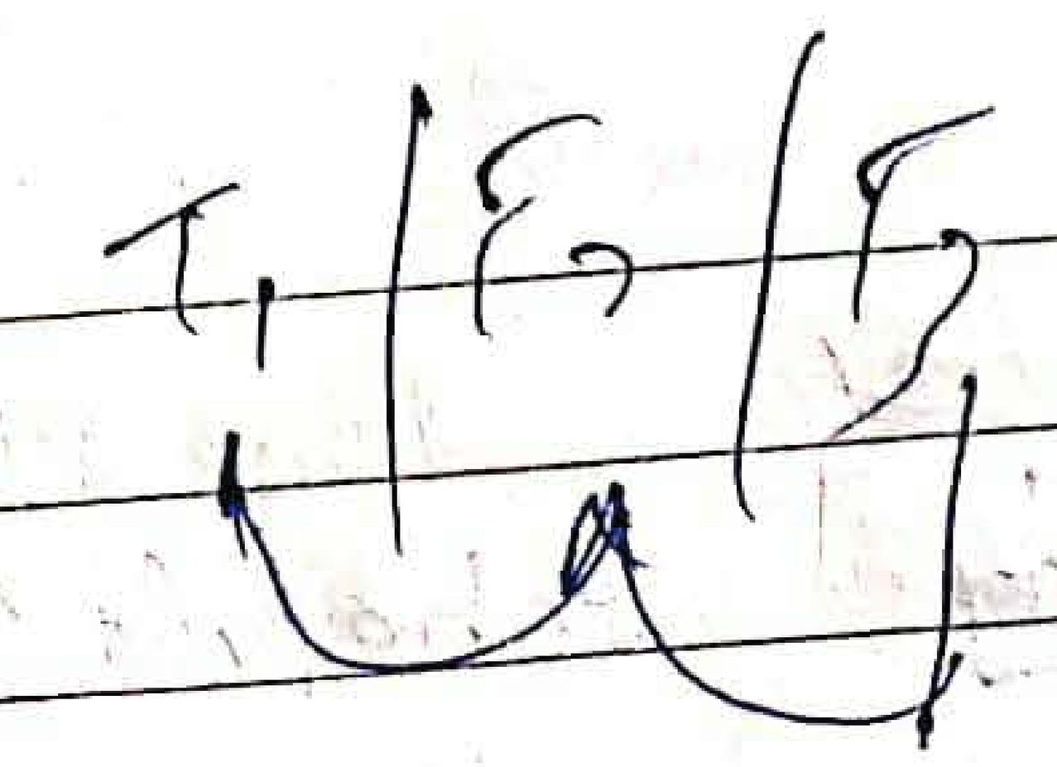
3) time stamp (Ti) :-

steps for time-stamp protocol :-

I.) If transaction (Ti) issue read(x) operation

a.) If TS(Ti) < W-time stamp (x)
then
operation reject & Roll-back.

Page No. ______
Date ______



b) If $Ts(T_i) >= W\text{-timestamp}(x)$
then
operation executed and updated
R-timestamp to max (R-timestamp, $Ts(T_i)$)

(maximum को set करते हैं)

(अगर read के लिए time stamp का value write-time stamp से ज्यादा होना चाहिए)

II.] If transaction T_i issue write(x) operation:-

a) If $Ts(T_i) <$ R-timestamp (x)
then
operation reject & rollback

(अपनी write के लिए read और write दोनों से Ts का value ज्यादा होना चाहिए)

b) If $Ts(T_i) <$ W-timestamp (x)
then
operation reject (If update old data value of x)
rollback

c) otherwise operation execute and
update W-timestamp(x) to $Ts(T_i)$

(time stamp को set / update करते हैं)

eg 1.) Suppose dataitem(x) has a read time stamp of 2:30, and a write-time stamp of 2:35. Can transaction (T) with time stamp 2:29 perform read(x) operation

2:29 < 2:35

reject

eg 2.) Suppose dataitem (x) has a r-time stamp of 2:30 and w-time stamp 2:35, if time-stamp of transaction (T) is 2:32. Can transaction (T) perform write(x) operation

operation reject

2:32 < 2:35

a) Consider that we have applied time-stamp based protocol on the following schedule

	T_1	T_2	T_3
Timestamp	2	4	6
operations	R(A)	R(A)	R(A)
	W(A)		W(A)

a) If the execution seq happens like:

T_0's R(A)

T_0's W(A)

T_2's R(A)

T_2's W(A)

T_1's R(A)

Which of the following is correct

a) T_0 reads rollback

b) T_0 " "

c) T_2 " "

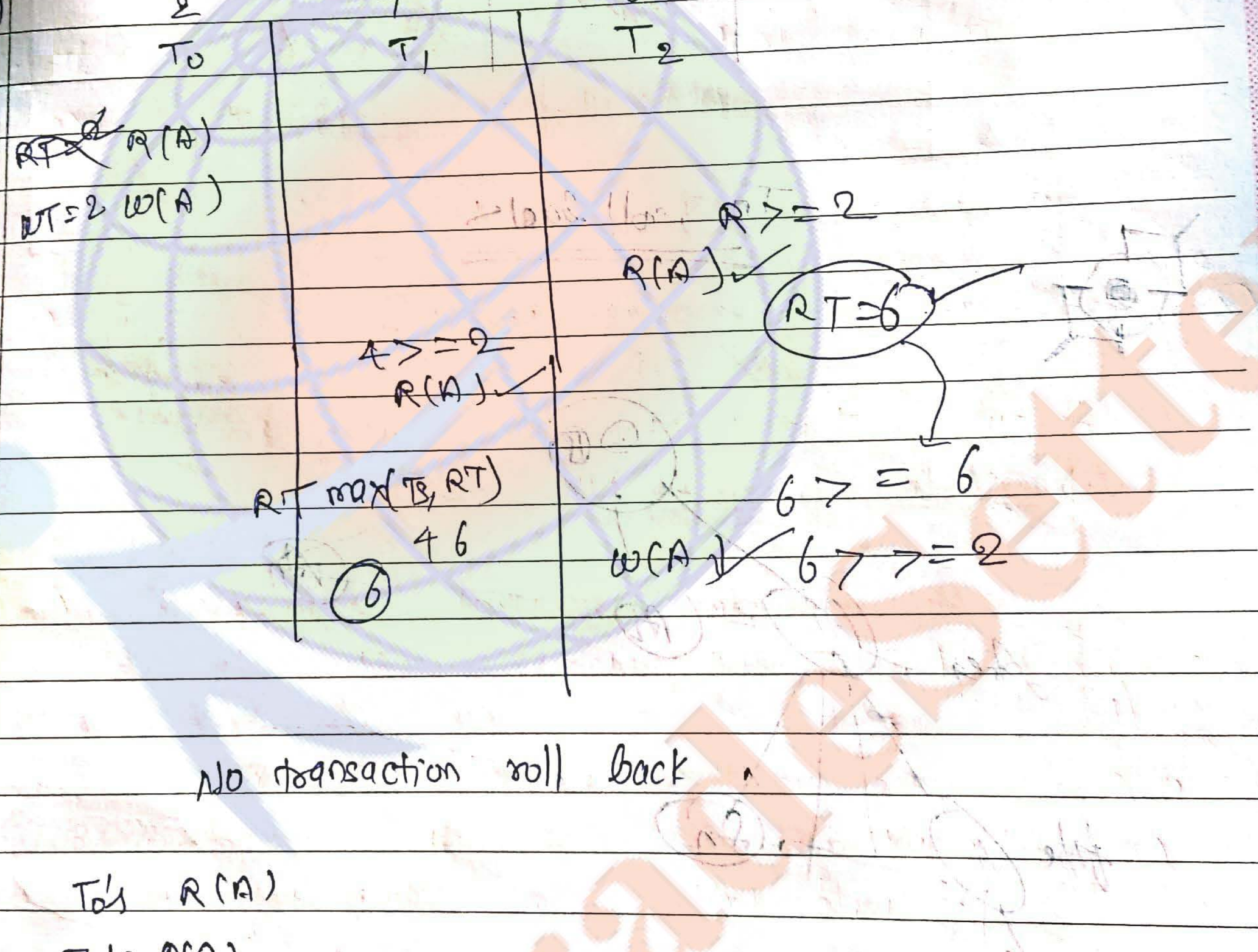
d) No transaction rollback.

2 T_0	4 T_1	6 T_2
2 R(A)		
2 W(A)		
		R(A) $T_2(6) >= 2$
		W(A) RT(2,6) = 6
	R(A) $T_1(4) >= 6$	
		$T_2(6) >= 6$
		$T_2(6) >= 2$
		WT(6)

Page No.
Date

Q6) If the execution seq' happens like:-

T_0's R(A)
T_0's W(A)
T_2's R(A)
T_1's R(A)
T_2's W(A)



No transaction roll back.

Q7) T_0's R(A)

T_1's R(A)

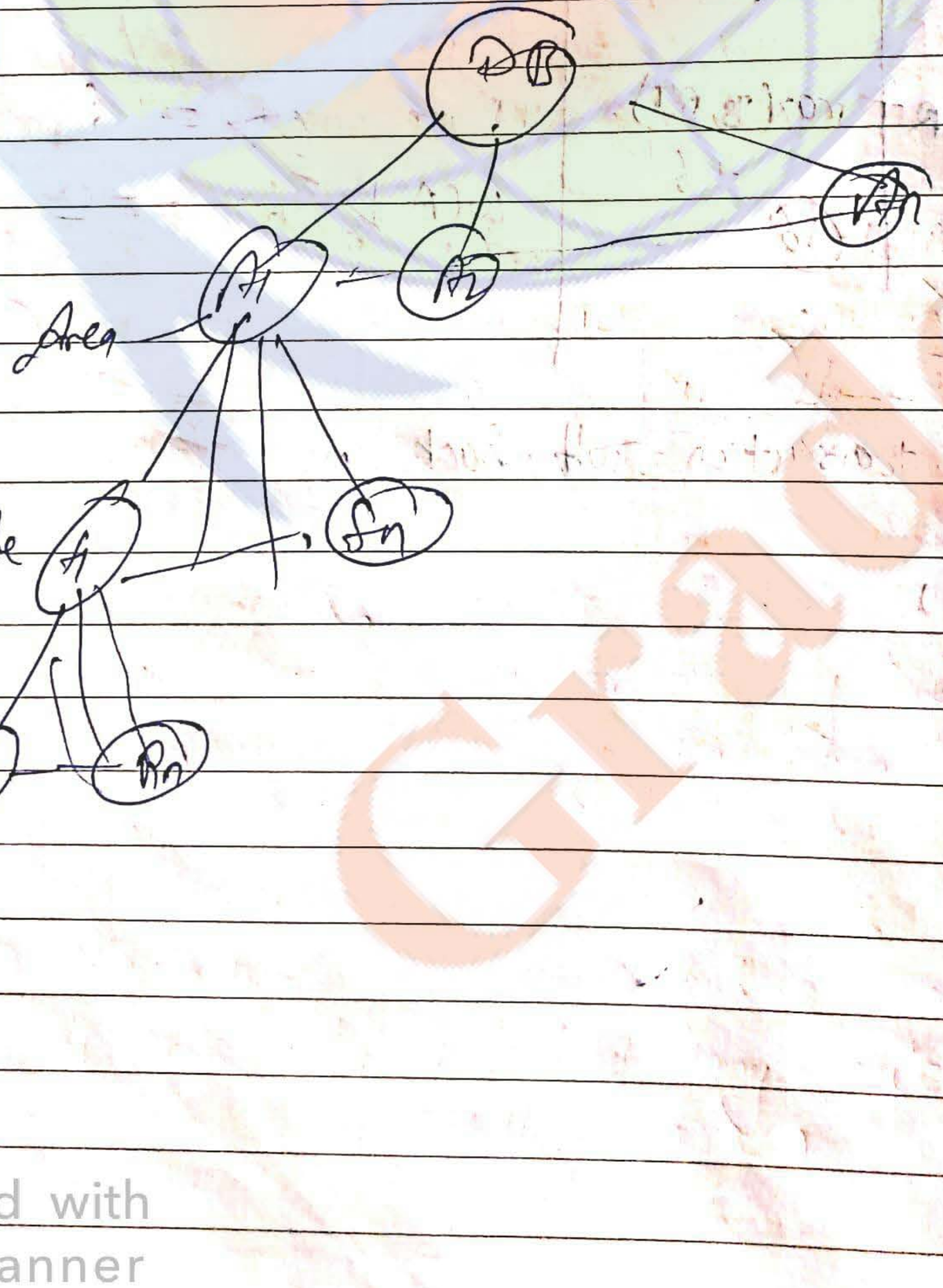
T_0's W(A)

T_2's R(A)

T_2's W(A)

2	4	6
T_0	T_1	T_2
RT=2 R(A) ✓	4>=2	
RT>2 (RTS>2)	R(A) ✓	
WT>2 W(A)	(RTS=4)	W R(A)
4		W(A)

To, roll back

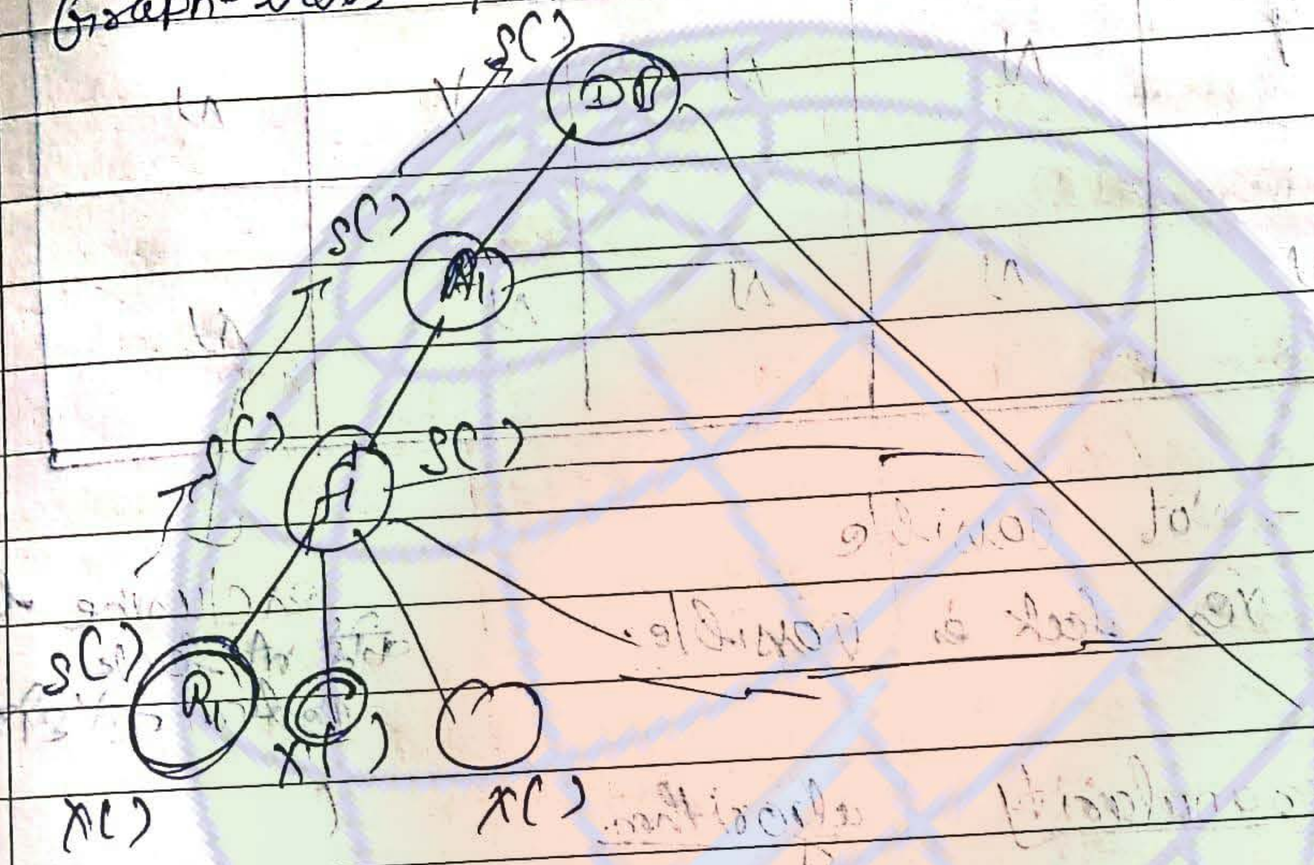


Page No.
Date

① Graph-based Protocol:
(tree-based protocol)

② multiple granularity :-

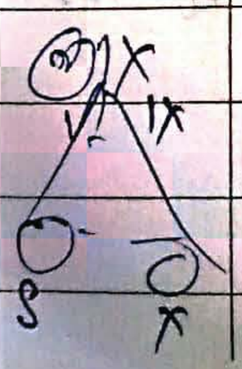
① Graph-based protocol:-

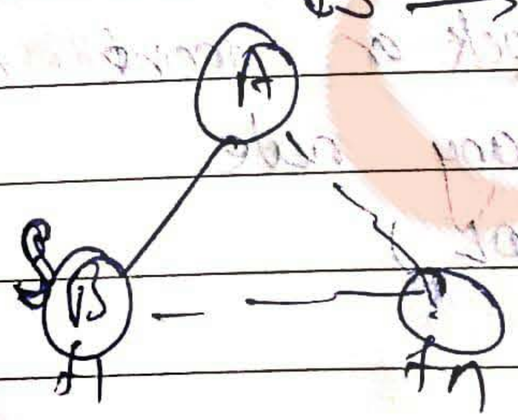


There are five different locks:-

① IS - Intension shares lock
② IX - Intension exclusive lock
③ SIX - shares and Intension exclusive lock
④ S - shared lock
⑤ X - exclusive lock

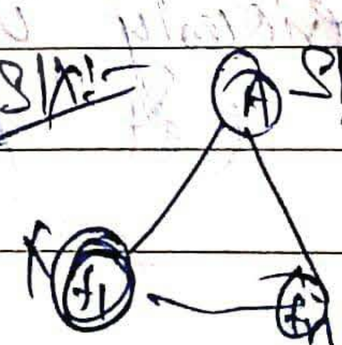
1S:-





IS → पूरा Area नहीं share हो रहा
कुछ file share हो रहा है

2 SIX:-



SIX → real पूरे Area पे कर रहा है
पर unte कोई file/area पे कर रहा है

	IS	IX	SIX	S	X
IS	Y	Y	Y	Y	N
IX	Y	Y	N	N	N
SIX	N	N	N	N	N
S	Y	N	N	Y	N
X	N	N	N	N	N

N! - Not possible
Y → Yes lock is possible.

exclusive lock ही
कोई भी lock perform नहीं होगा

II.) Multiple granularity algorithm.

① A node Q, can be locked by transaction (T_i) in S or IS mode only at the parent of Q is currently locked by T_i in either IX or SIX or IS mode.

② A node Q, can be locked by transaction (T_i) in X, SIX or IX mode only if the parent of Q, is currently locked by transaction T_i, in IX or SIX mode.

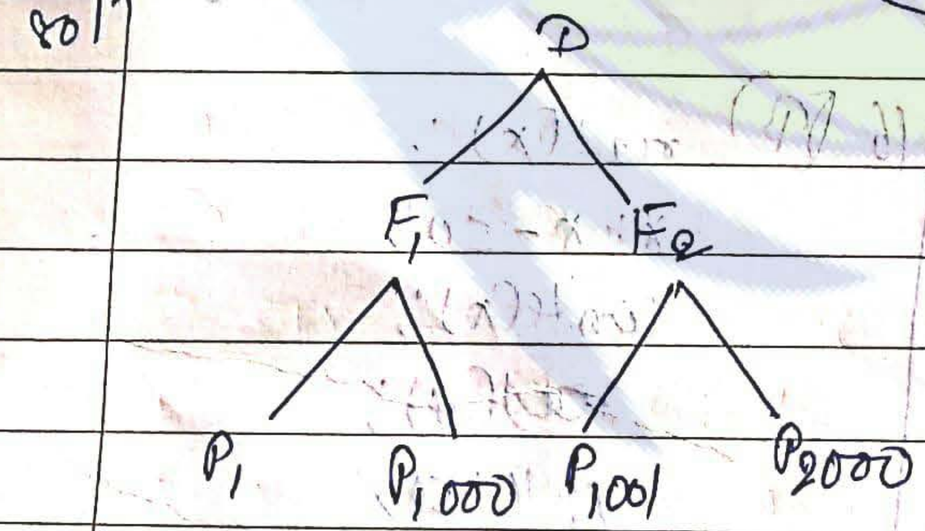
③ Transaction T_i can lock or more only if it has not previously unlocked any node
(It follow 2PL)

4) Transaction (T_i) can unlock a node Q, only if none of the child of Q, are currently locked by T_i.

