

DBMS

A. AM

Date
1

- E-R model
- Relational model
- Functional Dependency (F.D), Normalization
- SQL
- Relational Algebra, Relational Calculus
- Transaction Management, Concurrency Control
- File organization, Index

2
:- Abstract in nature (Raw fact)

Information :- Data with added meaning.

Record :- Collection of logically related data

ex:-

< 501 Rqj 530 >

Database :- Collection of records.
similar
(OR)

Collections of logically related data.

Management :- Through set of programs

DBMS :- Collections of logically related data and set of programs
to access those data.

Applications :-

- Banking
- Telecommunications
- Reservation Systems
- Sales
- Scientific applications

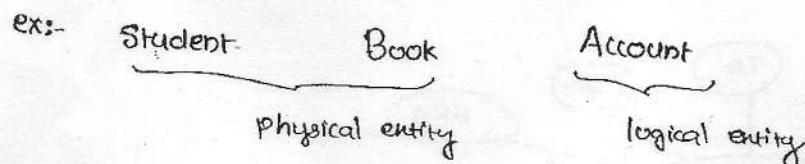
Goal of DBMS :- Effective storage and retrieval of Data from
DBMS.

Tree	Hierarchical DB	HDBMS	} outdated
Graph	Network DB	NDBMS	
Table	Relational DB	RDBMS ✓	
Objects	Object-oriented DB	OODBMS	
Object/Table	Object-relational DB	ORDBMS	

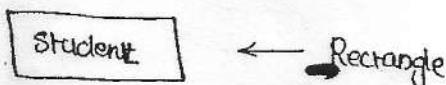
Conceptual Database Design using C. Entity-Relationship (ER) Model

Components of E-R Model

1) Entity : An object in the real world.
 "Noun" entities



2) Entity set :- Collection of similar entities.

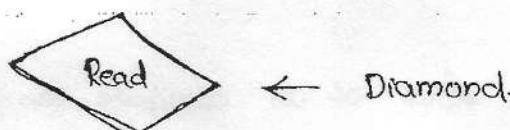


3) Relationship :- Association among the entities
 "Verbs" operations

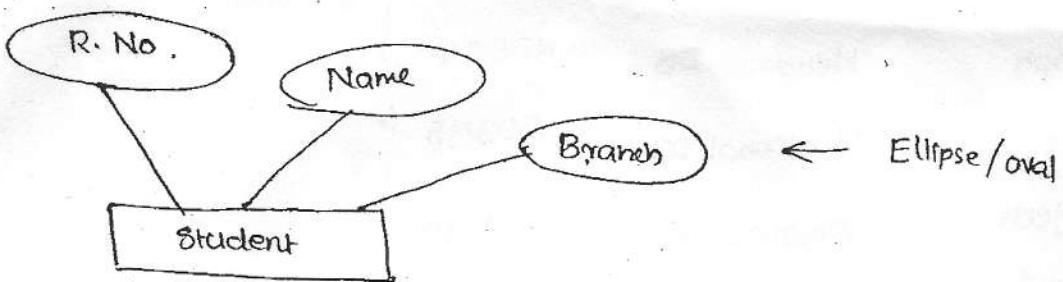
ex:-

- Reading
- Buying

4) Relationship set :- Collection of similar relationships.

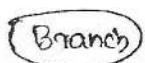


5) Attributes :- Which describes an entity.

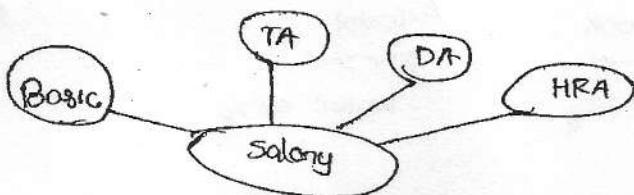


Classification of Attributes

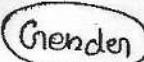
1) Simple Attribute :- Which can not be divided further.



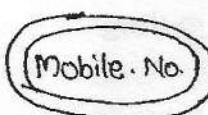
2) Composite attribute :- which can be divided further.



3) Single Valued attribute :- which takes one value per an entity.