Q.) Consider database D, contain two file f_1 and f_2 each file consists 1000 pages, f_1 contains P_1 to P_{1000} f_2 consists P_{1001} to P_{2000}.
each page contain 100 record, index of record is P, i. Database is manages using multiple granularity protocol.
So what would be the seq^n of lock. If Transaction(T) want to perform
read record $P_{1200:5}$

a) D: IS F2: IS P: 1200: IX P1200: 5: X
b) D: IS F2: IS P: 1200: IS P: 1200: 5: IS
c) D: IX F2: IX P: 1200: IX P: 1200: 5: IX
d) D: IS F2: IS P: 1200: IS P: 1200: 5: S

→ shared lock



(P: i) P_5: 1000

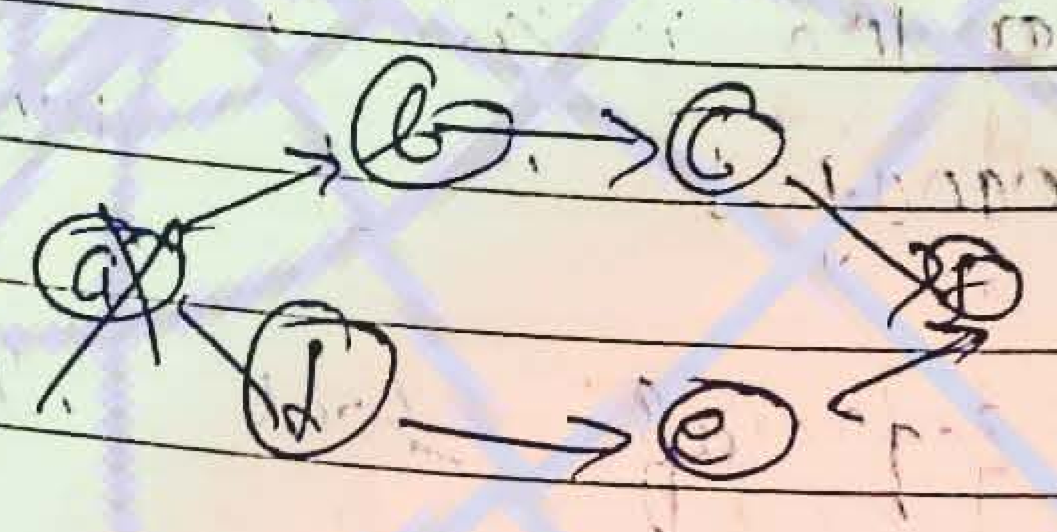
Q) Read Pages P_{10} to P_{980}
a) D: IS F_1: IS P_{10}: 5: S
b) D: IS F_1: IS P_{980}: 5: S
c) D: IS F1: IS
d) D: IS F1: S → पूरे को share कर देंगे

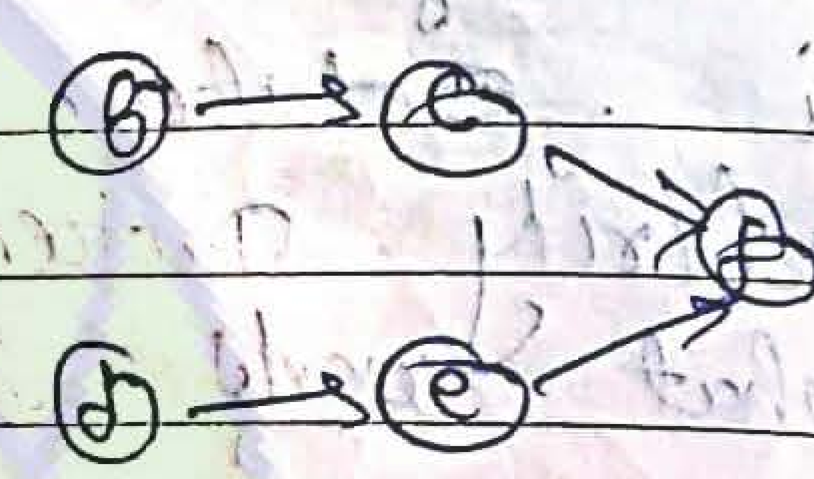
Q) Delete Record P1200:'98

a)	D:1S	F2:1S	P1200:1X	P1200:'98:X
b)	D:1X	F1:1X	P1200:1X	P1200:'98:X
~~c)~~	D:1X	F2:1X	P1200:1X	P1200:'98:X
d)	D:1S	F2:1X	P1200:1S	P1200:'98:X

★ Grate Question (Previous year)

Q27 (B)





b → c

d → e

(8) (A)

4 (A) Order of execution → on the basis of topological order

5) (C)

T_1	T_2
	$r_2(x)$
$r_1(x)$	
	$r_2(y)$
$w_1(x)$	
$r_1(y)$	
	$w_2(x)$
q_1	
	q_2

(6 (B)) read(x);
x := x - 50;
write(x);
read(y);
y := y + 50;
write(y);

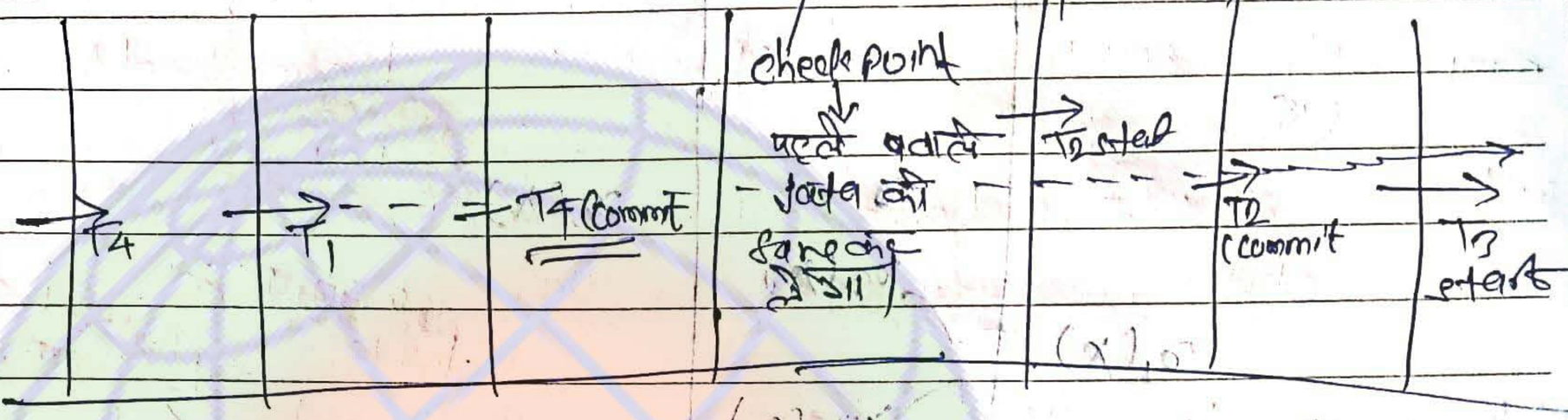
Page No.
Date

Q7) (A)

(Start T4);
(write T4, y, 2, 3);
start(T1);
commit(T4);
write(T1)

2, T4

Database में through, perform होता है
↓
Serve/Total
का apply करता है



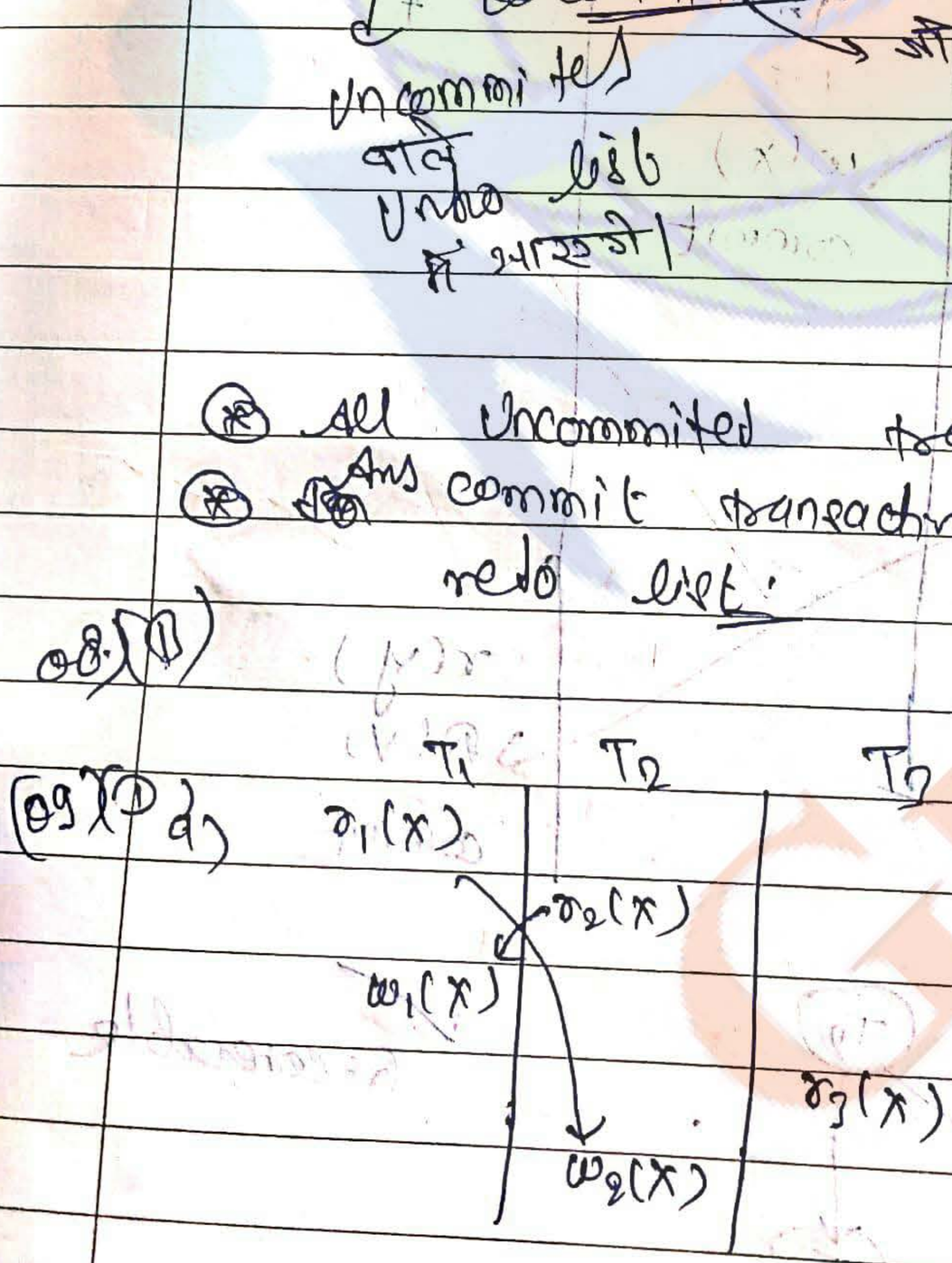
Undo list है :- T1, T3
Redo list है :- T2

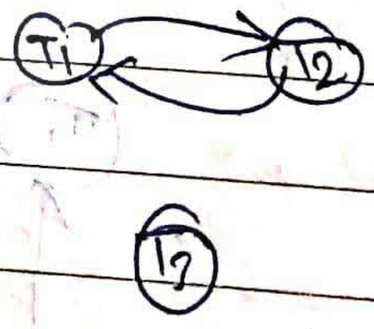
Committed → इन्हें redo से करते हैं।
जो CheckPoint के बाद commit हुए हैं।

Uncommitted वाले Undo list में रहेंगे।

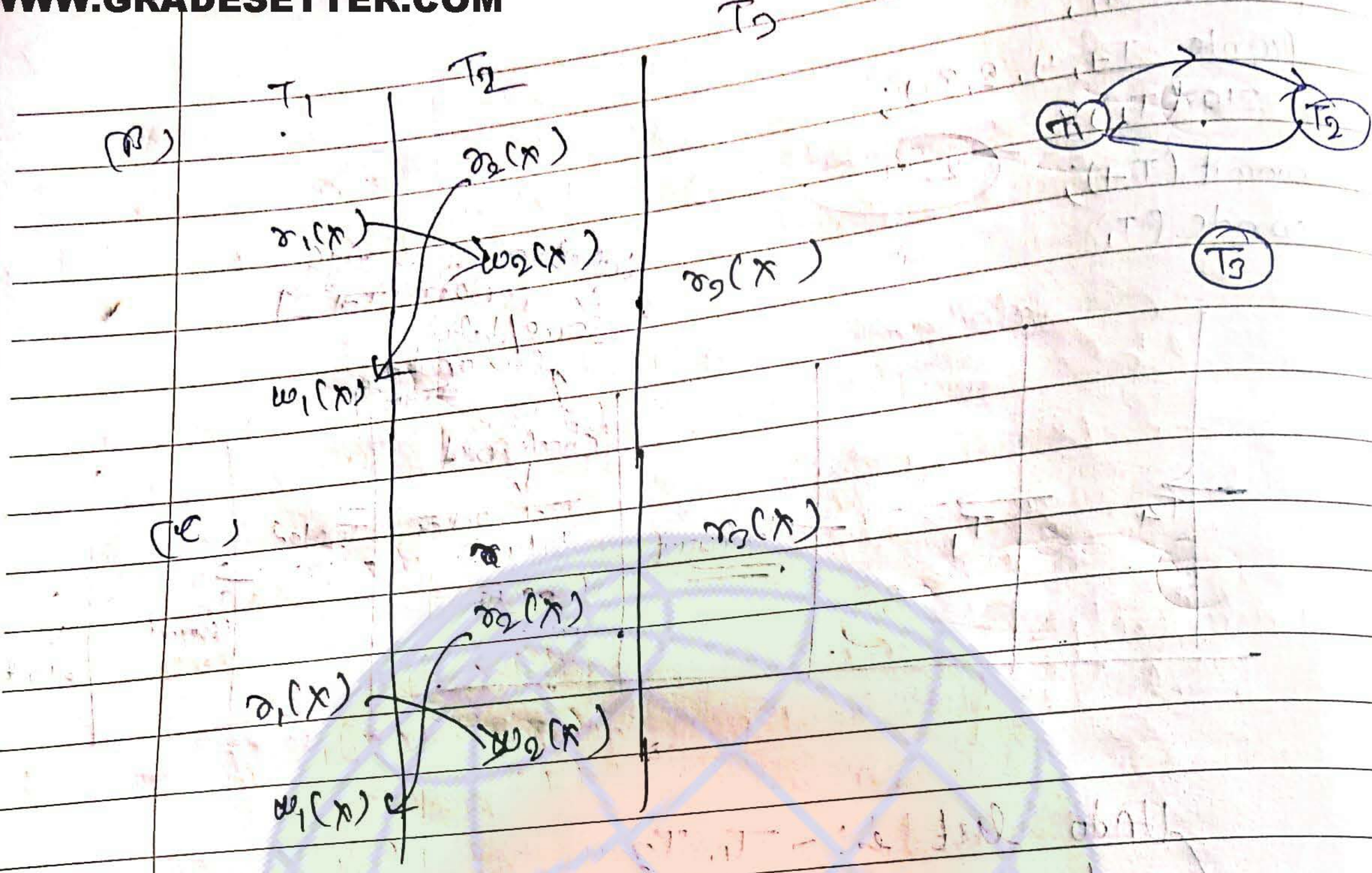
* All Uncommited transaction in "Undo list".
* Ans commit transaction after check point is in redo list.

Q8) (1)

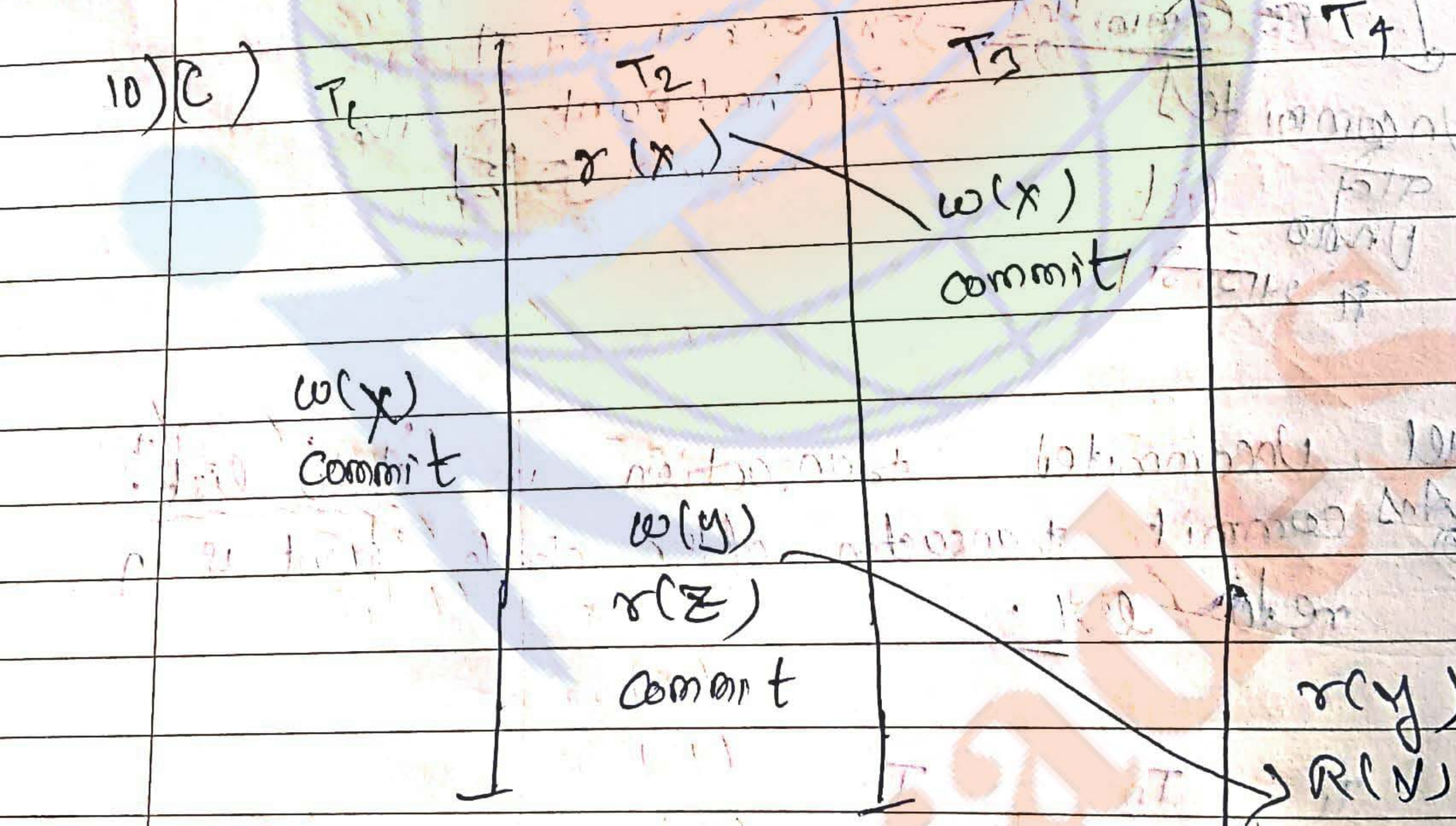


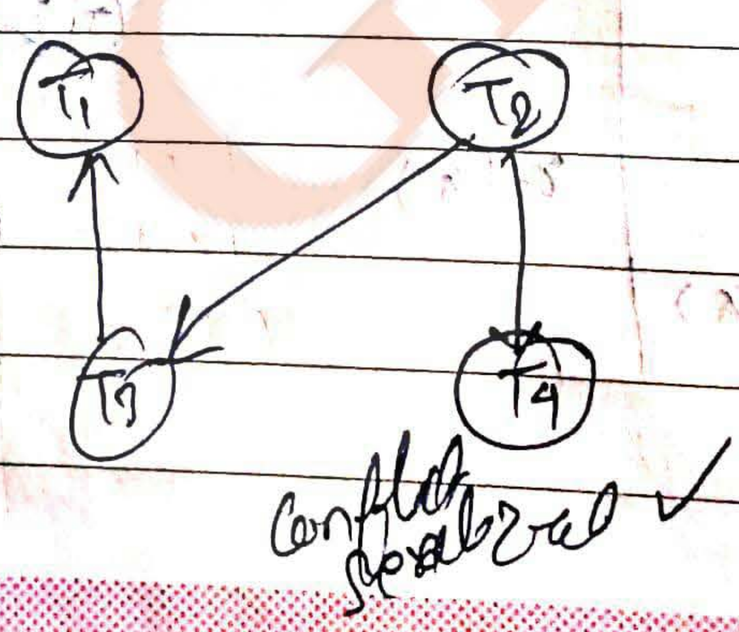


(B)



(C)





Recoverable

Conflict Serializable

Page No.

Date

T_1 T_2 T_3

T_1 T_2

T_3

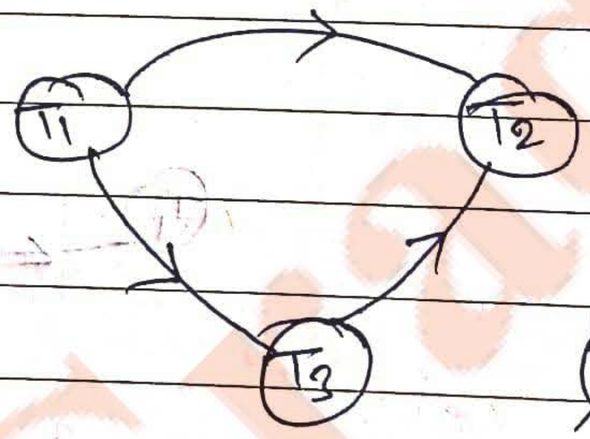
5)

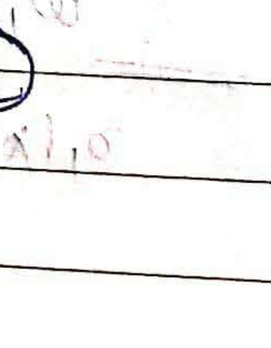
T_1	T_2	T_3
$r_1(x)$		
	$r_2(y)$	
	$r_2(z)$	
	$w_2(x)$	
$r_1(z)$		
$w_1(x)$		
$w_1(z)$		
		$r_3(y)$
		$r_3(x)$
		$w_3(y)$
$r_1(x)$		
	$r_2(y)$	
		$r_3(x)$
$r_1(z)$		
	$r_2(z)$	
		$w_3(y)$
$w_1(x)$		
	$w_2(z)$	
$w_1(z)$		

12) ✗

13) ✗ ⇒ ✓

14) ✗





$T_1 \rightarrow T_3 \rightarrow T_2$

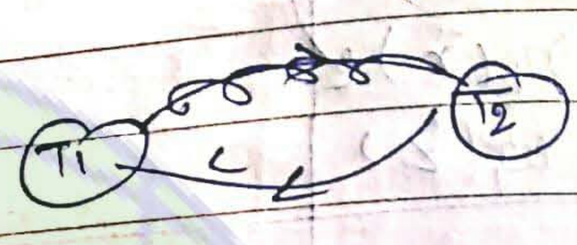
Date

15) (B)

T_1	T_2
$R_1(x)$	
	$R_2(x)$
	$R_2(y)$
$w_1(x)$	
$w_1(y)$	
	$w_2(y)$

S_2:

T_1	T_2
$R_1(x)$	
	$R_2(x)$
	$R_2(y)$
$w_1(x)$	
	$w_2(y)$
$w_1(y)$	

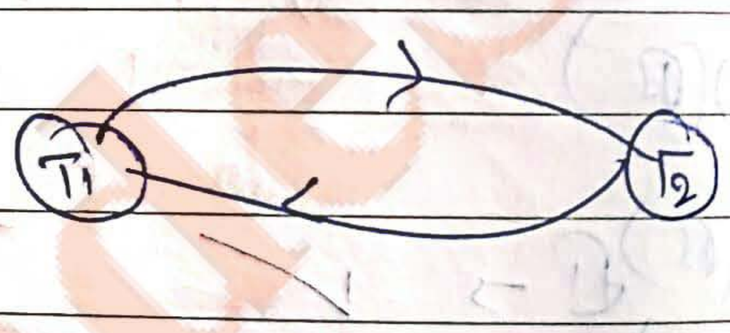


S_3:

T_1	T_2
$R_1(x)$	
$w_1(x)$	
	$R_2(x)$
$w_1(y)$	
	$R_2(y)$
	$w_2(y)$

16) (C)

T_1	T_2
$r_1(x)$	
$r_1(y)$	
	$r_2(x)$
	$r_2(y)$
	$w_2(y)$
$w_1(x)$	
$r_1(x)$	
	$r_2(x)$
	$r_2(y)$
	$w_2(y)$
$r_1(y)$	
$w_1(x)$	



17)

18)

19)

17.) A ✗
B ✗
C ✗
D ✓

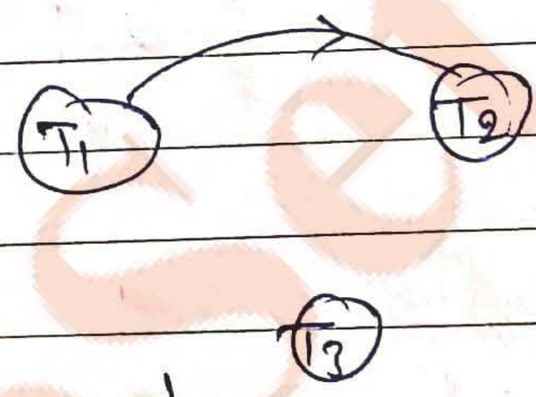
(D)

T_1	T_2
W(A)	
	R(A)
	C

→ irrecoverable

18.)

T_1	T_2	T_3
	R(D3)	
	R(D2)	
	W(D2)	
		R(D2)
		R(D3)
R(D1)		
W(D1)		
		W(D2)
		W(D3)
	R(D1)	
R(D2)		
W(D2)		
	W(D1)	



apply
Directly view serializable

T_2 T_3 T_1 ✗

T_2 T_1 T_3 ✗

T_3 T_2 T_1 ✓

19.) If $TS(T_2) < TS(T_1)$
then
T_1 is killed;
else
T_2 wait

Hashing & Indexing:

- To eliminate unnecessary searching or performing hashing is used.