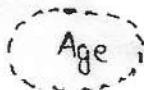
4) Multivalued attribute :- which takes more than one value per an entity.



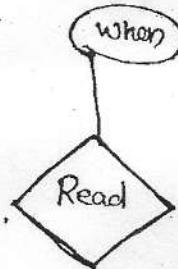
5) Stored attribute :- which does not require any updation.



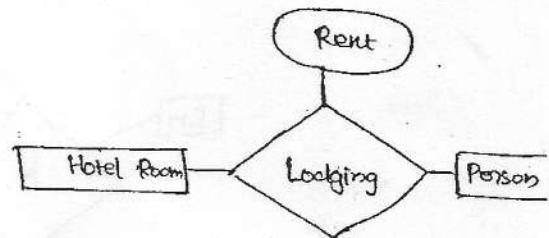
6) Derived attribute :- The Value of an attribute can be derived from other attributes.



7) Descriptive attribute :- which gives information about the relationship set.

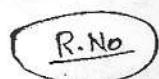


ex:-



6-2003

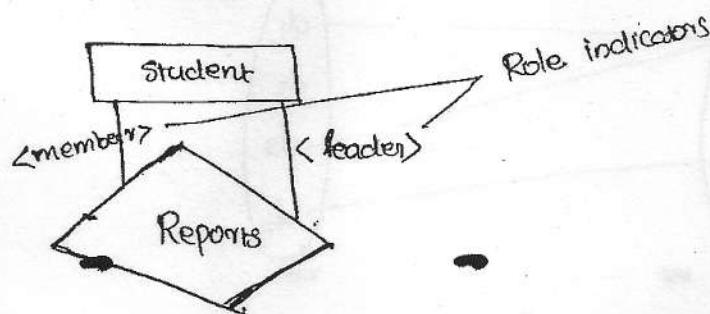
8) Key attribute :- which uniquely identifies an entity in the entity set.



6

Degree of relationship set :- Specifies the no. of entity sets participates in a relationship set.

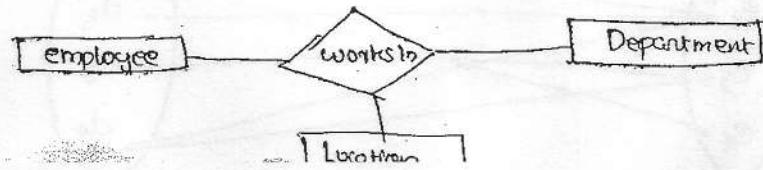
1) Unary :- Relationship among two entities of the same entity set (one entity). (Recursive relationship set)



2) Binary Relationship set :- The relationship among two entity sets.

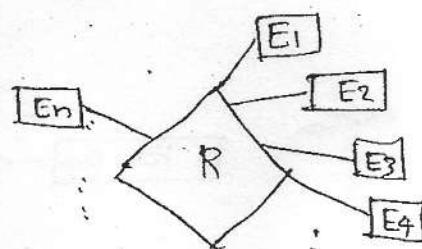


3) Ternary relationship :- Relationship among three entity sets.



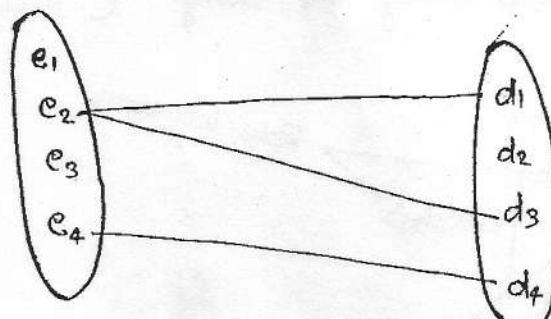
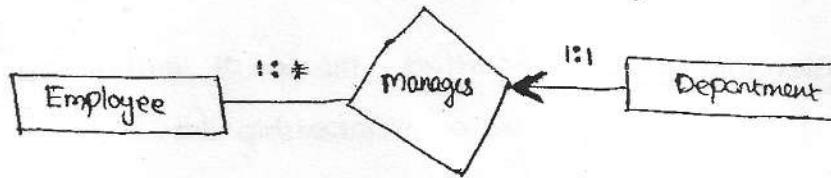
4) n-any :- A Relationship Among n-entity sets

6



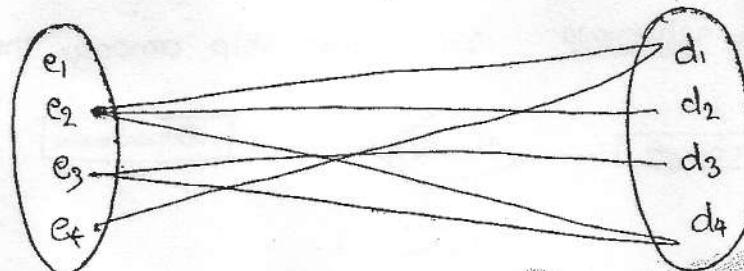
| Key Constraint :- An entity is acting as a key to another entity through the relationship set it is denoted in E-R model using an Arrow

" Each department is managed by atmost one employee "



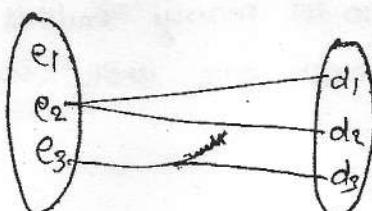
8) Participation Constraint :- If every entity in the entity set participates in a relationship set it is called total participation denoted by double line (thick line). otherwise it is called partial participation. (Thin line or single line)

" Each department is managed by atleast one employee "



"Each Dept is managed by exactly one employee"

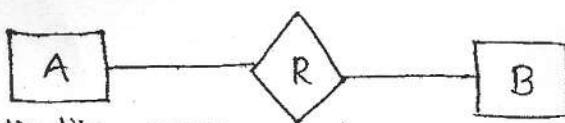
7



* Mapping Cardinality (Cardinality Ratios)

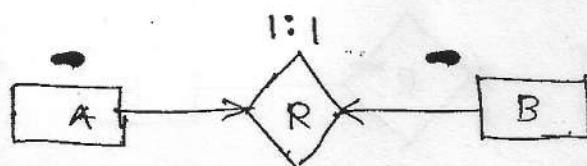
It Express the no of entities to which another entity can be associated via a relationship set.

* (only on binary relationships)

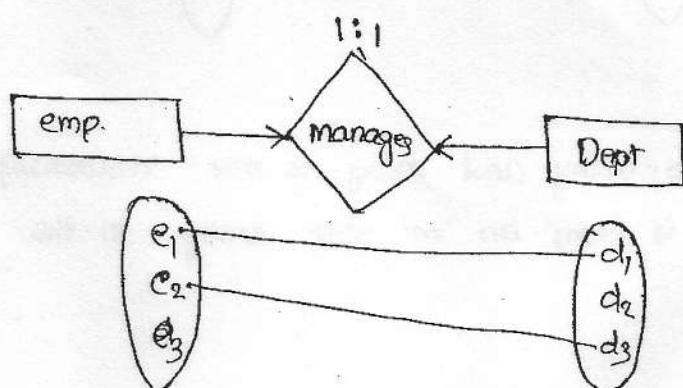


Note:- The cardinality ratios can be expressed on a binary relationship set only

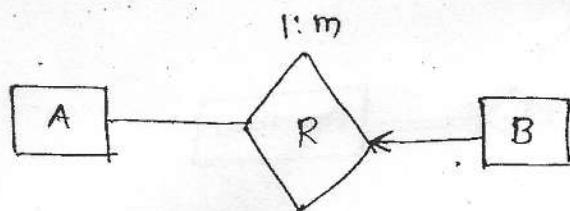
• One to one (1:1)