Data ⇒ Hash table
store ⇒ hash function
retrive ⇒ index

- linear - O(n)
- binary - shorter order (logn)

but we want
O(1)
for that we use hashing

Page No.
Date

* Keys:- 10 5 17 29 38 43
Data:- A B C D E F

H(K) = K % 10

Hash table size = 10

0	10
1	
2	
3	43
4	
5	5
6	
7	17
8	38
9	29

F

h(43) = h(53)

Hash collison

To handle hash collison, we use the following technique :-

I. Open addressing
   - (i) Index Probing
   - (ii) Quadratic Probing
   - (iii) Double hashing

II. Chaining

1) Linear Probing :-

Keys: 10, 15, 17, 16, 25, 28, 30, 39, 45

hash table size = 10

Hash function = K % 10

0	10
1	30
2	39
3	45
4	
5	15
6	16
7	17
8	25
9	28

अगर collison होगा तो next available location पे store होगा

Total collison = 16

2.)

Key: => 15, 7, 25, 35, 45, 79, 47, 23

H.f → (3K+7) % 10

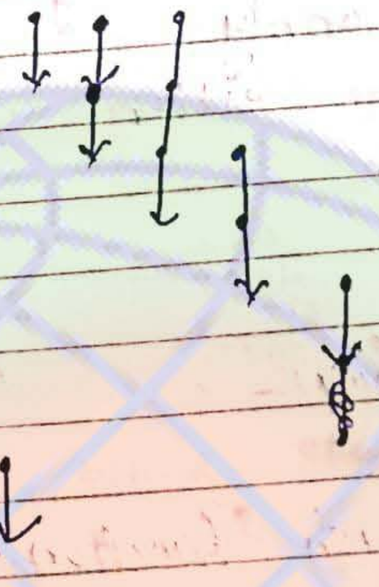
H.T.S = 10

Total collison =

Page No.
Date

Hash collison & searching

0	
1	
2	15
3	25
4	35
5	45
6	39
7	23
8	7
9	47



Total collison = 10

2) Quadratic Probing :-

0th collison 1st collison 2nd collison 7th collison

$P, P+(1)^2, P+(2)^2, P+(3)^2, -----$

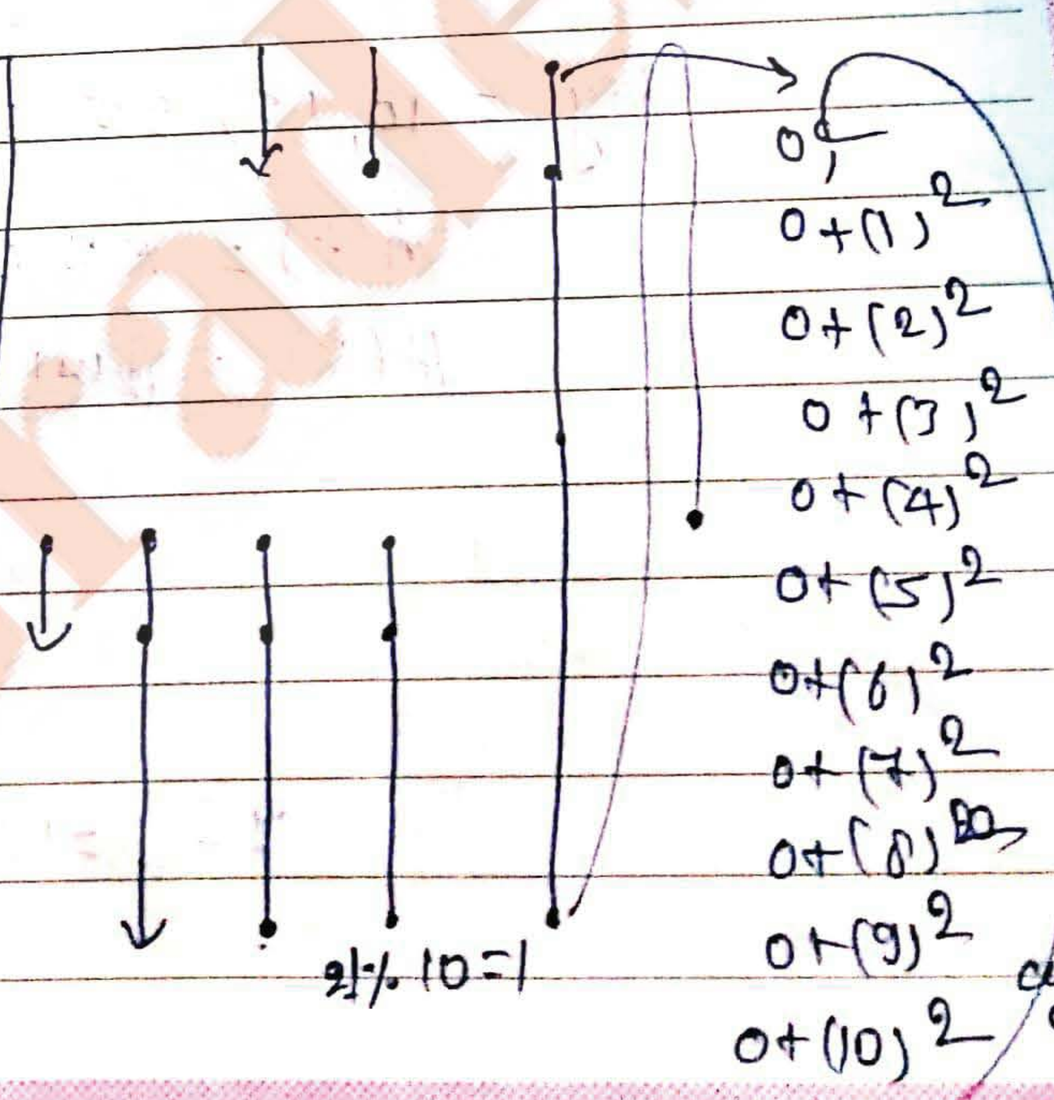
$---------$ $[P+(i)^2 \% \text{ max size}]$

Key : 5, 10, 15, 25, 35, 45, 38, 40

H.T.S = 10

$H(K) = K \bmod 10$

0	0
1	45
2	
3	
4	35
5	5
6	15
7	
8	38
9	25



maximum collision - 16 बार या maximum table size की collision तक allowed है।)

To remove the problem of repeating of the index again (means if cycle occur) apply either Double hashing or chaining.

3) Double hashing:-

main hash function : h(K) → P

to resolve collision : H(K) → P'

collision होने पे इसके अनुसार ही value put करते हैं।

P+P'
P+2P'
P+3P'
⋮
% max size

key = 10, 15, 25, 35, 38, 49, 47

$h(k) = k \% 10$

$H(K) = (2K+1) \% 10$

0	10
1	
2	47
3	
4	
5	15
6	25 — 5+1
7	35 — 5+2*(1)
8	38
9	49

7+5=12

II.) Chaining:

10,(

a) Consi

WWW.GRADESETTER.COM

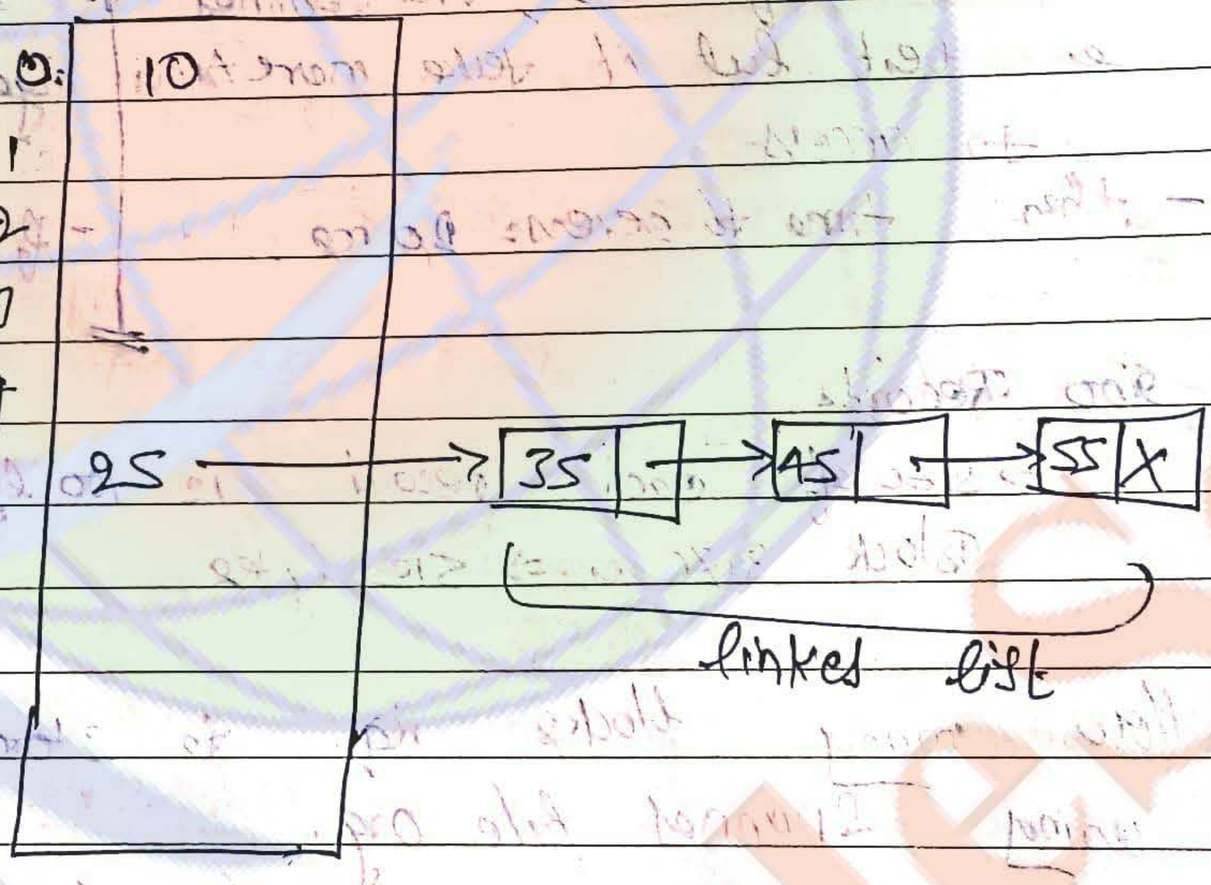
f) Chaining! -

10, 25, 35, 45, 55, ---

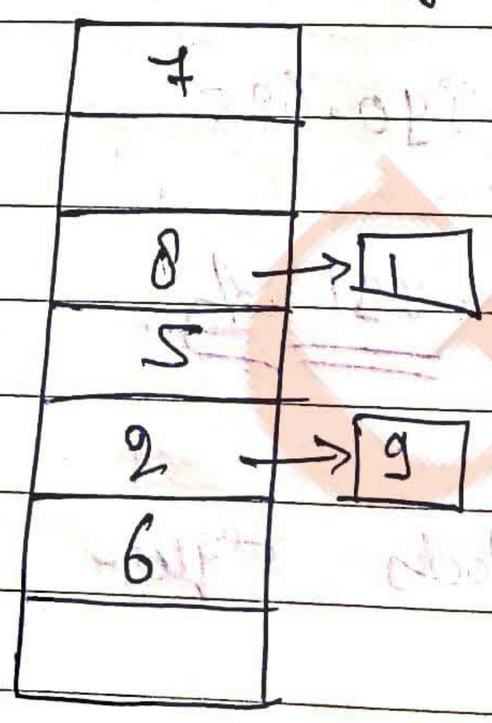
Problem was because of "5"

So to solve the proble

Not solve collide data on same base function



a) Consider the following hash table, It was ~~been~~ been obtained by successfully inserting the numbers 2, 7, 8, 6, 5, 9, 1 into the empty hash table.



which hash function

a) $h(k) = (2+k) \bmod 7$

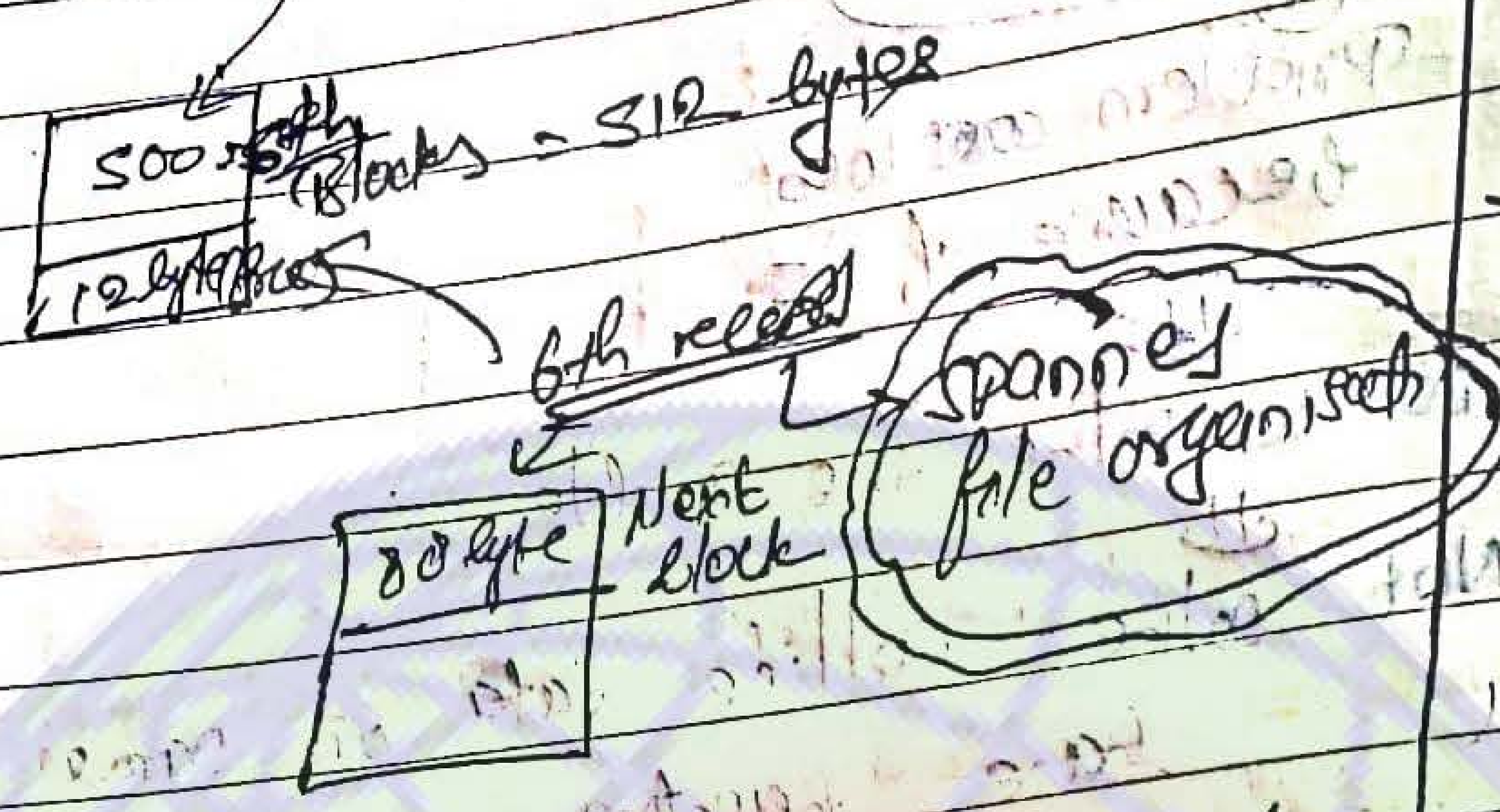
b) $h(k) = 2k \bmod 7$

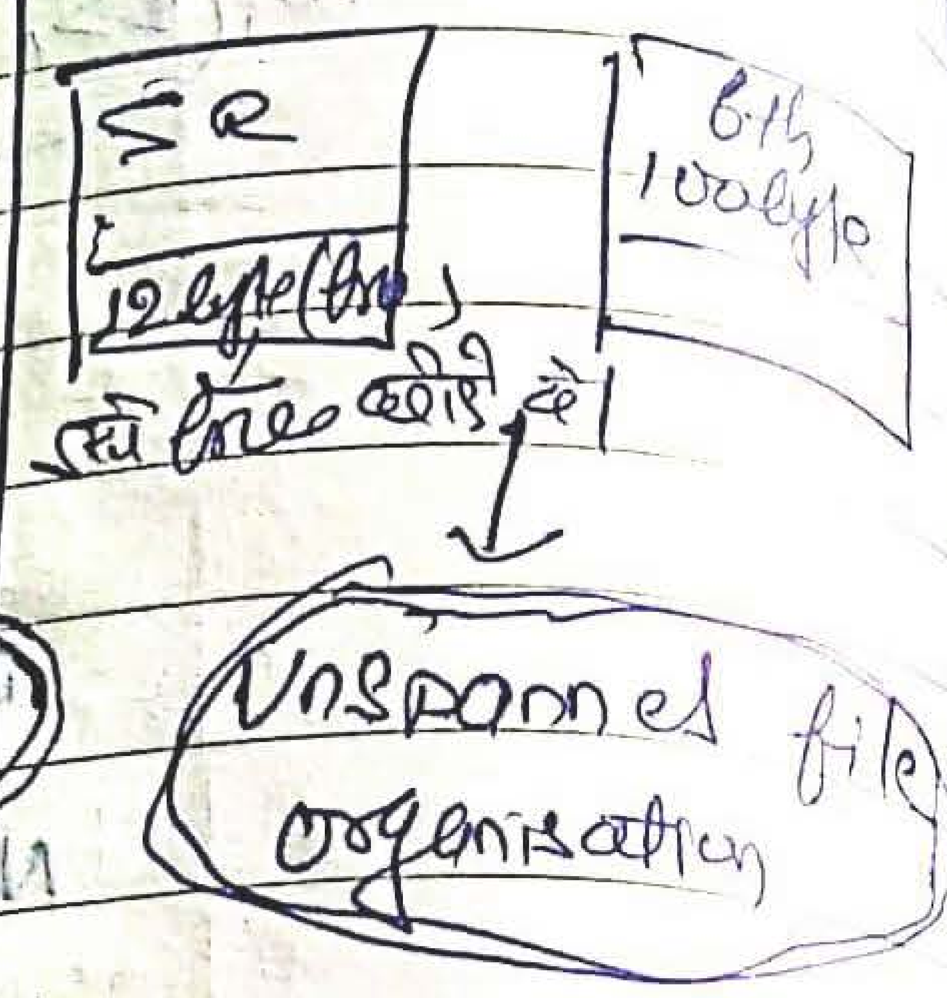
c) $h(k) = \lfloor 0.5k+3 \rfloor \bmod 7$

d) $h(k) = k^2 \bmod 7$

Indexing

1.) Record : 100 bytes





- In terms of size, the spanned is best but it take more time to access.
- then time to access = 20 ms.

- Normally this file organisation techn. is used.
- time = 10 ms

2) 200 Records
 ↳ size of each record is 100 byte
 Block size is ⇒ 512 bytes

a) How many blocks req. to stores all records using spanned file org.
b) " " " unspanned "

sol^n a) $\frac{200 \times 100}{512} = \frac{20000}{512} = 390.625$

$512) 20000 (390.625$
$\underline{1536}$
4640

So, total block = 391 Ans

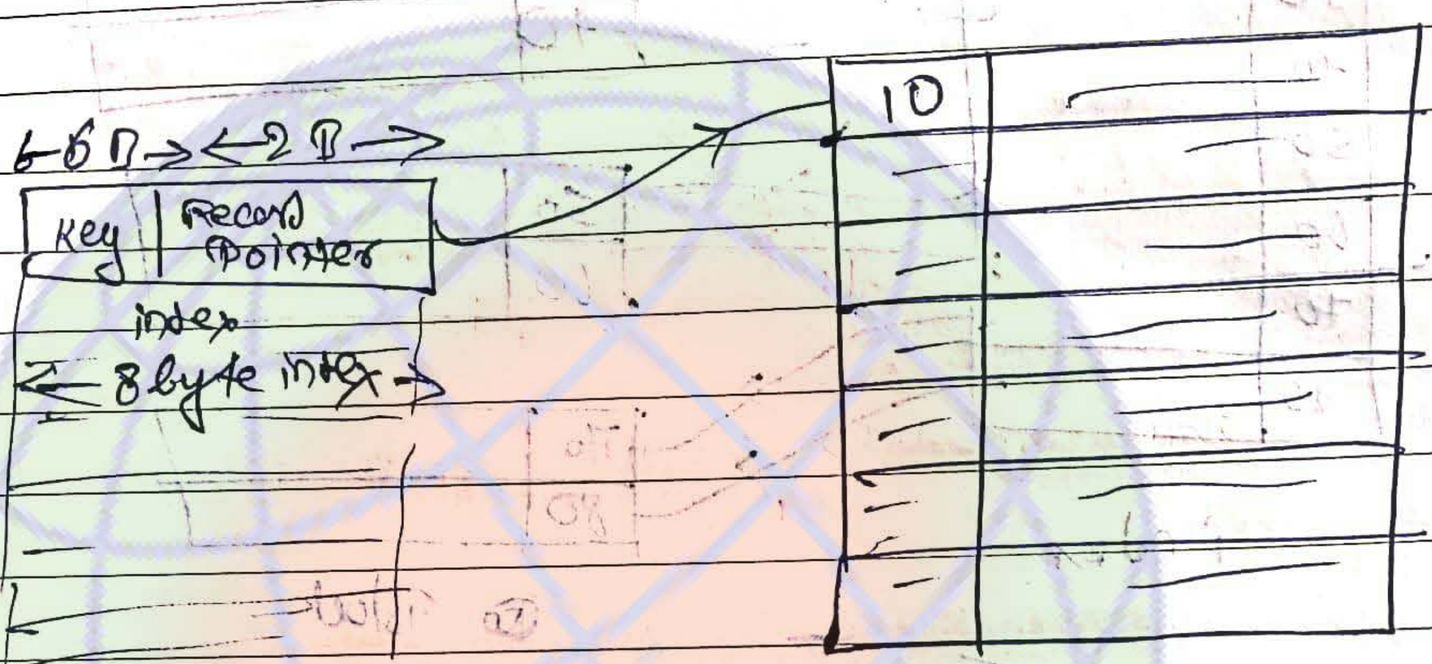
b) $\frac{2000}{5} = 400$ blocks require

6 6 D-
Key
we
which
So
Indexes.
a)
spanned file.
b) Unspanned

Page No.
Date

key	Record Pointer

Index



(we are storing index in a index file) which is stored in hard disk

Indexing index file file size.