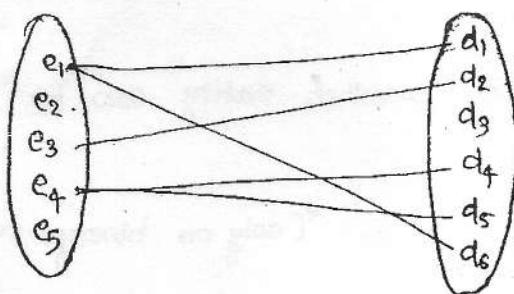
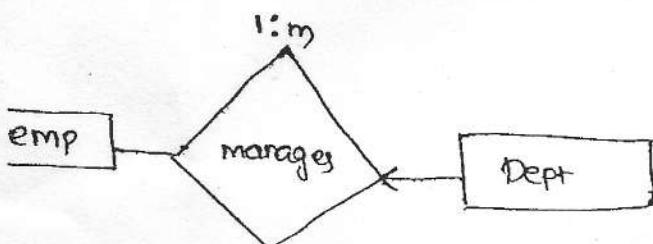
An entity in A is associated with atmost one entity in B and an entity in B is associated with atmost one entity in A.



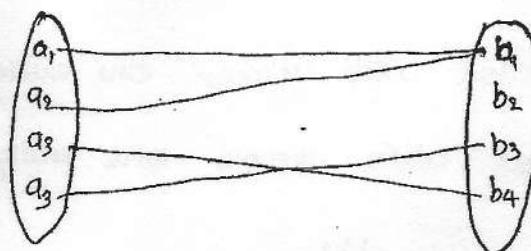
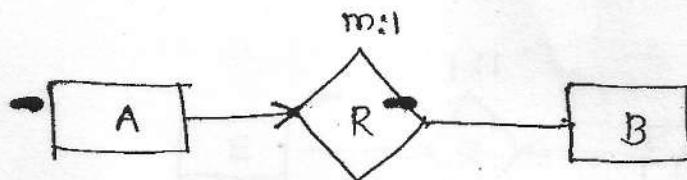
③ one-to-many (1:m)



An entity in A is associated with zero or many entities in B and an entity in B is associated with almost one entity in A.



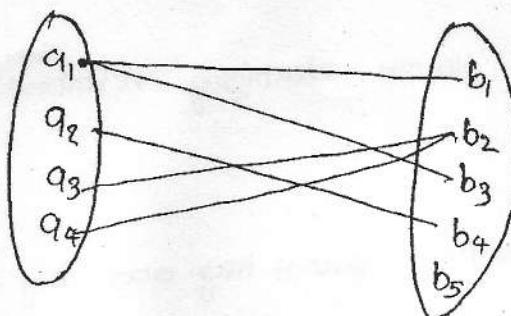
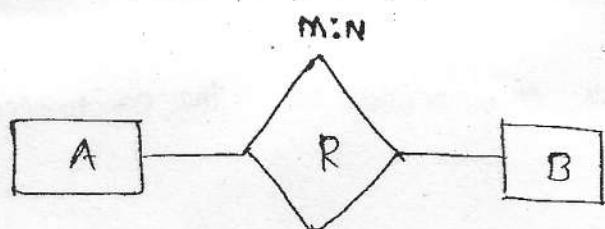
④ many-to-one (M:1)



Note: In one-to-many and many-to-one relationship set the key constraint is from an 'm' side entity to the relationship set.

Many-to-many (M:N)