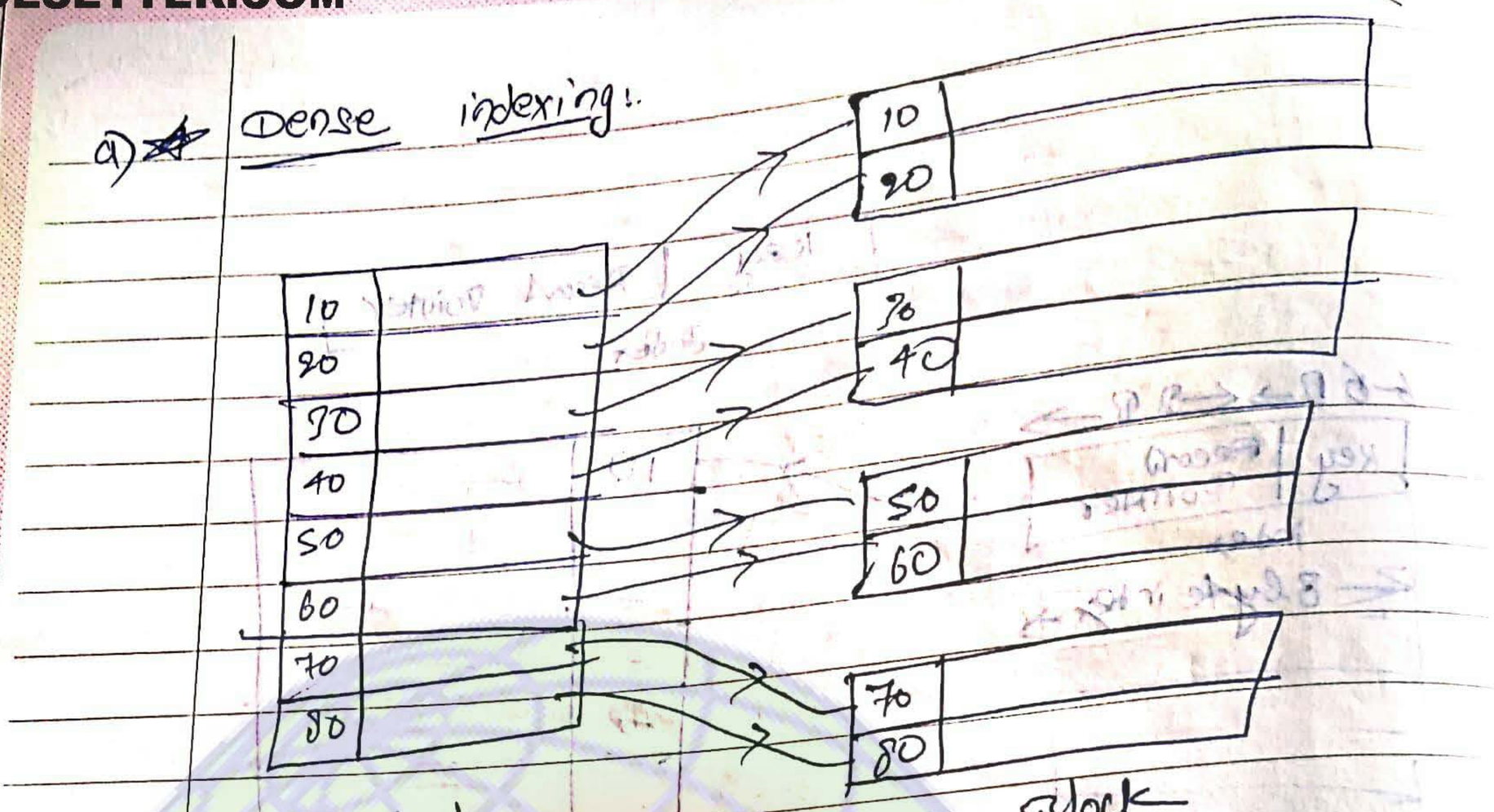
a) $\frac{2000 \times 8}{512}$ (each index file) $= \frac{16000}{512}$ byte $= 31$ blocks blocks required.

spanned file:

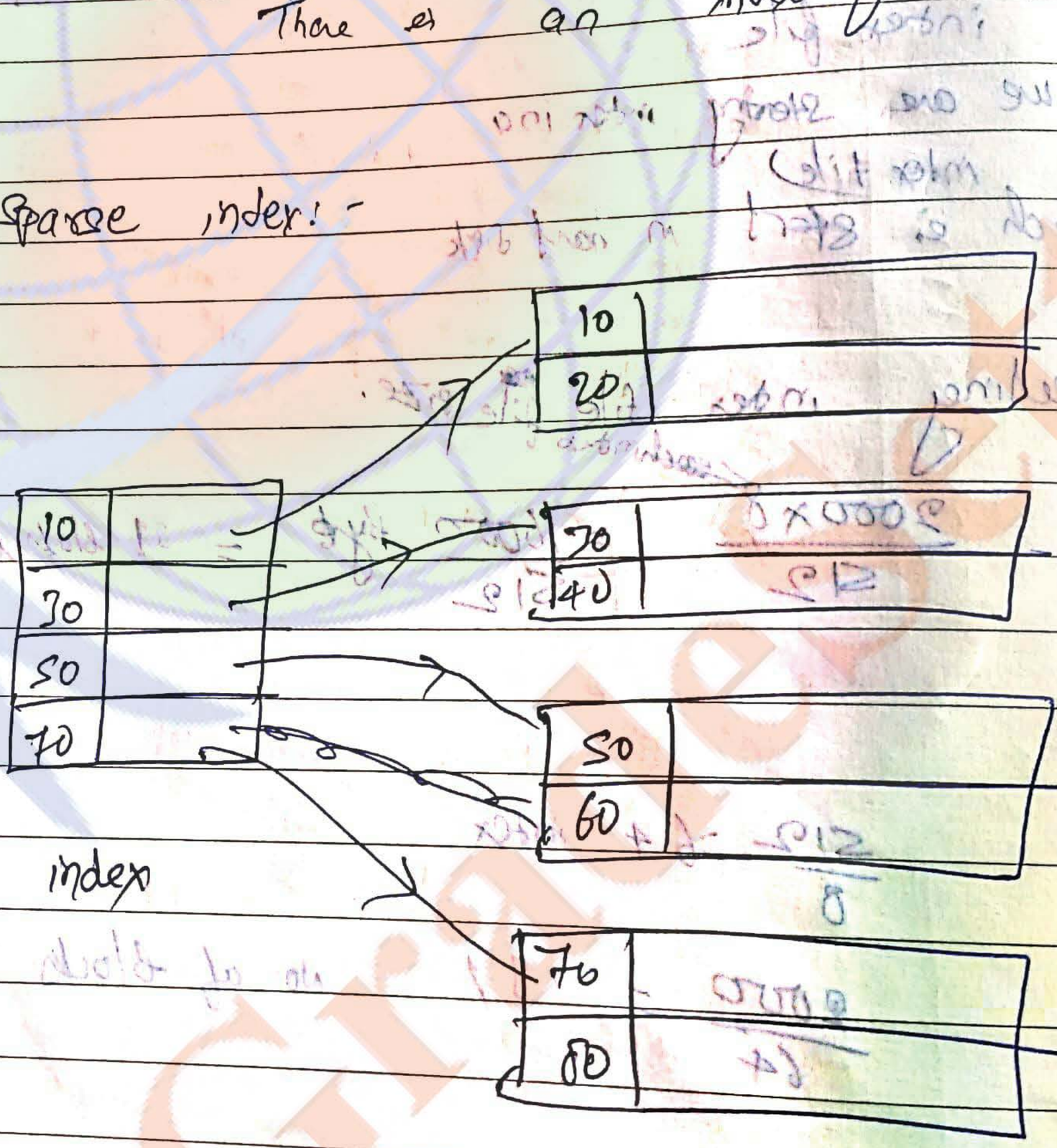
unspanned b) $\frac{512}{8} = 64$ index

$\frac{2000}{64} = 31$ No. of Blocks

a) Dense indexing:-



b) Sparse index:-



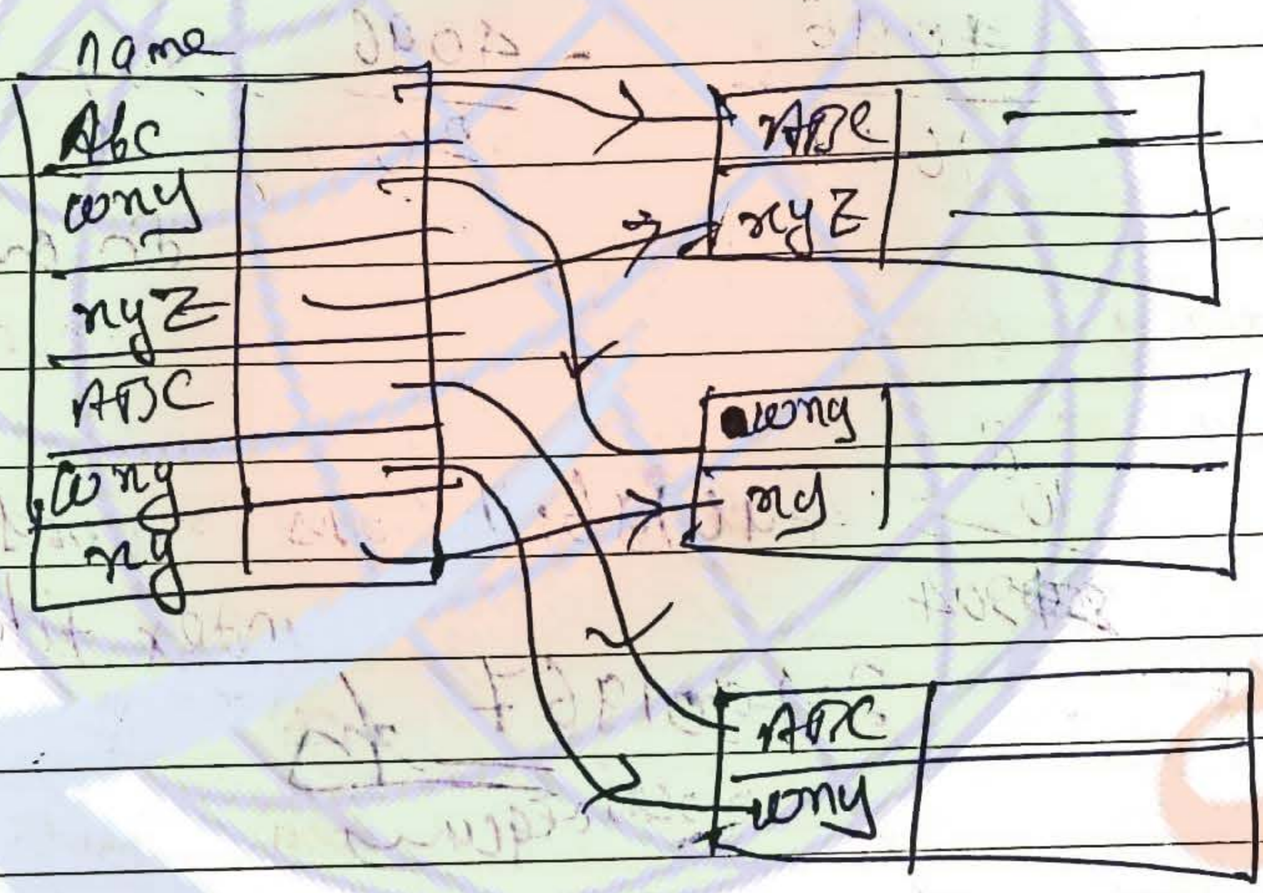
There are different type of index:-

I. Primary index
II. Secondary index
III. Cluster index

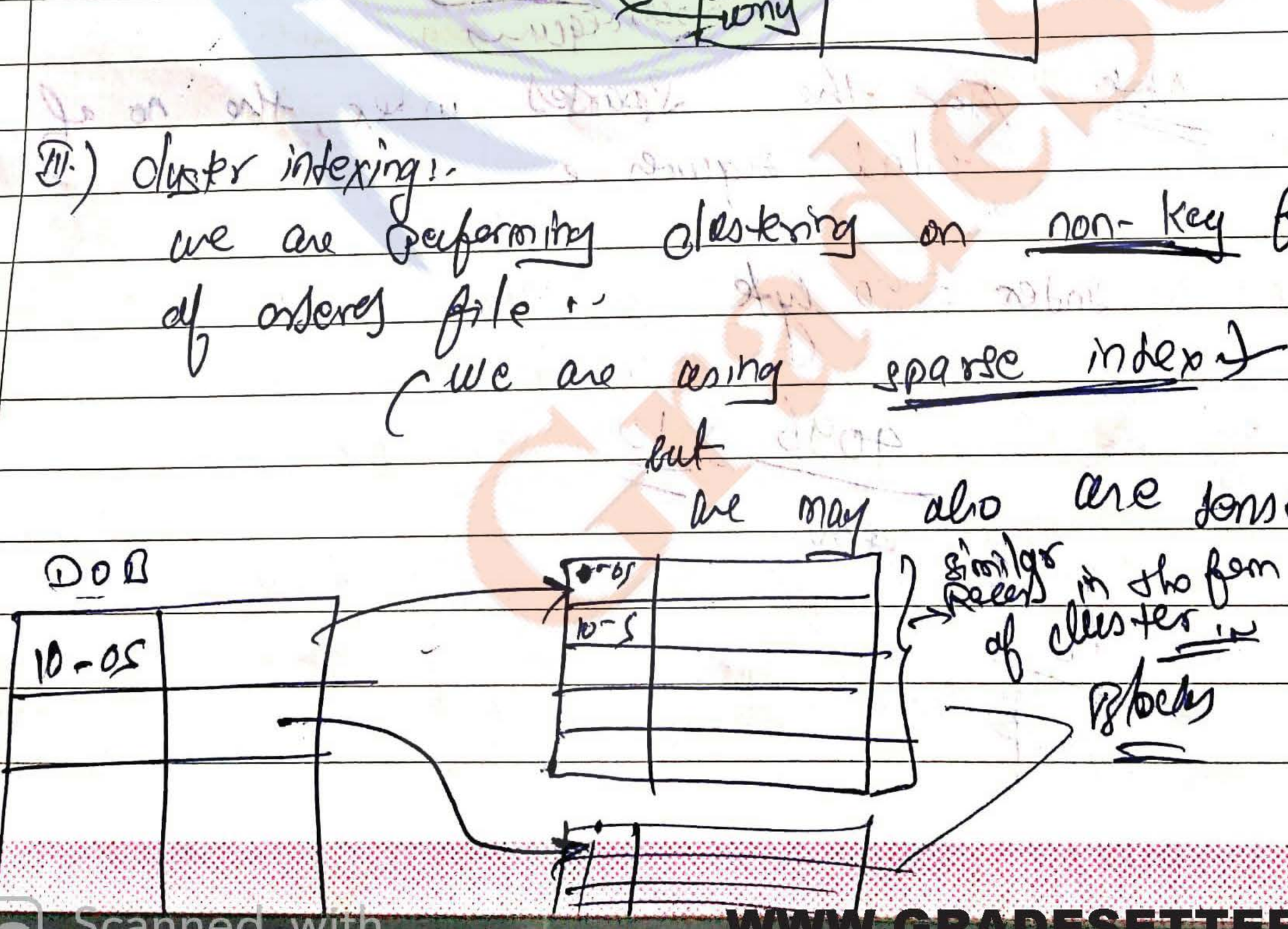
I) Primary:-
on a key fields of ~~ordered~~ ordered file.
(sparse index)

II) Secondary index-
we can perform either on key or non key fields of unordered file.

eg;



III) cluster indexing:-
we are performing clustering on non-key fields of ordered file:-
(we are using sparse index but we may also are dense index



Q.) Consider an ordered file with 100000000 records with a the record size of 400 byte. Stores in a Unspanned file organisation we will assume that the system relates information is stores within a block. Suppose that we construct single level dense secondary index for the file where the search key field is 16 byte long and block pointer is 4 byte long.

So how many blocks requires to store index file. assume block size is 4096 byte.

For Unspanned :-

$$\frac{4096}{16+4} = \frac{4096}{20} = 204 \text{ index}$$

In one block 204 index will come

$$\frac{10^8}{204} = 490196.1 \text{ blocks requires to store index file.}$$

$$\approx 4901967 \text{ blocks requires}$$

Note: For the Spases index, the no of block requires is

Index = 20 byte

$$\frac{4096}{400} = 10$$

$$\frac{1}{10}$$

One level indexing or single level indexing

multilevel indexing (Double level indexing)

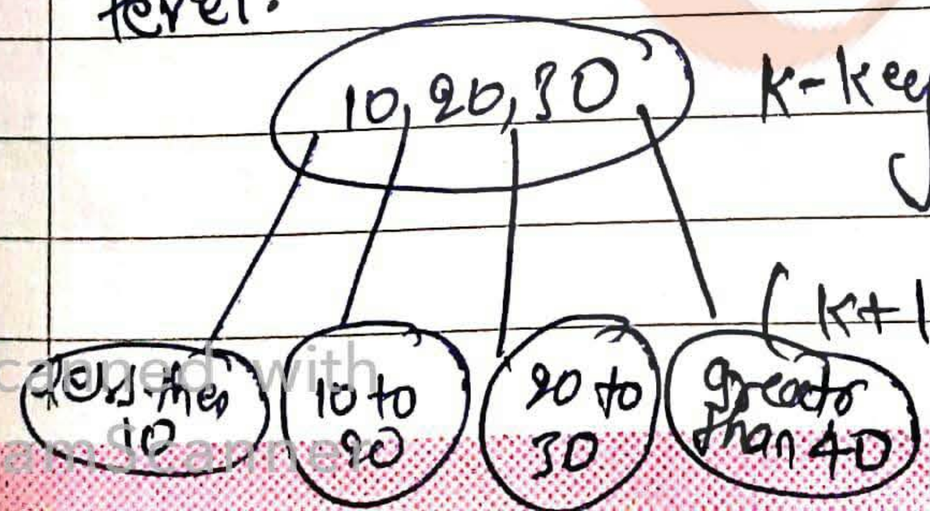
Balanced.

* B tree or B+ tree.

Whenever we want to perform searching on database or secondary memory.
then we can use either B-tree or B+ tree

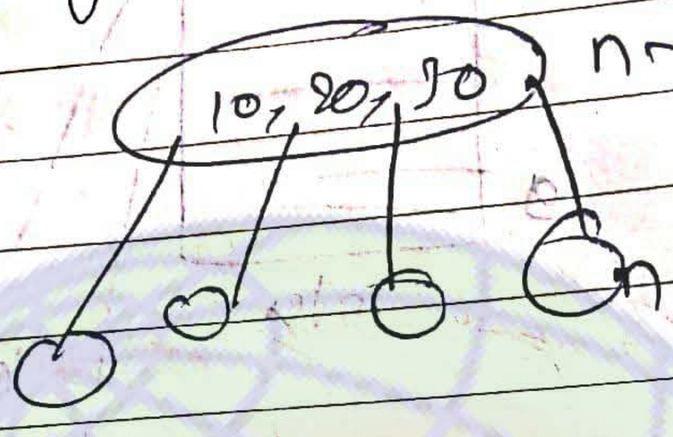
B-tree

B-tree is a multiway search tree, we can store multiple keys in a single node.
In which all leaf node should present at a same level.



Order of B-tree :-

The maximum no. of children can have is



for Non-root node :-

max key $\Rightarrow n-1$

min key $\Rightarrow \left\lfloor \frac{n-1}{2} \right\rfloor$

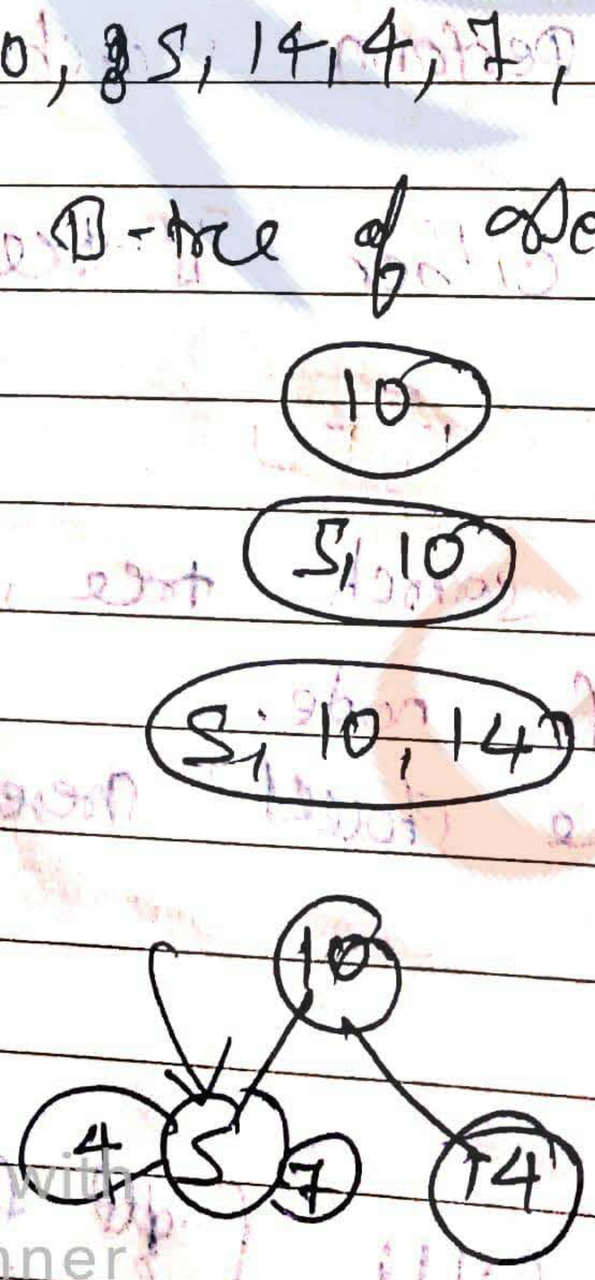
(for root-node)

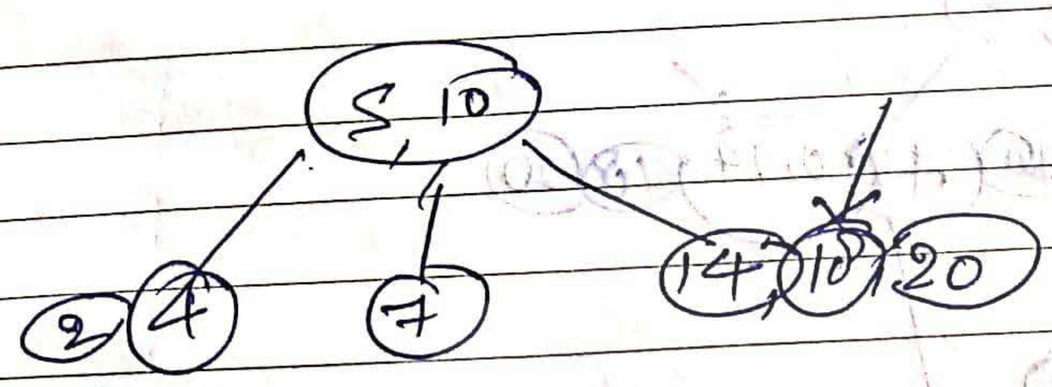
max key = n-1

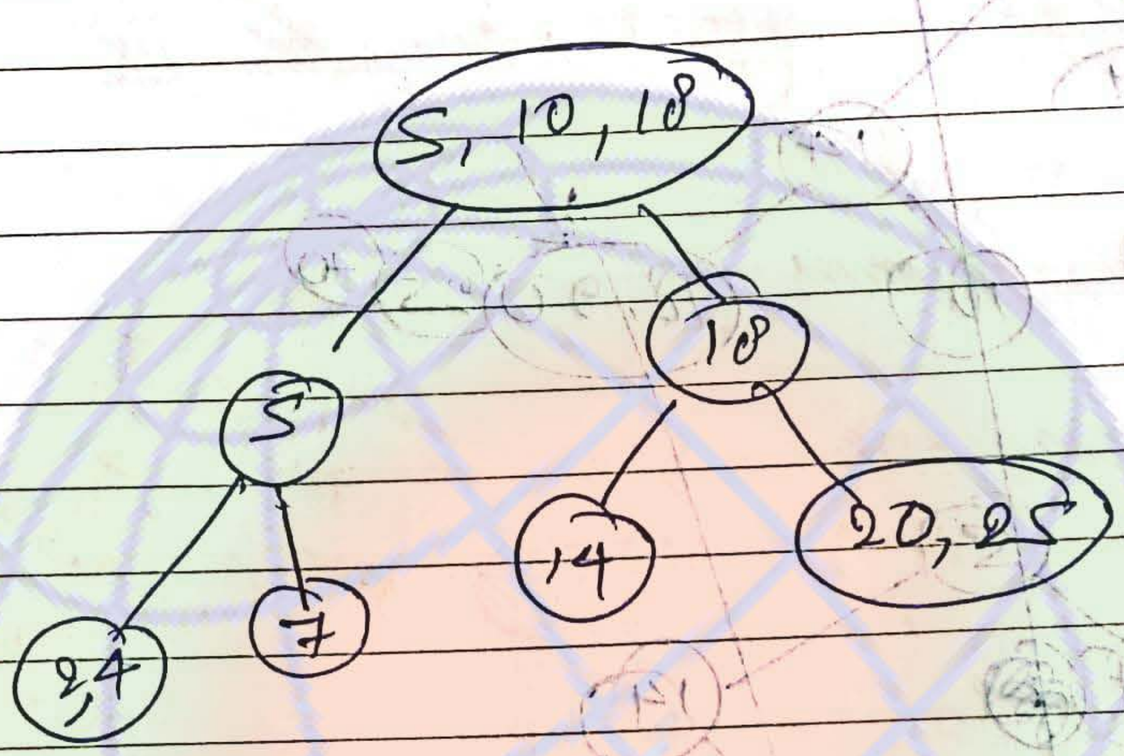
min key = 1

eg.) 10, 5, 14, 4, 7, 2, 18, 20, 25

B-tree of order 3 :-



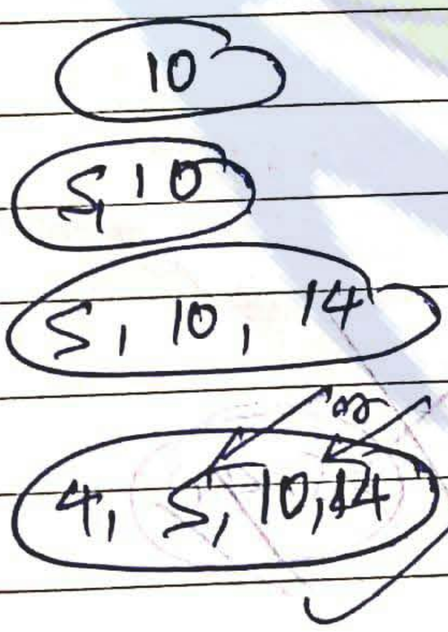


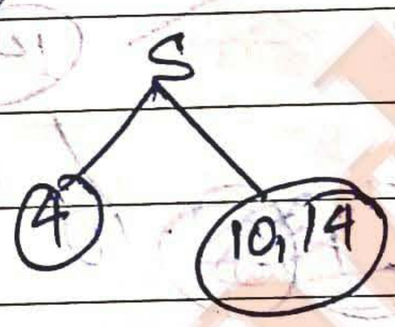


q.) 10, 5, 14, 4, 7, 2, 18, 20, 25, 15, 40, 1, 9

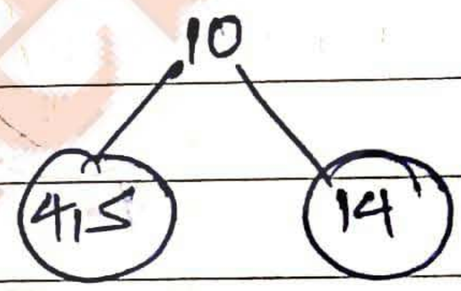
order of 4

sol.)



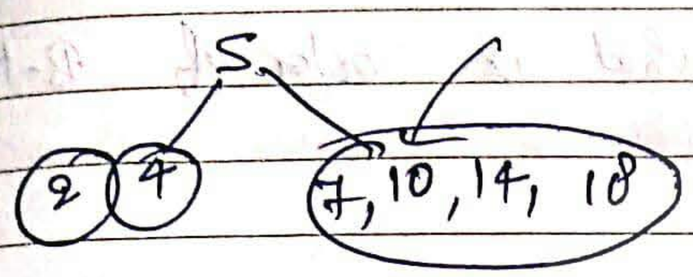


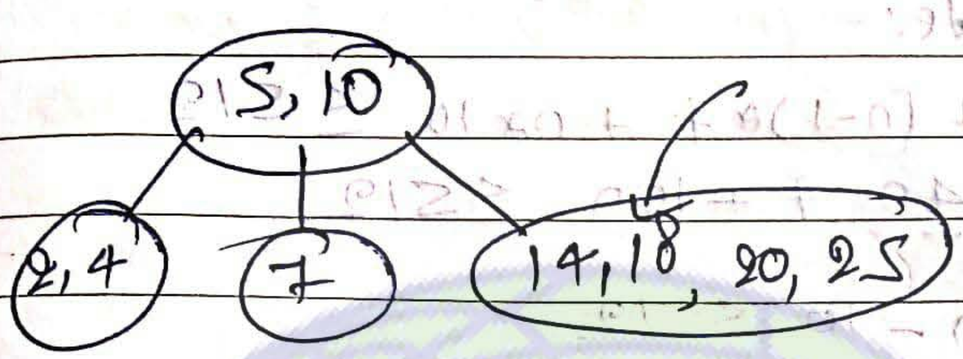
right Biasing (right side weightage जाएगा)

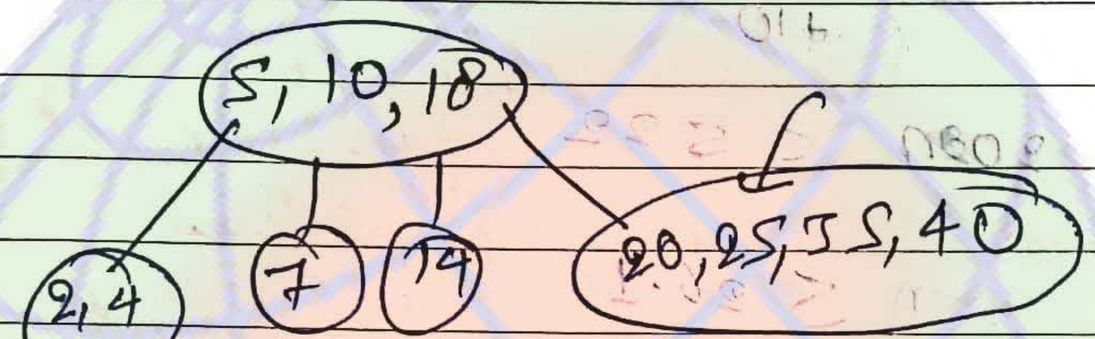


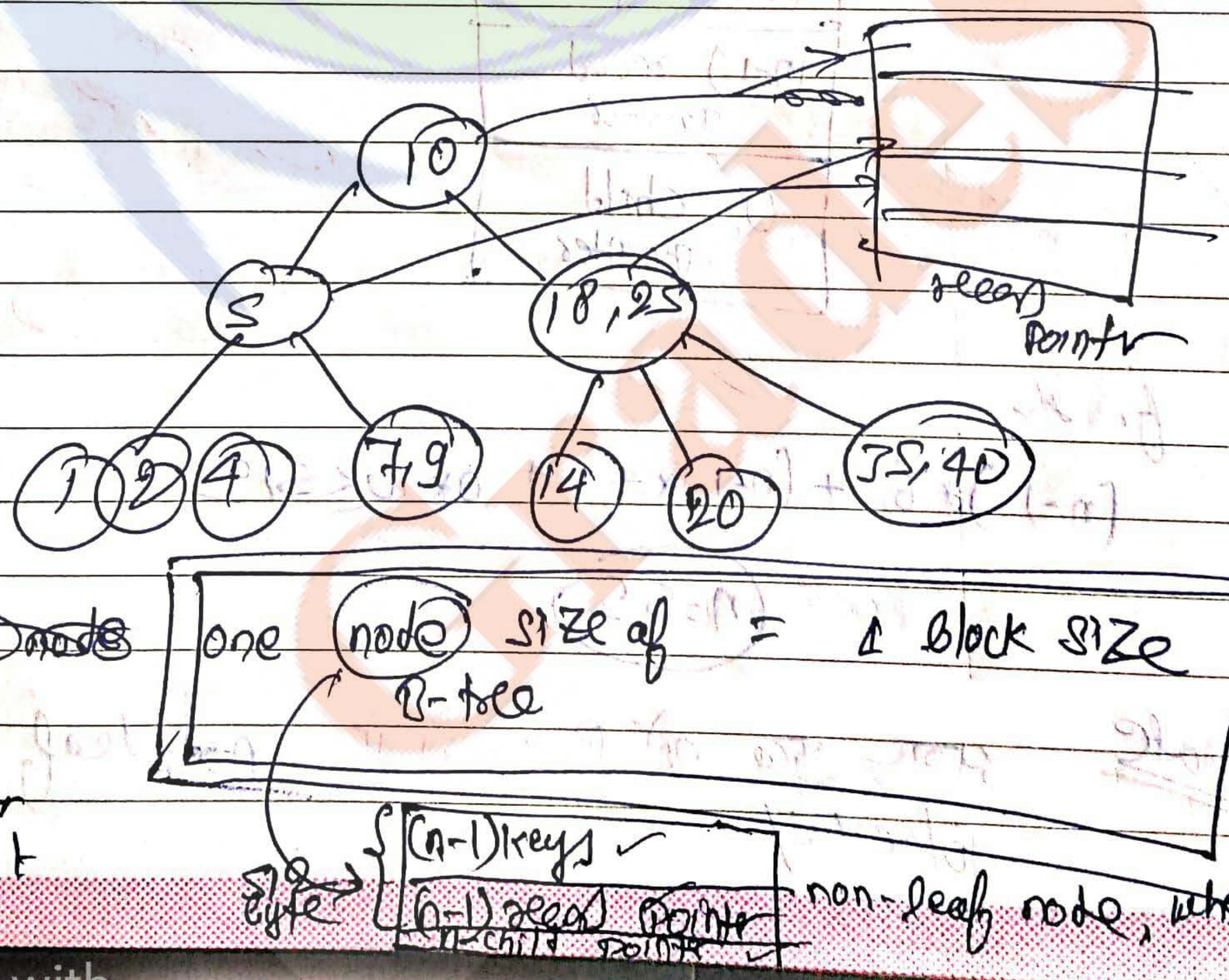
left Biasing.

sol.) design the tree, according to weight biasing -









one node size of = 1 block size
B-tree

(n-1) keys
(n-1) record pointer
(n) child pointer

type { (n-1) keys, (n-1) record pointer, n-child pointer } non-leaf node, where n = order of B-tree.

Q) key = 6 byte
R.P = 4 byte
C.P = 10 byte

What is order of B-tree

Sol for non-leaf nodes —

$(n-1)*6 + (n-1)*4 + n*10 \leq 512$

$6n-6 + 4n-4 + 10n \leq 512$

$20n - 10 \leq 512$

$20n - 10 \leq 512 + 10$
$+10$

$20n \leq 522$

$n \leq 26.1$

$\boxed{n = 26}$

b) for leaf node:

(n-1) keys
(n-1) record pointer
0 child pointer

for this

$(n-1)*6 + (n-1)*4 + 0*10 \leq 512$

$n = 52$

Note अगर कुछ नहीं दिया हो तो non-leaf के according solve करो।

Page No.
Date

Q) key size = 16 byte
Record pointer size = 4 byte
child pointer size = 8 byte
Block size = 1024

What is the order of B-tree per leaf node?

Sol^n $(n-1)16 + (n-1)4 + 0*8 \leq 1024$

$16n - 16 + 4n - 4 \leq 1024$

$20n \leq 1044$

$n \leq 52$

Ans ~~maximum no.~~ Now, if the definition of order will change.
order:- the maximum no. of pair of key and record pointe

Ans $\max(key, R.P)$

$S2-1 = S1$ A

★ Indexing with B+ tree:-

It is also multiway search tree, in which all leaf node, should at same level.

for non-root:- max key = $n-1$
min key = $\lfloor \frac{n-1}{2} \rfloor$

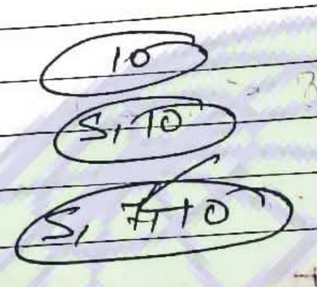
for root:- max key = $n-1$
min key = 1

eg) * 10, 5, 1, 2, 14, 7, 8

order = ?

we want to design B+ tree of order 3.

Sol

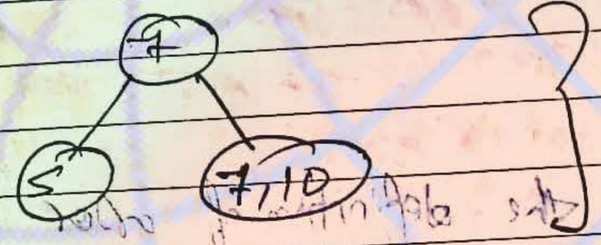


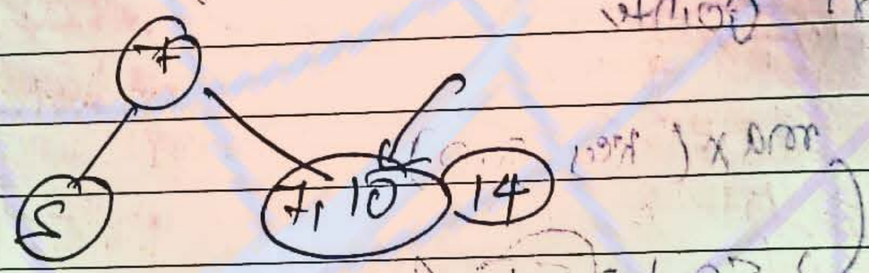
leaf node

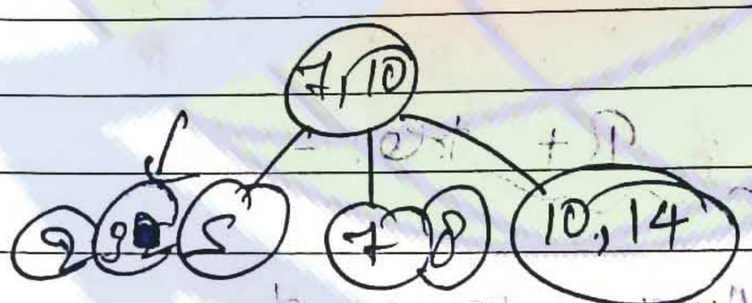
Note: (B+)

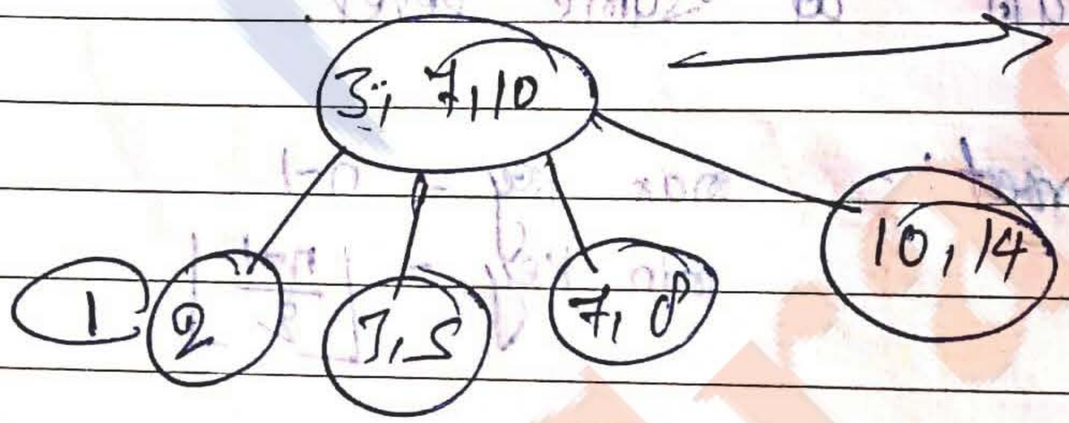
when ever leaf node is splitted.

The value reflecting Right hand side node



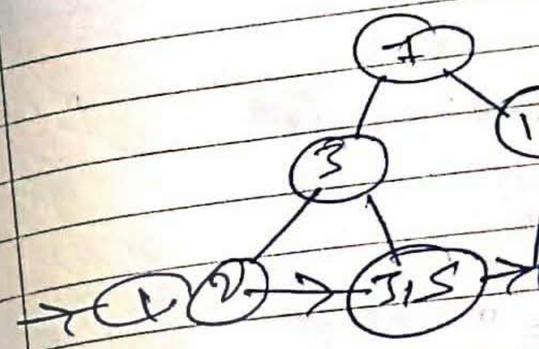






यहाँ पर value right में नहीं आएगा, split करते वें

एक child को split करते वें value आता है



Note • In B+
All key's are
• All leaf a
• All record
• leaf node

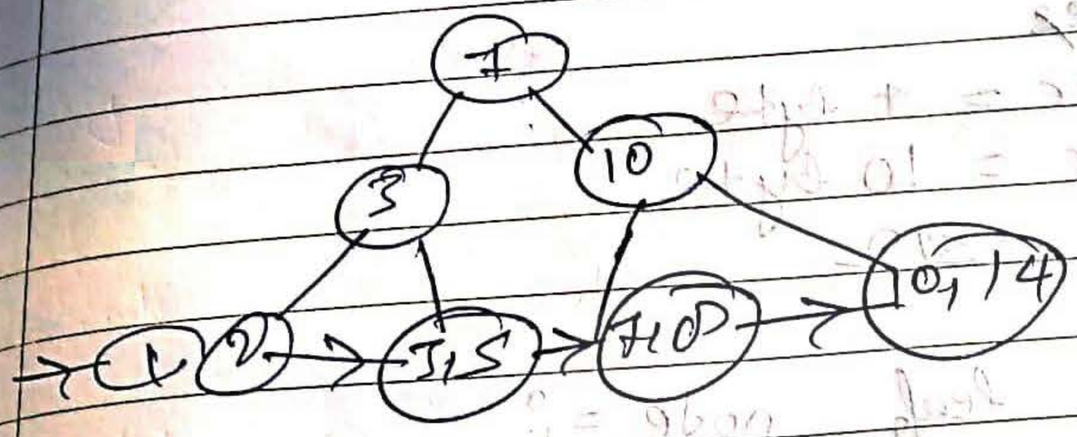
Comparison

B-

- for single
B-tree is

- Height

Non-leaf



Note: • In B+ tree
All key's are present at the leaf node.
• All leaf are connected via linked list.
• All record pointers available at a leaf node.
• leaf nodes are in sorted order.

Q.No Comparison b/w B & B+ tree

B-tree	B+ tree
- for single value searching B-tree is best.	- for range wise searching (eg rollno 10 to 20) is best on B+tree.
- Height is less than B+	- Height of B+ tree more as compare to B-tree

Q.No Non-leaf node of (B+ tree.)

(n-1) key
0 record pointer
n child pointer

Leaf node of B+ tree:-

(n-1) key
(n-1) record pointers
1 child pointer.

eg.) Key = 6 bytes
Record pointer = 4 byte
child pointer = 10 byte
Block size = 512

order for leaf node = ?
order for non-leaf node = ?

sol) for non-leaf node,
$(n-1)6 + 0*4 + n*10 \leq 512$

$6n - 6 \leq 512$ $n \leq 32$

$6n \leq 518$

$n \leq 518/6$

$n \leq 86$

for leaf node:-

$(n-1)6 + (n-1)*4 + 1*10 \leq 512$

$6n - 6 + 4n - 4 + 10 \leq 512$

$n \leq 51$ Ans

Q) B+ tree: Consider leaf file of 16784 records

16784 records
4 72 byte long
key = 6
block = 10
block size = 1024

16704 nodes

Page No.
Date

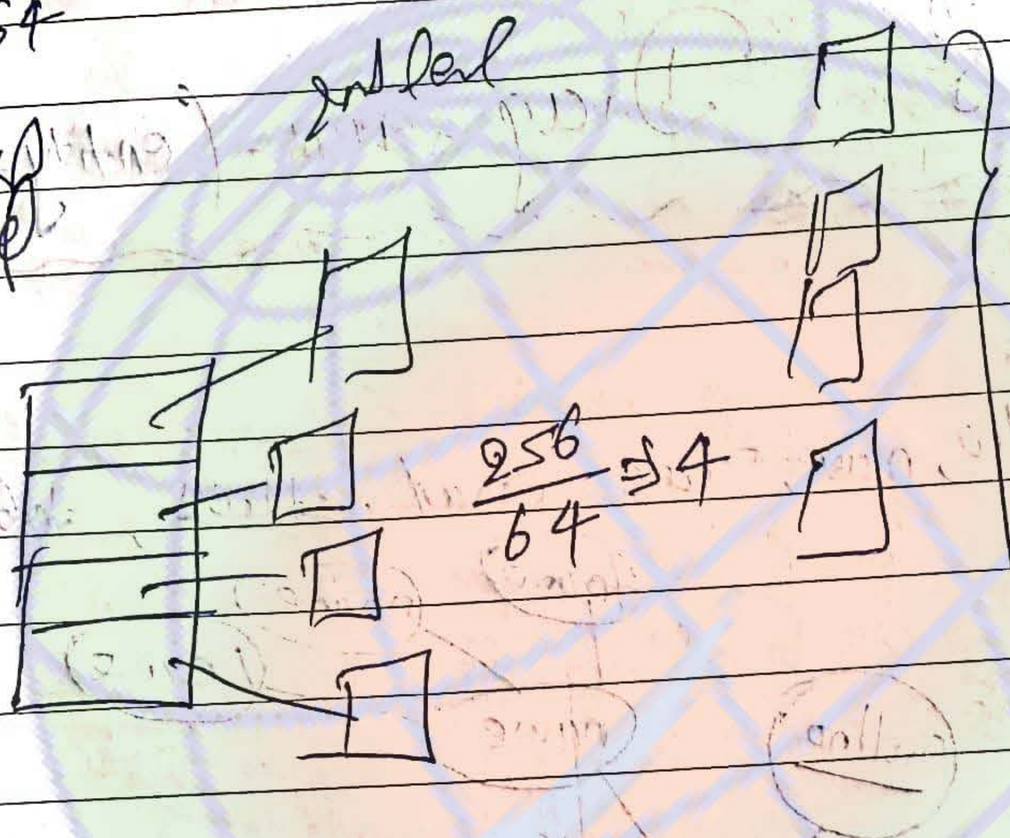
16

$\frac{1024}{16} = 64$

$\frac{6384}{64} = 256$

16384

bt level



64+16 bytes

$\frac{1024}{32} = 32$

512

1

Q15

$(n-1)*8 + (n-1)*2 + 6 \leq 1024$

$n \leq 64$

63 order

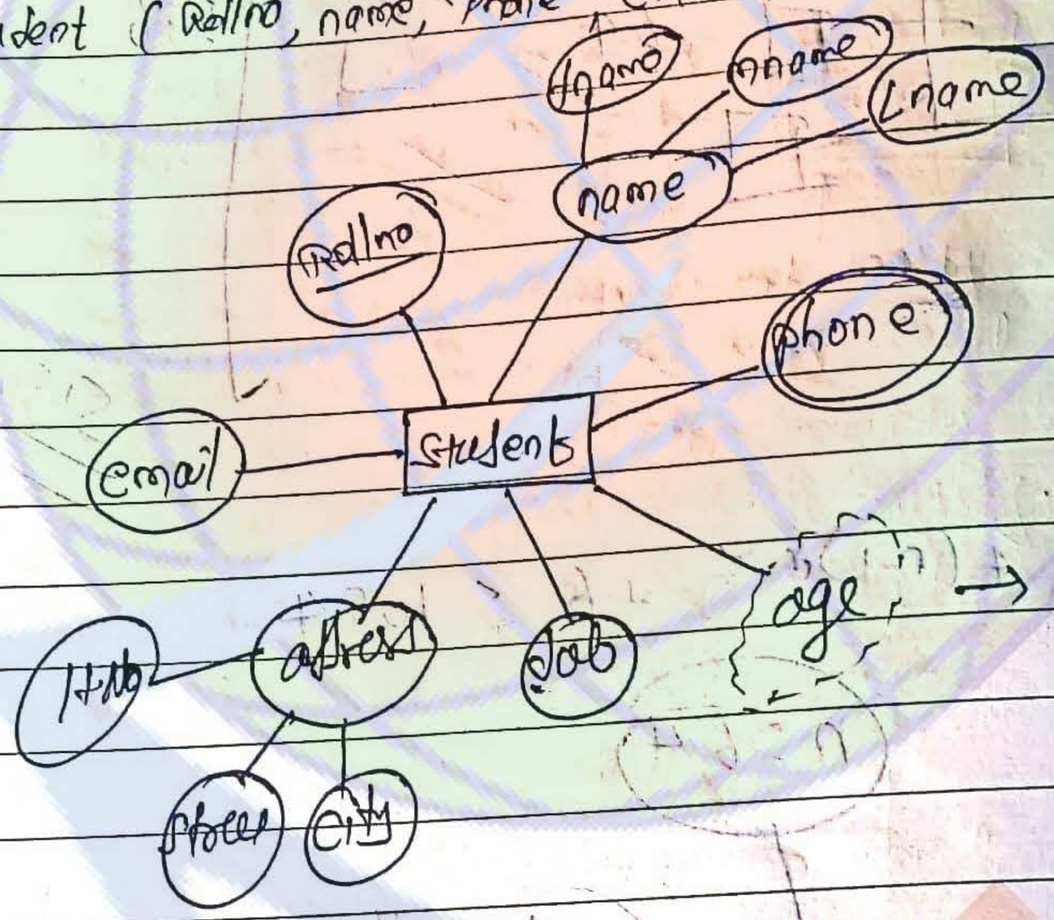
bear pairs it can hold.

64+63 Ans

05) Ost tree
key = 12 byte
Block size = 1024
Rec. Point = 10
Block point = 8

E-R Diagram :- (entity-relationship diagram)

Student (Rollno, name, Phone, email, address, dob, age, marks)



age → यहाँ पर dob है। इसलिए age derived attribute होगा

(1) Single value :
(11) multivalued :
(111) Composite :
(IV) Derived :-

Student m

Note :- Binary relation
relation
two ent
ie known
relation

There are

(I) Unary Relati
(II) Binary relat
(III) ternary rel
(IV) n-ary

E1

(arrow represents one)

E1

E1

E1

Page No.
Date

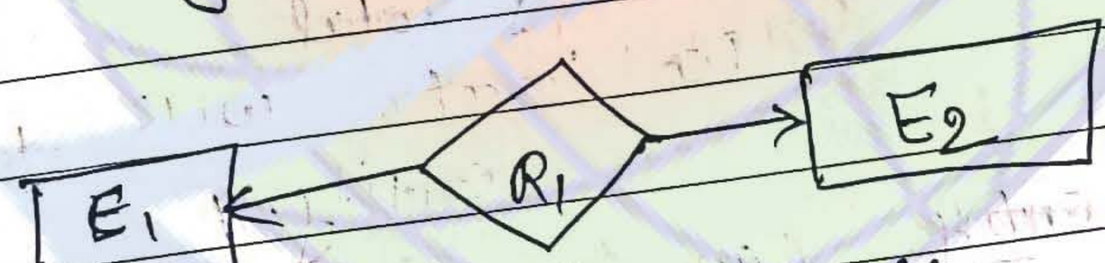
Student — m — enrolled — n — Course (many to many)

Note: → Binary relation - relation b/w two attribute is known as binary relation.

(I) one to one relation
(II) one to many
(III) many to one
(IV) many to many.

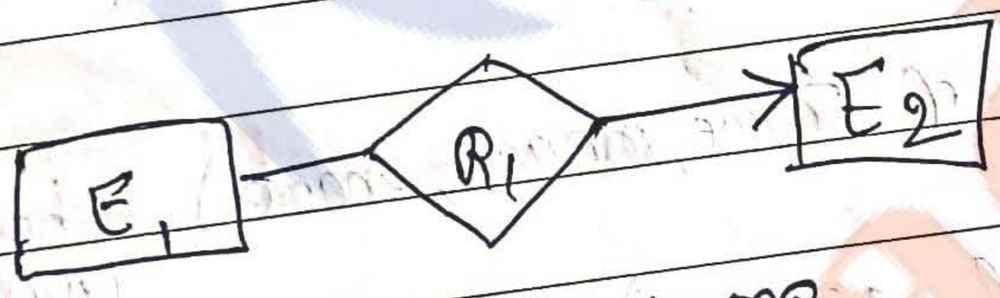
There are multiple types of rel'

(I) Unary Relati- entity related to itself
(II) Binary relation - relation b/w two
(III) ternary rel' - rel' b/w three entity
(IV) n-ary relation - rel' b/w n.



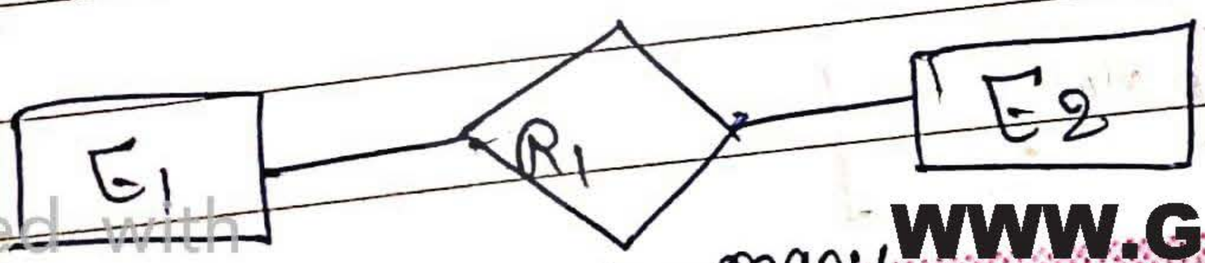
one to one rel'

(arrow represent one)



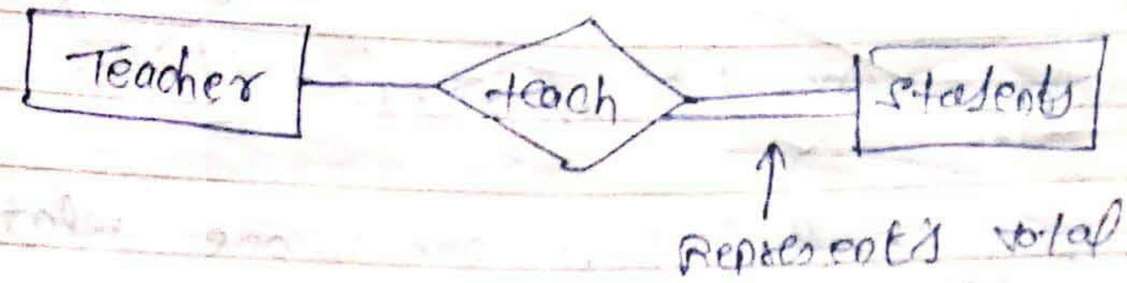
many to one

one to many



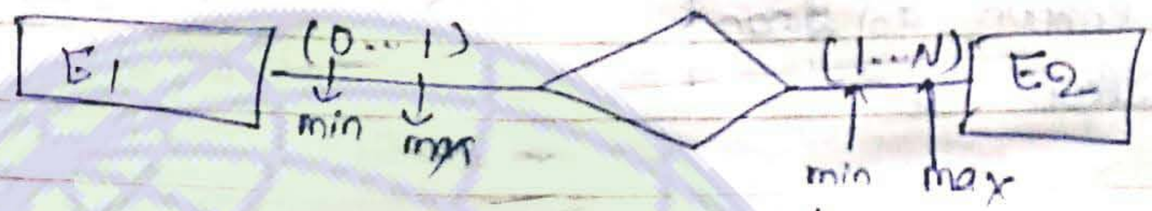
many to many

* Total participation.

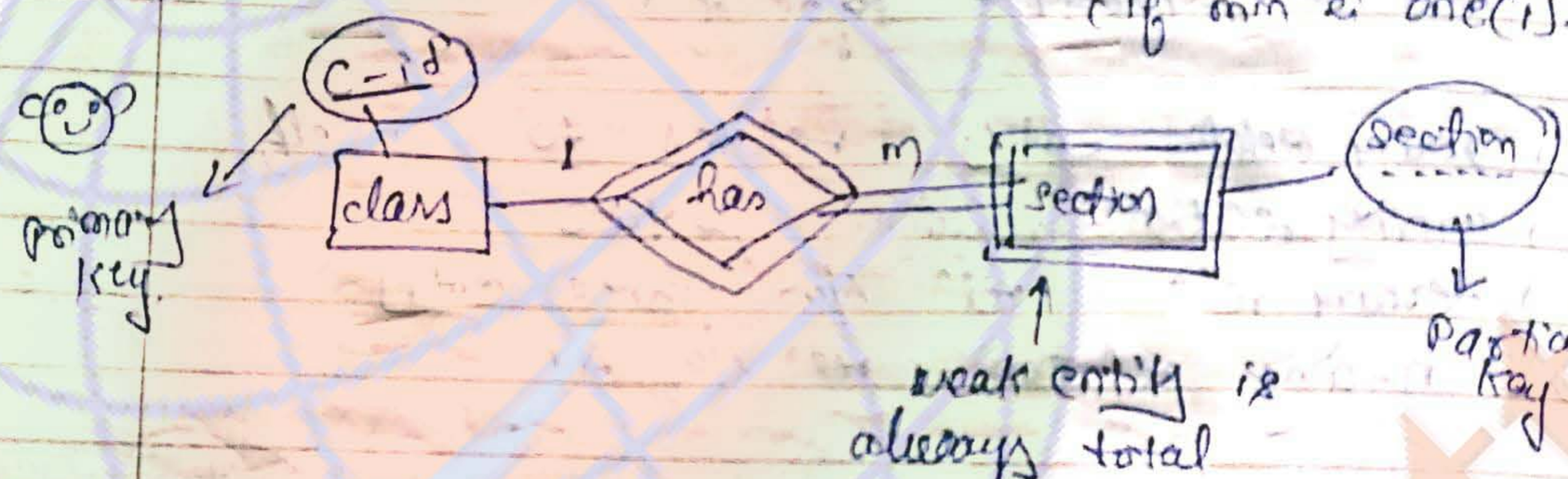


Represents total participation.

* Cardinality



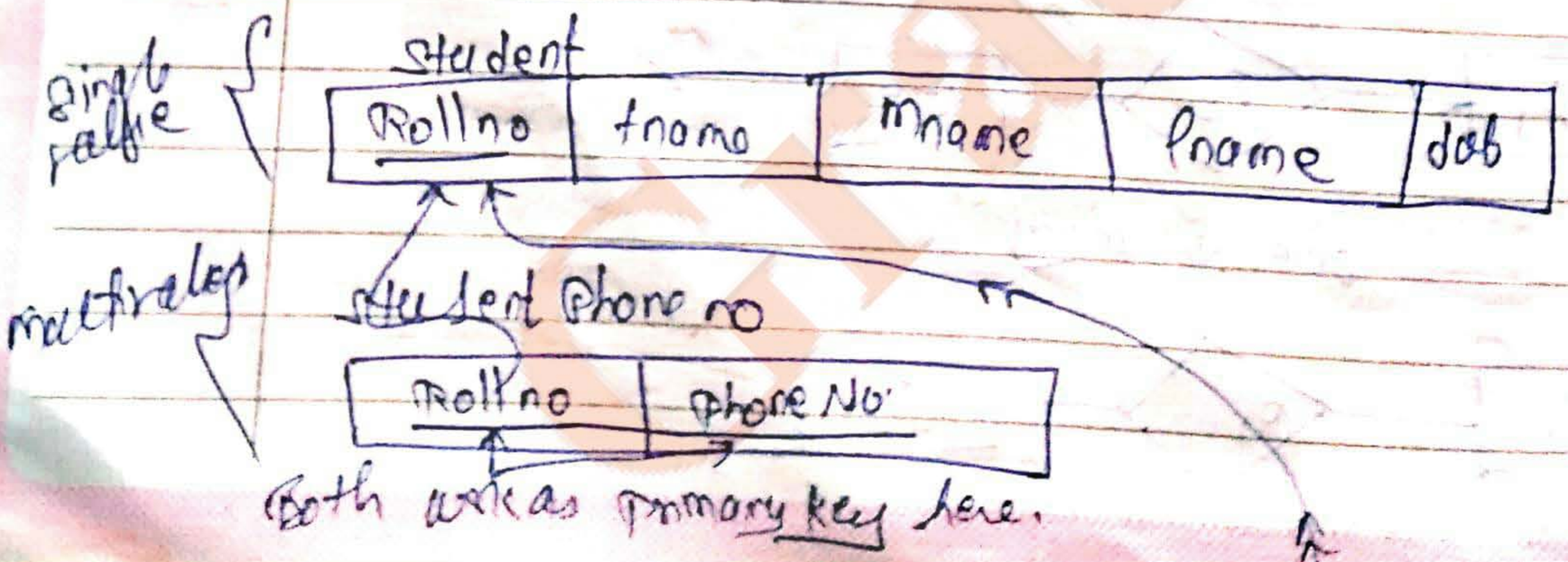
Represents total participation if min = one(1).

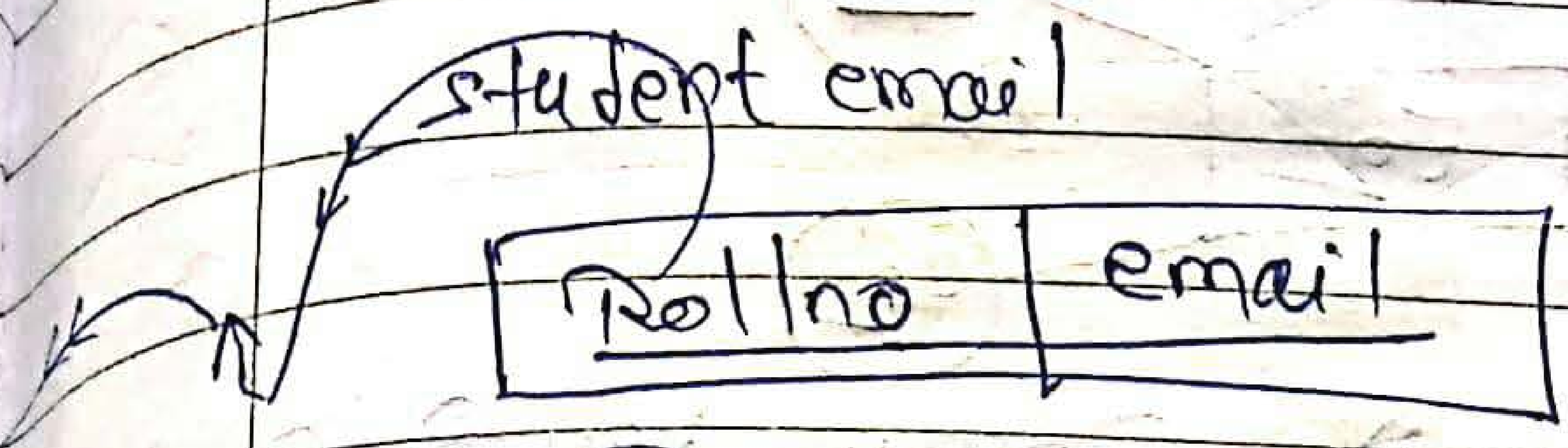


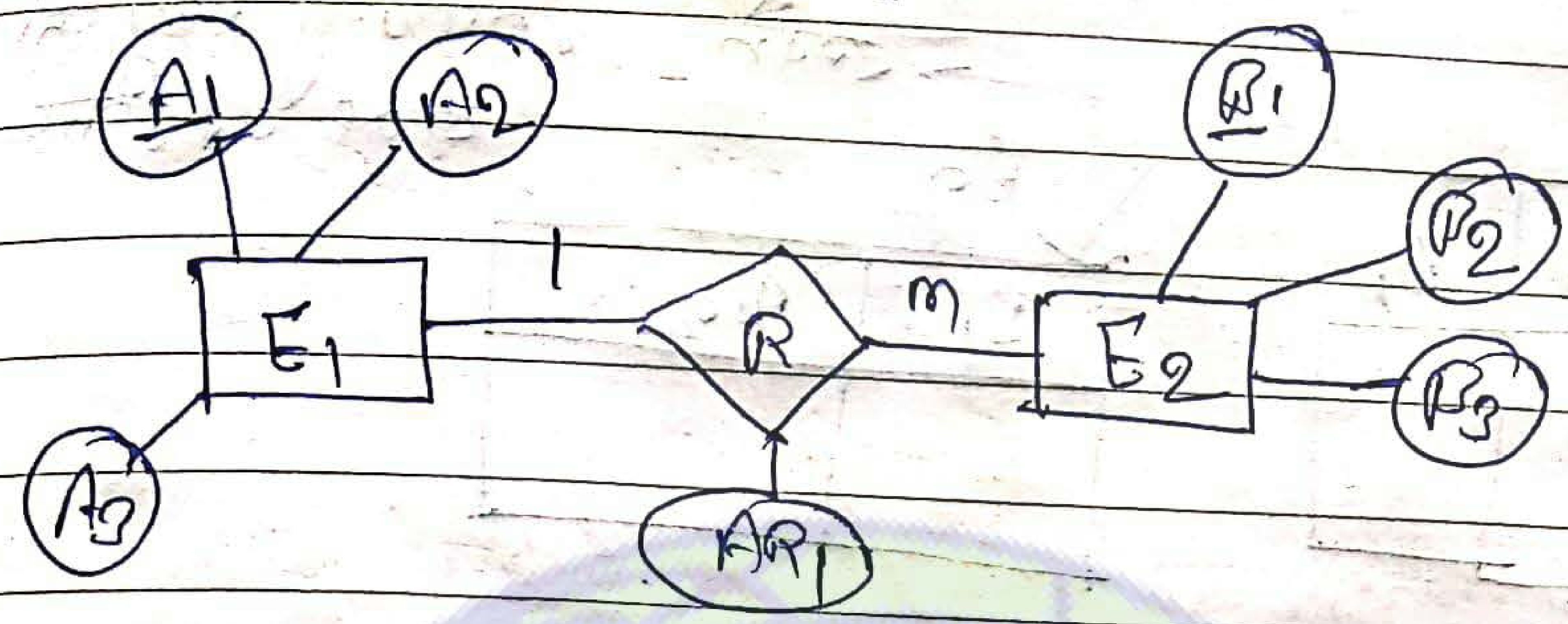
Note: weak entity can be changed to multivalue attribute and also vice-versa.

student (Rollno , name(fname, mname, lname), phoneno (multivalue), email (multi), dob, age (derives))

single value





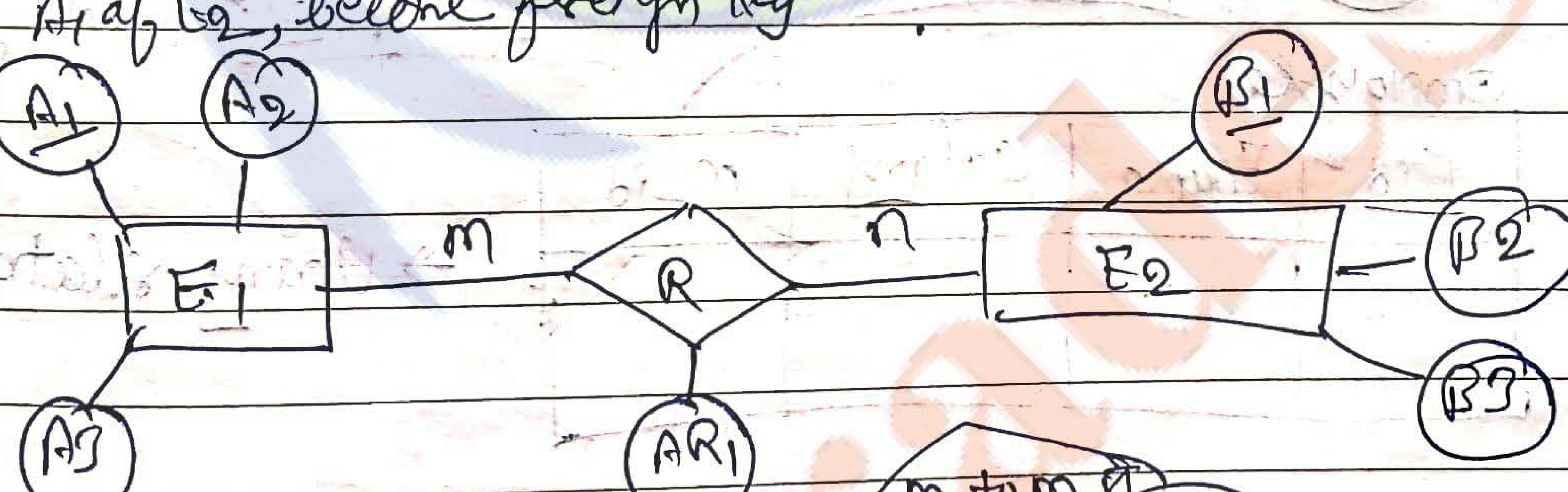


one to many rel^n

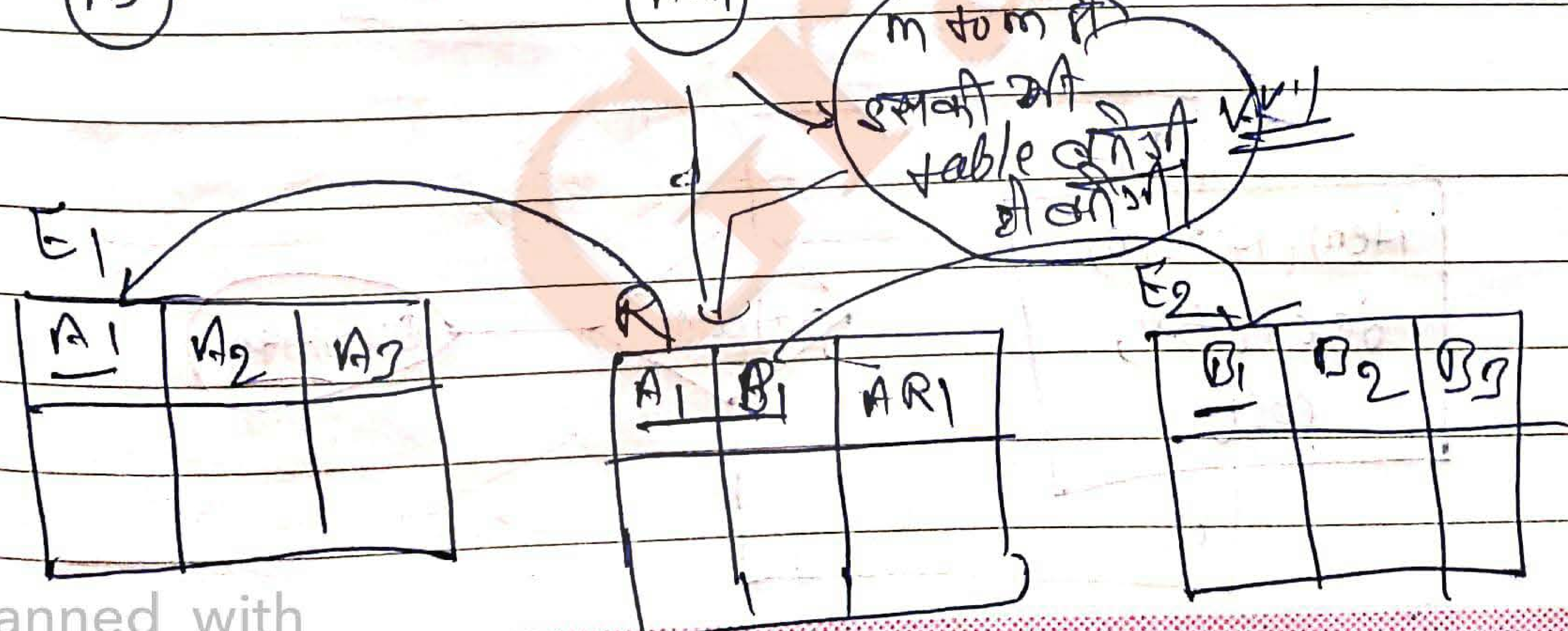
It should be primary key or Unique key for (It may contain null value)

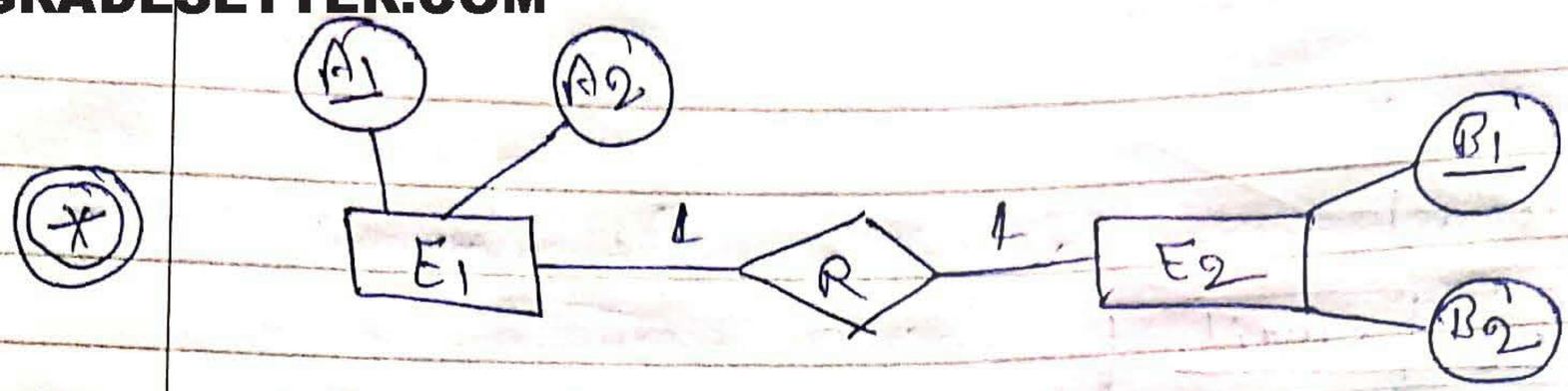
A1 of E2, become foreign key

many वाले side पे 1 का primary key + relation का attribute//



many to many





one to one

→ इसमें Primary key को किसी side से लेते हैं।

E1

A1	A2	D1

E2

D1	D2

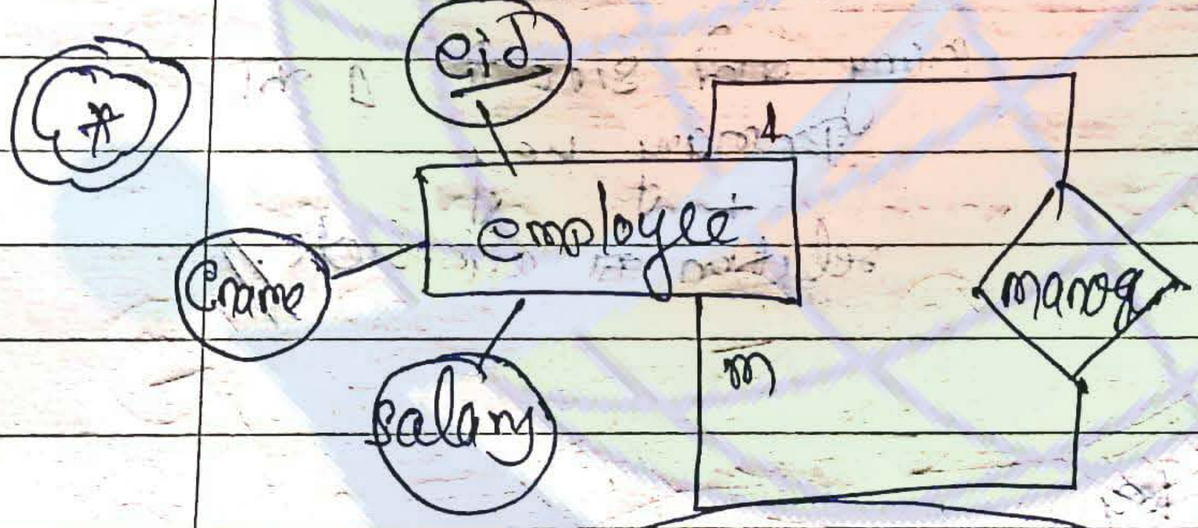
or

E1

A1	A2

E2

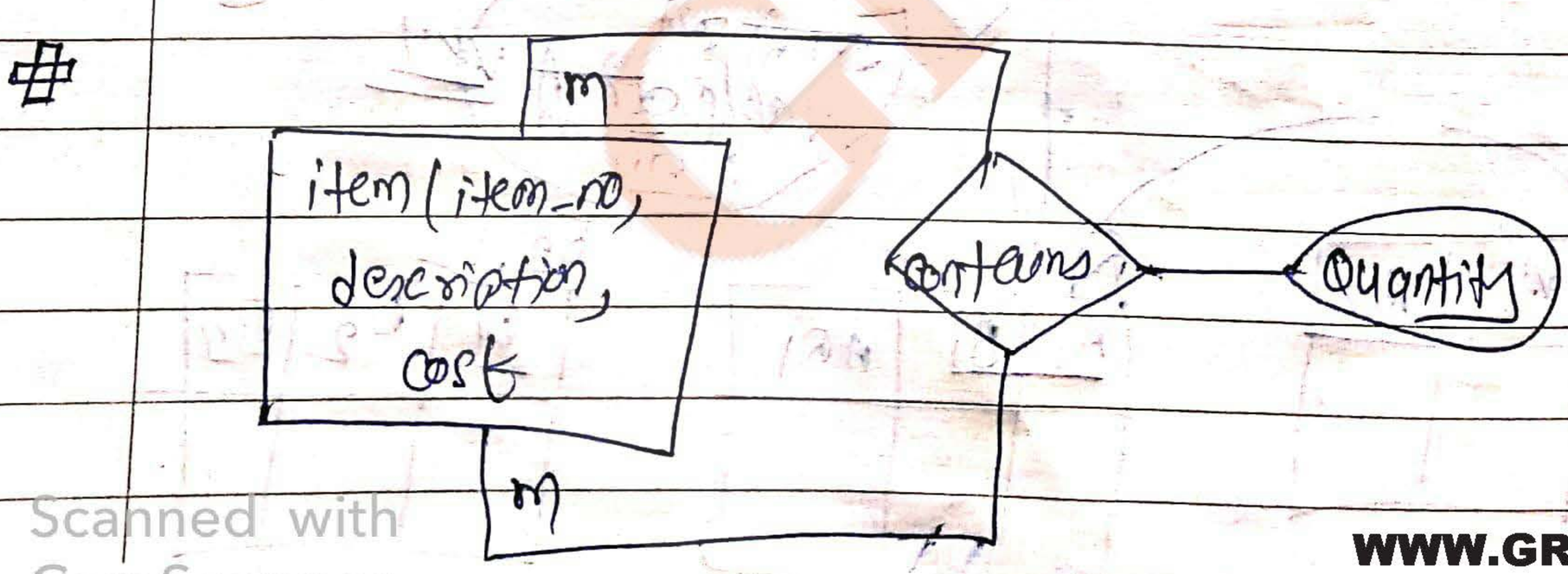
A1	B1	B2

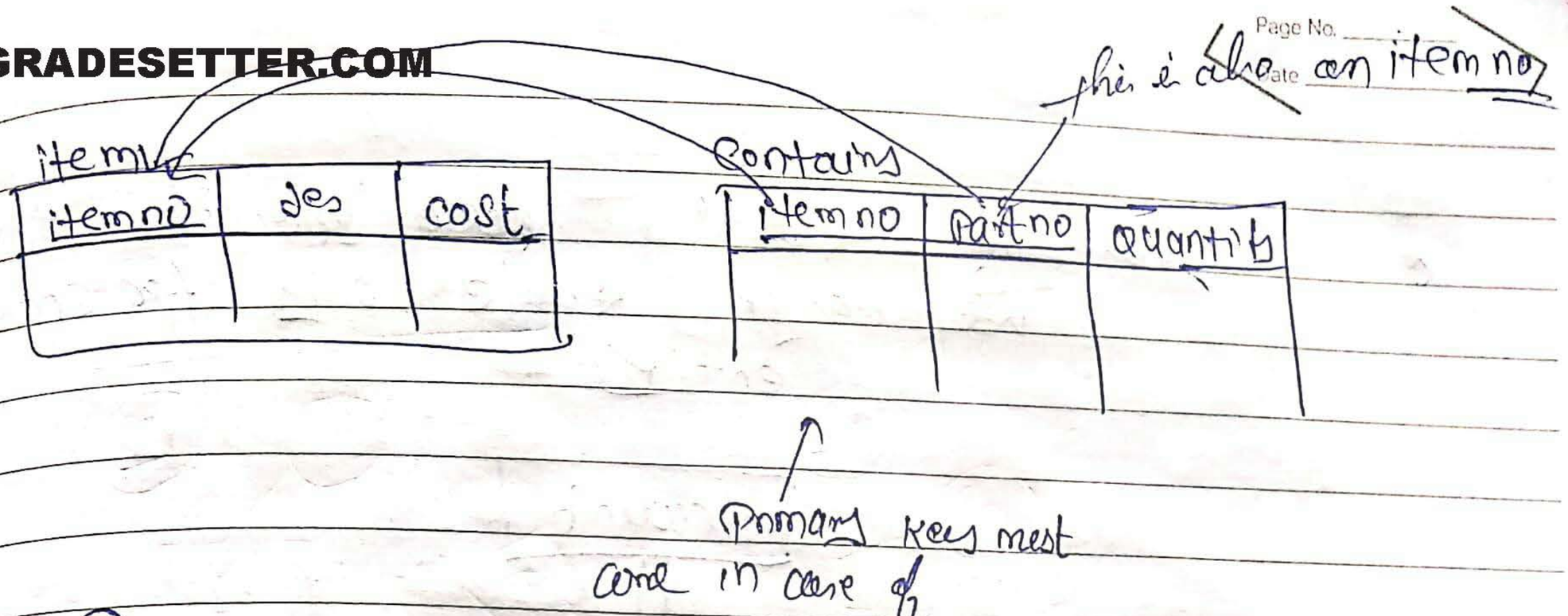


employee

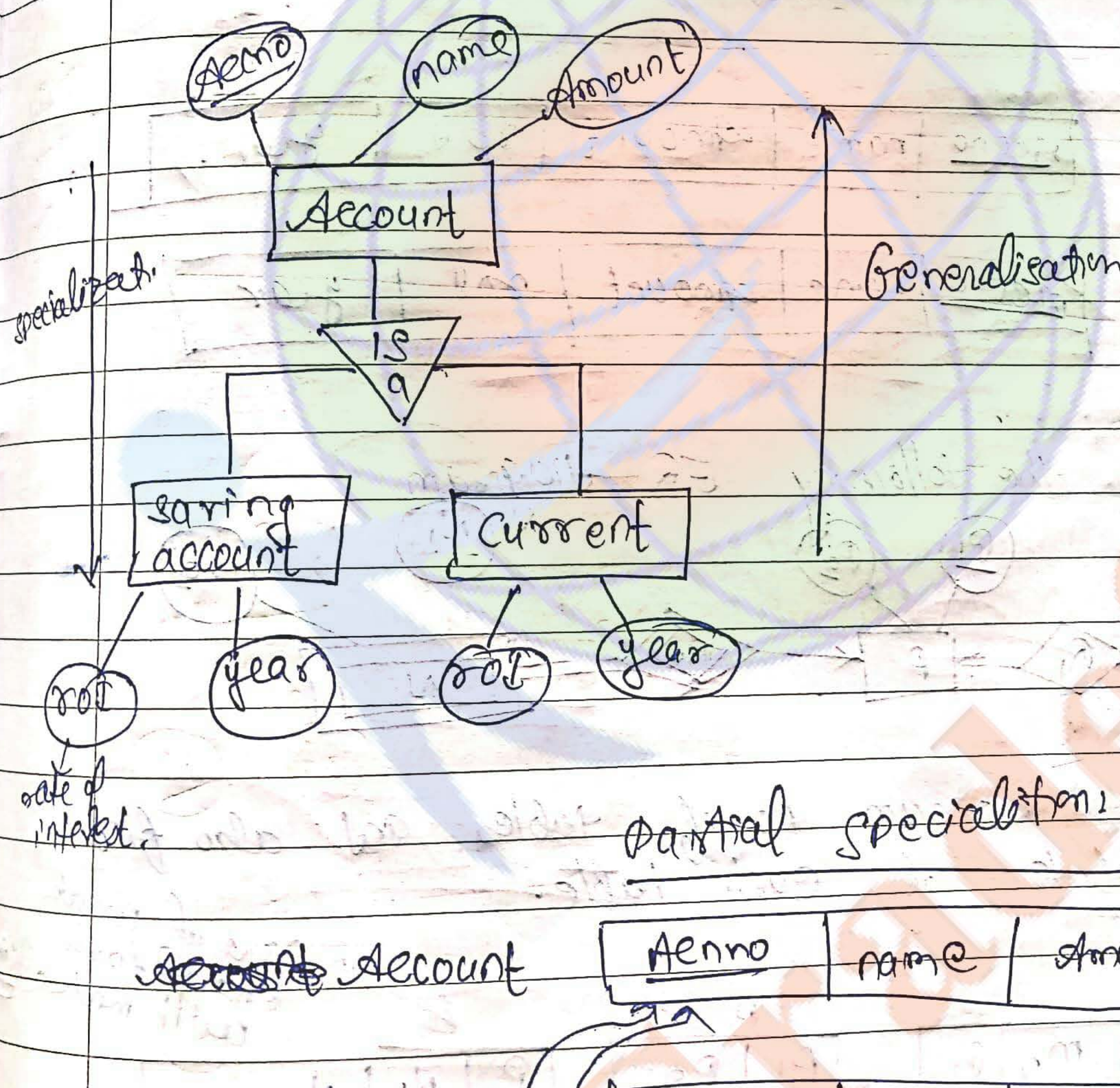
Eid	Ename	salary	m_id

→ Unary relation





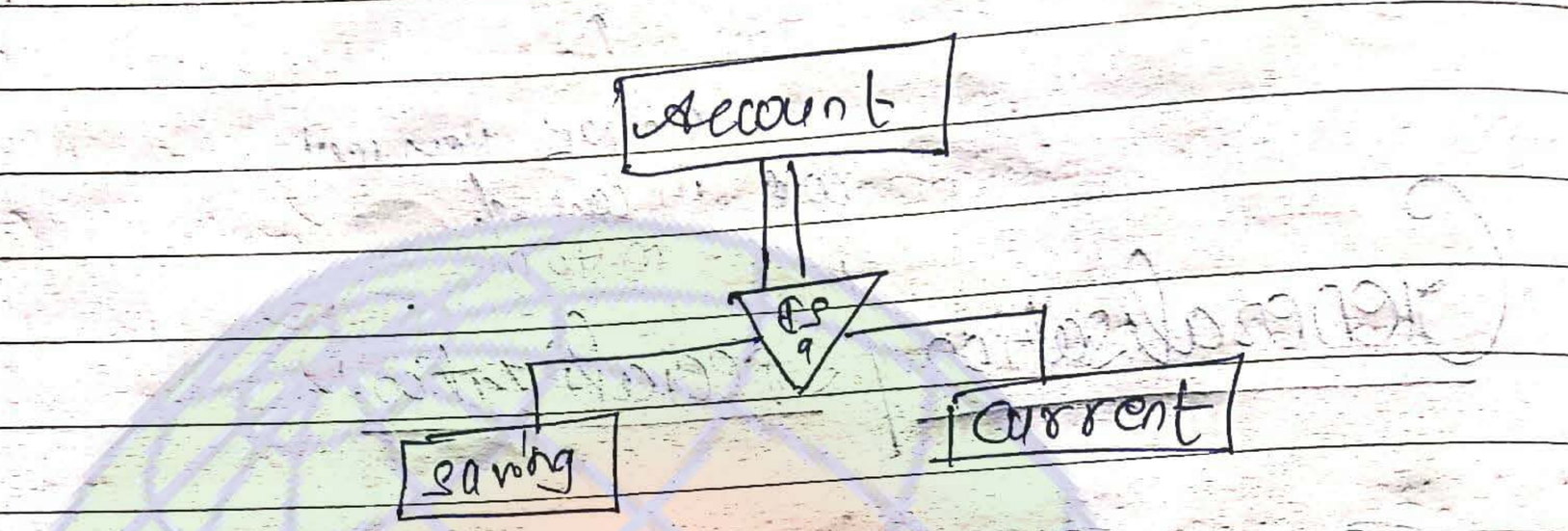
Generalisation / Specialization :-



Partial specialization:

Page No.
Date

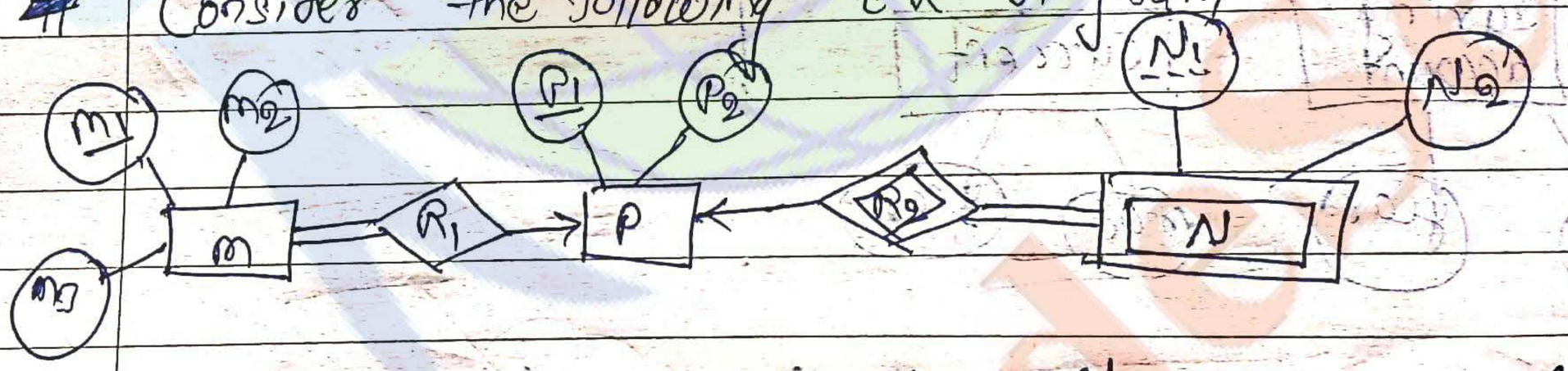
* In case of total specialism we design table only for specialise entity



saving | Accno | name | Amount | ROI | year

current | Accno | name | Amount | ROI | year

* Consider the following ER diagram



a) The minimum no. of tables and also find attribute of each table.

in case of weak entity both will make key

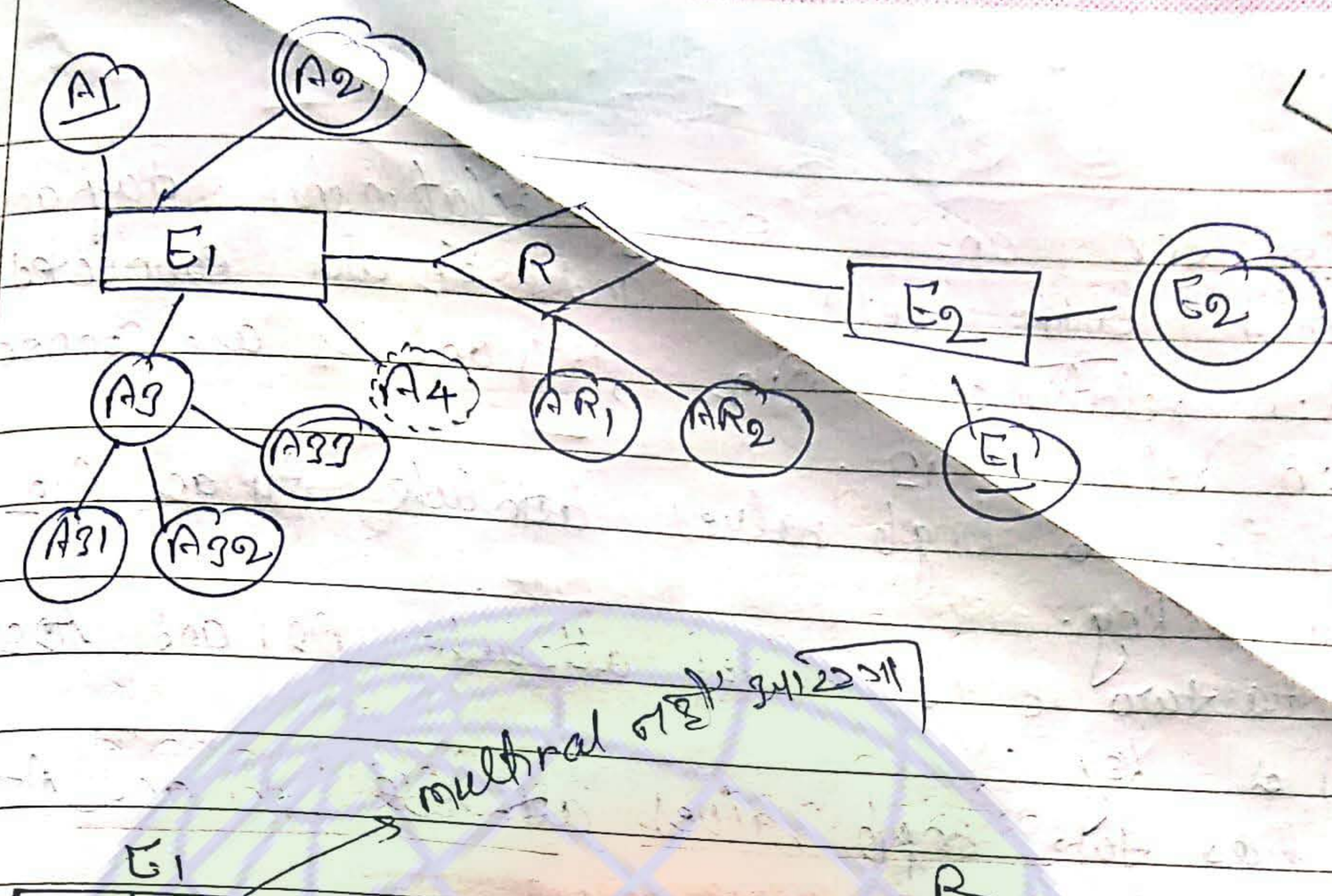
m

m1	m2	m3	P1

P

P1	P2

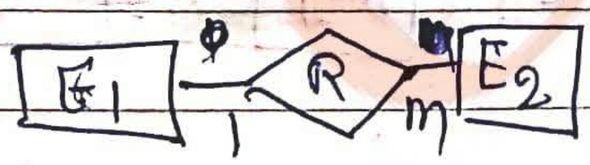
P1	N1	N2

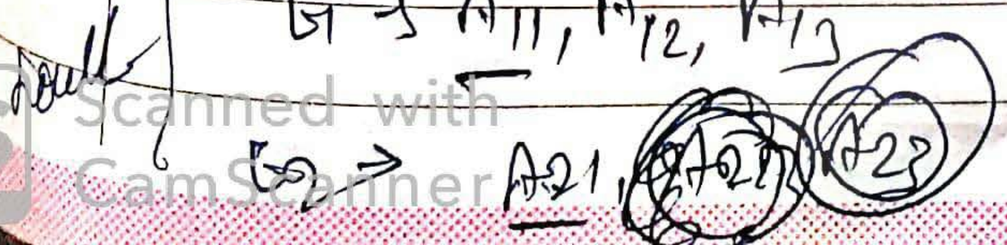


Q) Consider the following entity relationship diagram, the attribute of E1 is A_{11}, A_{12} and A_{13} where A_{11} is primary key.

the attribute of E2 are A_{21}, A_{22}, A_{23} where A_{21} is primary key and A_{23} is multivalues attribute. relation R does not have any attribute. A relational database containing minimum no. of tables, with each table satisfy 1st normal form

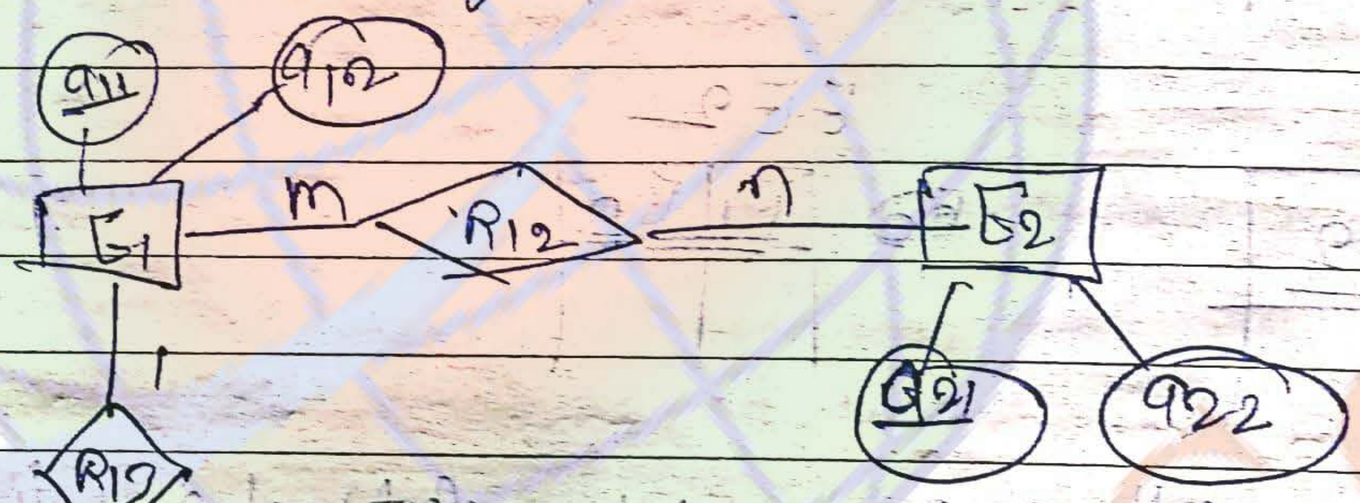
The min. no. of table for database is.





Q) Consider an entity relationship diagram in which entity sets E_1 and E_2 are connected by m:n relationship R_{12}, E_1 and E_3 are connected by 1:n relation R_{13}.
E_1 has two single values attribute A_{11} and A_{12} where A_{11} is key:
E_2 has two single values attribute A_{21} and A_{22} where A_{21} is key
E_3 has two single values attribute, A_{31} and A_{32} where A_{31} is key.
The relationship does not have any attribute, write the min. no. of table, generated by the rel^n if all rel^n is in 3NF.

Sol^n



q_{11}	q_{31}	q_{32}
1	a	x
1	b	y
2	c	x
3	d	x

E_1 | q_{11} | q_{12} | ✓

E_2 | q_{21} | q_{22} | ✓

R_{12} | q_{11} | q_{21} | ✓

E_3 | A_{31} | q_{32} | q_{11} | ✓

Page No.
Date

.) A ER model of a database consist in entity and relationship. Do which of the bellow statement is incorrect

1) An attribute of an entity can have more than one value

2) an attribute of a entity can be composite

3) In a row of relational table, an attribute can have more than one value.

4) In a row of relational table, an attribut can have exactly one value or a null value.

SQL - Structured Query Language

i) It is non-procedural query language.

ii) It is case insensitive

Three types of statements:-

1) DDL: (Operation on structure of table)
- Create, alter, drop, truncate.
- They all are autocommitted.

2) DML:
- select, delete, update, insert
- They are not autocommited.

3) DCL:
- commit, rollback, grant, revoke

① DL command:

① Create table:

Create table student (Rollno varchar(20), name char(10), phon no number(10));

② Primary key:

create table student (Rollno varchar(20) Primary key, name char(10), phono number(10));

(3) create table (two or more attribute Primary key)

create table student (enrollno number(10), Rollno varchar(20), name char(10), Primary key (enrollno, rollno));

Page No.
Date

(4) Alter:

Alter command is used to update the structure of existing table.

⇒ Add column.

Alter table student add marks number(3);

⇒ ~~Delete~~ Drop

Alter table student drop phonno;

⇒ To change the size of column.

Alter table student Alter phonno number(11);

↑ It change from 10 to 11.

Note: Restrictions on alter

1) Can not change the name of table
2) " " " " " column
3) " " decrease " size of datatype, if data exist.

5) Rename:

⇒ Rename student to employee;

6) Drop:

Drop table student;

(Complete table is drop with structure.)

7) Truncate:

Truncate table student;

(delete the data ~~but~~ permantly, but ~~not~~ structe is not deleted)

- Data permantly delcates.

Page No.
Date

DML Command:

① Insert

Insert into student values ('0812', 'ABC', 9891, 98);

⇒ Insert on specific columns -

⇒ Insert into student (col1, col2) values (val1, val2);

② Delete:

Delete * from student;

→ Student table का सारा data delete हो जाएगा पर rollback से वापस आ या जाएगा।

→ Structe not deleates.

→ Delete phono from student;

⇒ Delete phono, marks from student;

⇒ Delete * from student where rollno = 0812;

⇒ Delete specific column

Delete phoneno from student where rollno = 0812;

3) Update:

Update student set Phone no = 9891;

⇒ specific row

Update student set Phone no = 9891 where rollno = 0812;

Page No. _____
Date _____

4) Select :-

to retrive data from table

Select * from student;

⇒ Select name from student;

⇒ Select * from student where dob = '27105';

⇒ Select name from student where rollno = 0812;

⇒ select name, marks ——— 11

emp (eno, ename, job, salary, dno)

write a query to print -

a) name of all employes who work in a dno = 10';

-Select name from emp where dno = 10';

b) Retrive name of emp work dno = 10, and salary > 5000

- Select name from emp where dno = 10 and salary > 5000;

c) either dno = 10 or dno = 20

" " " 10 " dno = 10 or dno = 20';

or

" " " " dno in (10, 20, 30, 40);

d) write a query to find name of emp where salary > 1000 and ≤ 2000

" " " " salary > 1000 and salary ≤ 2000;

or

>=

or

<=

or

salary between 1000 and 2000;

Page No.
Date

Note: Between always include boundary values.

So, between will be used in case of equals to (=).

① select ename from emp where ename between 'A' and 'D'

A ✓

A---✓

B---✓

C---✓

D ✓

Note: सिर्फ D आएगा।

③ ① Distinct

② select distinct ename from emp;

③ select distinct ename, job from emp;

Ename	Job
Ram	CS
Ram	CE
Raj	CS

All will print. Because ename and job are combination are distinct.

④ select ename, distinct job from emp;

↳ show error

(Note distinct will always apply either on whole or not)

a. empl eno, name, job, salary, dno, commision, email)
write a query to find name of employe with null commisson.

- select ename from emp where commision is null
or
is not null

Note:- we can not use directly these value with null
<=
>=
!

b) write a query to find name of emp. where 1st character is C.

select ename from emp where ename like 'C%';

c) atleast two a

" like '%a%a';

c) three length character long

" like '___';

d) atleast 5 character

" like '____%';

e) write a query to find name of emp, where email id contain Interscore(_);

like '%/_%' ;

चूकी (_) as a data चाहीये /

/% → % as a data चाहीये

// → forward slash as a data चाहीये

4) select name of emp in ascending order.

select ename from emp order by ename asc;

Note: for descending desc;

5) order by salary ; max at top and min at bottom.

" salary desc;

★ Group related function or Aggregate function

1) sum()
2) max()
3) min()
4) Avg()
5) Count
6) First()
7) last()

salary
1000
2000
NULL
3000
4000
5000
4000

a) select sum(salary) from emp;

→ Sum of all is shown

Sum salary
240000

Note

select sum(salary) as salary_sum from emp;

as → for change of heading.

salary_sum

b) max(salary) = 5000

c) min(salary) = 1000

Note: Null value will ignore in case of min.

d) Count :→ No of salary

Count(salary) ⇒ 7

Note: Null will ignore.

e) Count(distinct salary) ⇒ 5

f) Count (*) ⇒ 8

↑ मे पूरे table की content को देखेगा, अगर पूरा record null होगा तो count नहीं करेगा।

✓	✓			Null

Page No. ____
Date ____

g) Avg (salary) = sum(salary) / count (salary)

= 24000 / 7

h) First (salary) =

i)

o) Print name of emp, whose salary is max in dno=10;

Select ename from (emp where ~~max(salary) and dno=10~~; (group by dno) (having max(salary)) (where dno=10);

( ) bracket जहाँ पर लगी है उसको पहले समझना है

aggregate funct. की value having आता है

dno के after change कर सकते है

पहले execute है where है execute होगा

Select ename from emp where dno=10 group by dno having max(salary);

Note

group by → having → aggregate function

*a) select ename, sum(salary) from emp;
↳ show error.

b) select dname, sum(salary) from emp group by dname;

dname	sum(salar)
CS	—
ME	—

(grouping के स्थान पर dname, तो दी ज्या सकता है

Inner Query Concept

Q) Find the name of employee drawing highest salary from the company.

select ename from employee where
salary = (select max(salary) from emp);
↓
This is concept of inner query.

Q) " " "
second highest salary from the company.

select ename from employee where
salary = (select max(salary) from
emp where
salary < (select max(salary)
from emp));

Q) Find the name of employee whose salary is greater than the salary of Raj.

select ename from employee where
salary >All (select salary from
employee where ename='Raj')

>All → सभी Raj से बड़ा

>Any → किसी भी Raj से बड़ा

In or = Any (किसी भी Raj के बराबर)

= All (Not possible)

Note

>All (1000, 2000, 3000) ✓

>Any (1000, 2000, 3000) ✓

= Any or In ✓

<> All ⇔ not in

↓ ↓

not all equal

= Any ⟺ in

& Multiple table

emp (ename, eno, Dno)
dept (dno, dname)

Select ename from emp where dno = (select dno from dept where dname = 'research')

↓
find the name of emp who works in research dept.

☆ Join :-

① Natural join (equi join) (inner join)

② Outer join
- ① left outer join
- ② Right outer join
- ③ full outer join

③ Cross join (cartesian product)
④ Theta join
⑤ self join.

Page No.
Date

* emp (ename, eno, dno)
dept (dno, dname)

emp

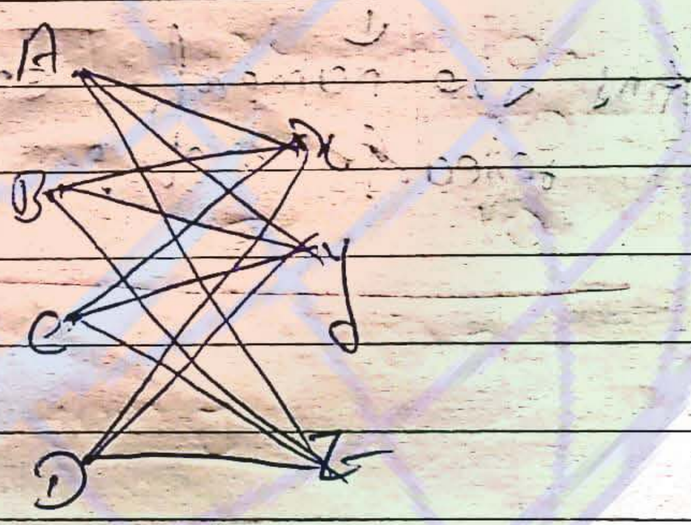
eno	ename	dno
1	A	10
2	B	20
3	C	10
4	D	30

dept

dno	dname
10	x
20	y
30	z

(1) Cartesian Product

select ename, dname from emp, dept;



⇒ 12

emp, dept

m x n

5 X 0 = ϕ

where

Output = m x n

(*)

(2) Natural/ equi join

select ename, dname from emp, dept
where

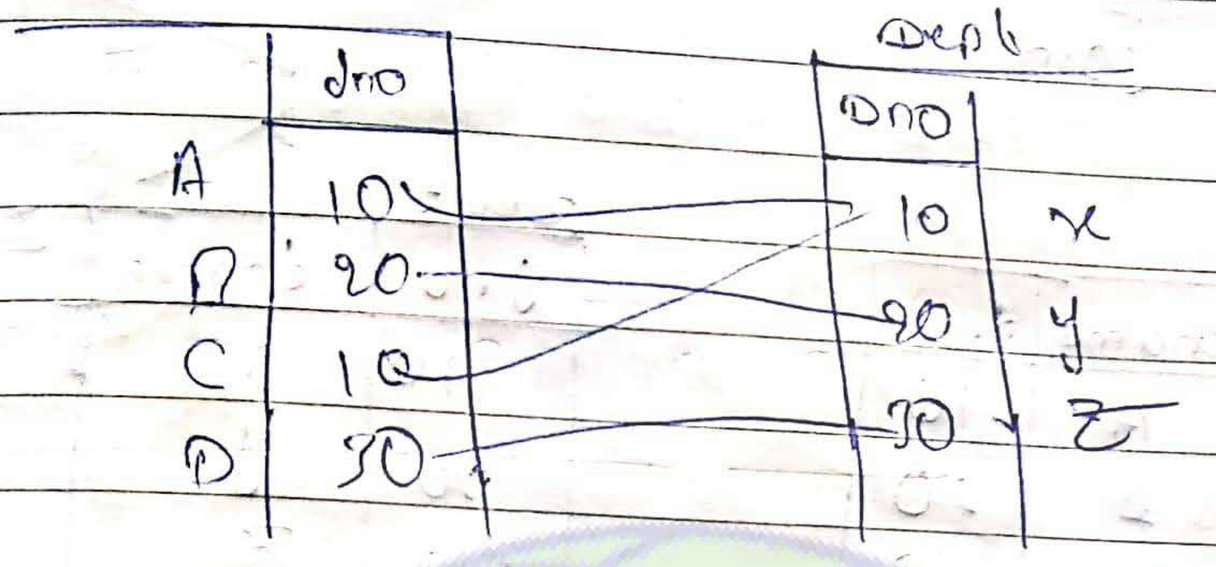
emp.dno = dept.dno;

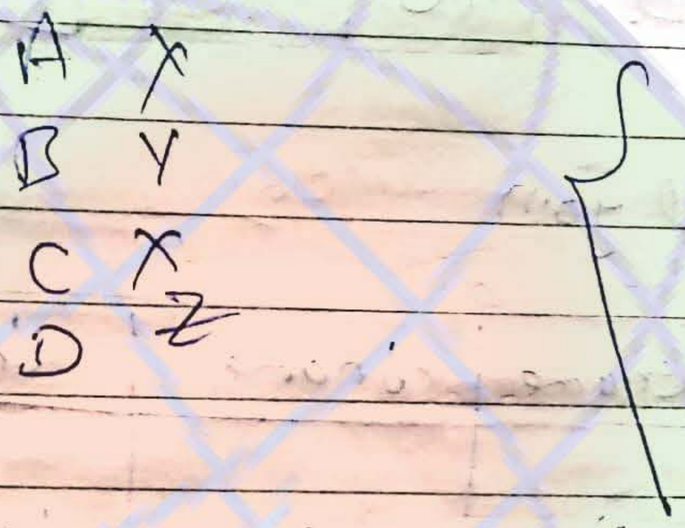
दोनों

का

equal होना चाहिए

A	X
B	Y
C	X
D	Z





Q.

Select ename, dname, from emp natural join dept;

Note

If there does not exist any common attribute, then it will perform cartesian product.

(*)

emp

A	B	C
1	2	1
2	2	3
3	4	3
4	2	4

dept

B	C	D
2	1	3
2	2	4
2	3	4
2	3	5

Q.

(3) Outer join:-

emp

eno	ename	dno
1	A	10
2	B	20
3	C	10
4	D	40

Dept

dno	dname
10	x
20	y
70	z

Natural join

ename	dname	dno
A	x	10
B	y	20
C	z	10

left outer join

ename	dname	dno
A	x	10
B	y	20
C	z	10
D		

select ename, dname, * from emp, dept
where
emp.dno = dept.dno (+) ;
↳ left outer join

Right

Full

Right outer join :-

ename	dname	dno
A	x	10
B	y	20
C	x	10
	z	30

where

emp.dno (+) = dept.dno ;

↳ Right outer join

Note

(+) outer की दिशा

Full outer join :-

where

emp.dno(+) = dept.dno(+) ;

ename	dname	dno
A	x	10
B	y	20
C	x	10
D		
	z	30

(4) ☆ Self join:

emp | e-id, ename, m-id

e1
e2

we are creating alias of the table

select e1.ename, e2.ename from emp as e1, emp as e2 where e1.e-id = e2.m-id;

Change of heading:

select e1.ename as ename, e2.ename as mname from — —, — —;

(5) Union, Intersection, except (minus)

(a) Union:

select ename from credit union select ename from debit;

⇒ select * from credit union select * from debit;

- Ascending order
- eliminate duplicate pair

ename

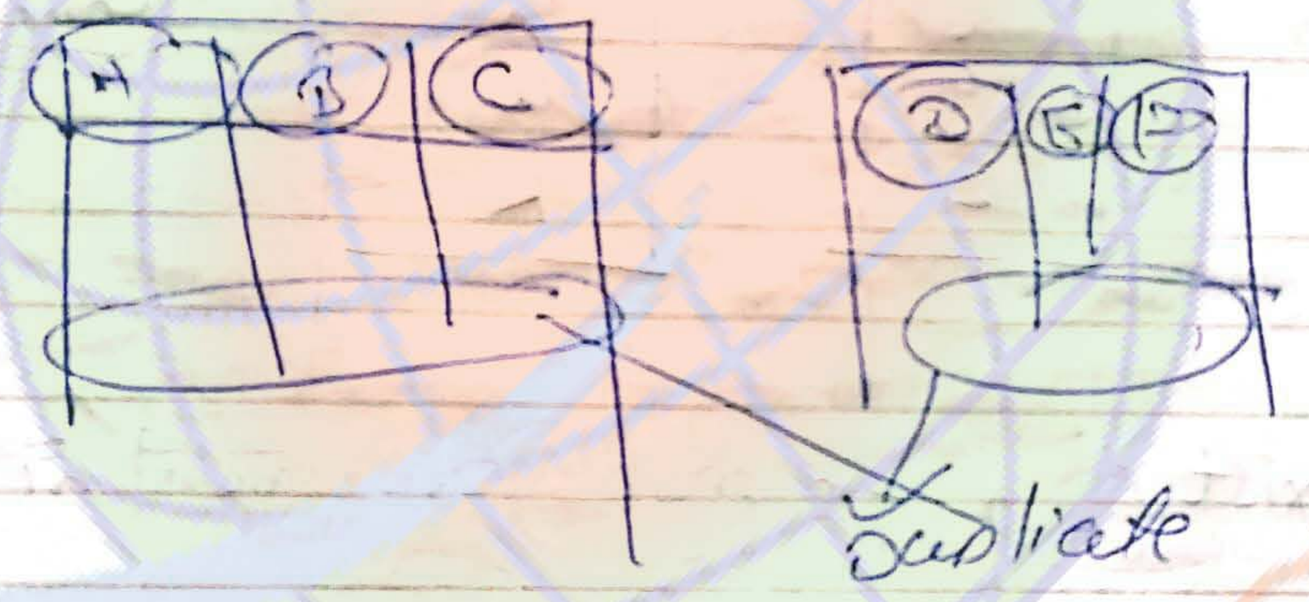
R1

Page No.
Date

Points

Output:- Default
a) always Print with the heading of 1st table
b) Result always in ascending order
c) By default eliminate duplicate values

#
① No of column in both table should be equal
② There exist one to one corresponding b/w datatype of both the table.



Note

If we want to print duplicate values
use 'All'

→ " Union All - "

b) Intersection (जो common होगा, Print होगा)
↳ as follow all rules of Union.

Select ~~Intersect~~ Name from Credit Select Name from debit.

(c) except :-

A - B

select Cname from credit except select dname from debit;

eg.

Cname
x
y
z

dname
x
w
k

Cname - dname =

y
z

exist(--) → यदि exist करेगा तो query file देगा ||
not exist(---)

* Constraints & Type of Constraints :-

(1) Primary key constraint :-
(2) Unique constraint
(3) Foreign key constraint
(4) check constraint

(1) Primary
value should be unique and does not contain null value.

① ~~Cdb~~
- There exist only one Primary key in table.

② Unique key :-
contain unique value but may contain null value

- we can design more than one key as Unique.

Note:
Unique not null → It will work like Primary key ~ but not a Primary key.

Create table student (Rollno varchar (10), name char(10), check (Rollno like '%cs%'));

③ Foreign key Constraint:

desired table (emp(eno, ename, dno) f.k)

(dept (dno, dname)

base table

Foreign key - referencial integrity

emp

eno	ename	dno
1	A	10
2	B	10
3	C	
4	A	30
5	B	70

dept

dno	dname
10	x
20	y
30	z
40	w