9



Strong Entity Set :-

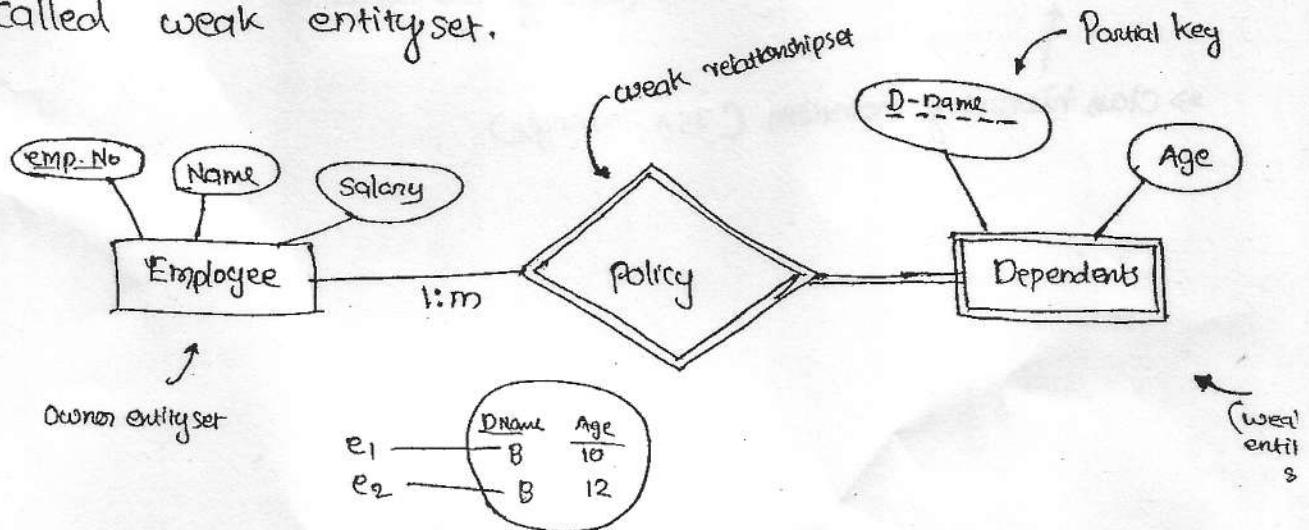
An entity set which has a key is called strong entity set

Ex:-



Weak Entity Set :-

An entity set which does not have a key attribute is called weak entity set.



→ Partial key are discriminating attribute.

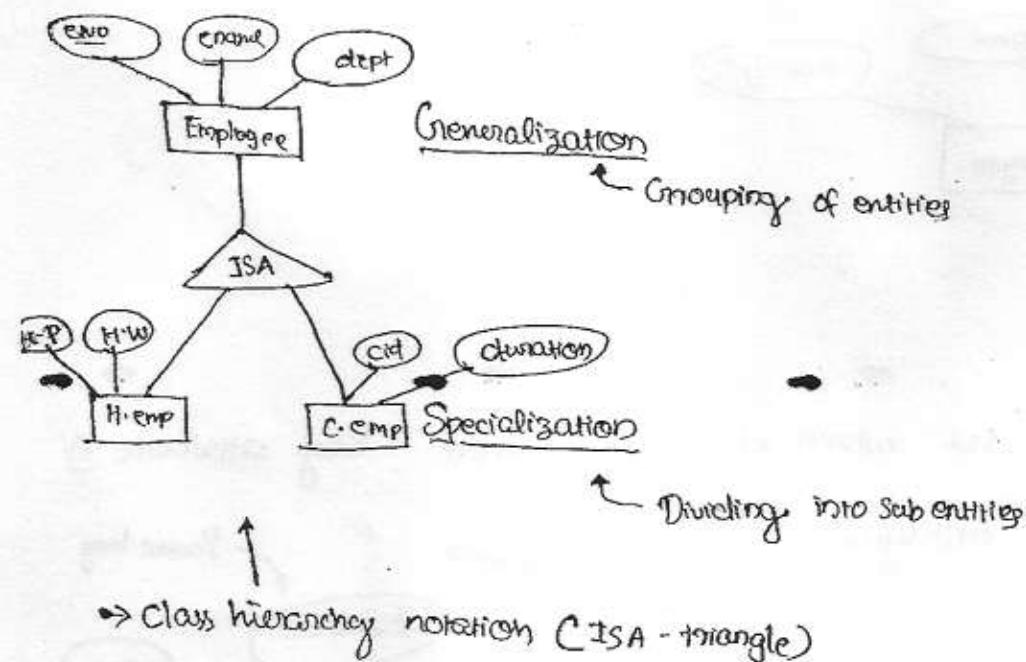
Weak relationship set (09) Identifying relationship set

10

→ Owner entity set (09) Identifying owner

- The owner entity set to the weak relationship set the cardinality ratio is one to many.
- ~~Participation / cardinality rule~~
The participation of weak entity set to the identifying relationship set is always total.
- The weak entity set is identified using partial key and key of the owner entity set.

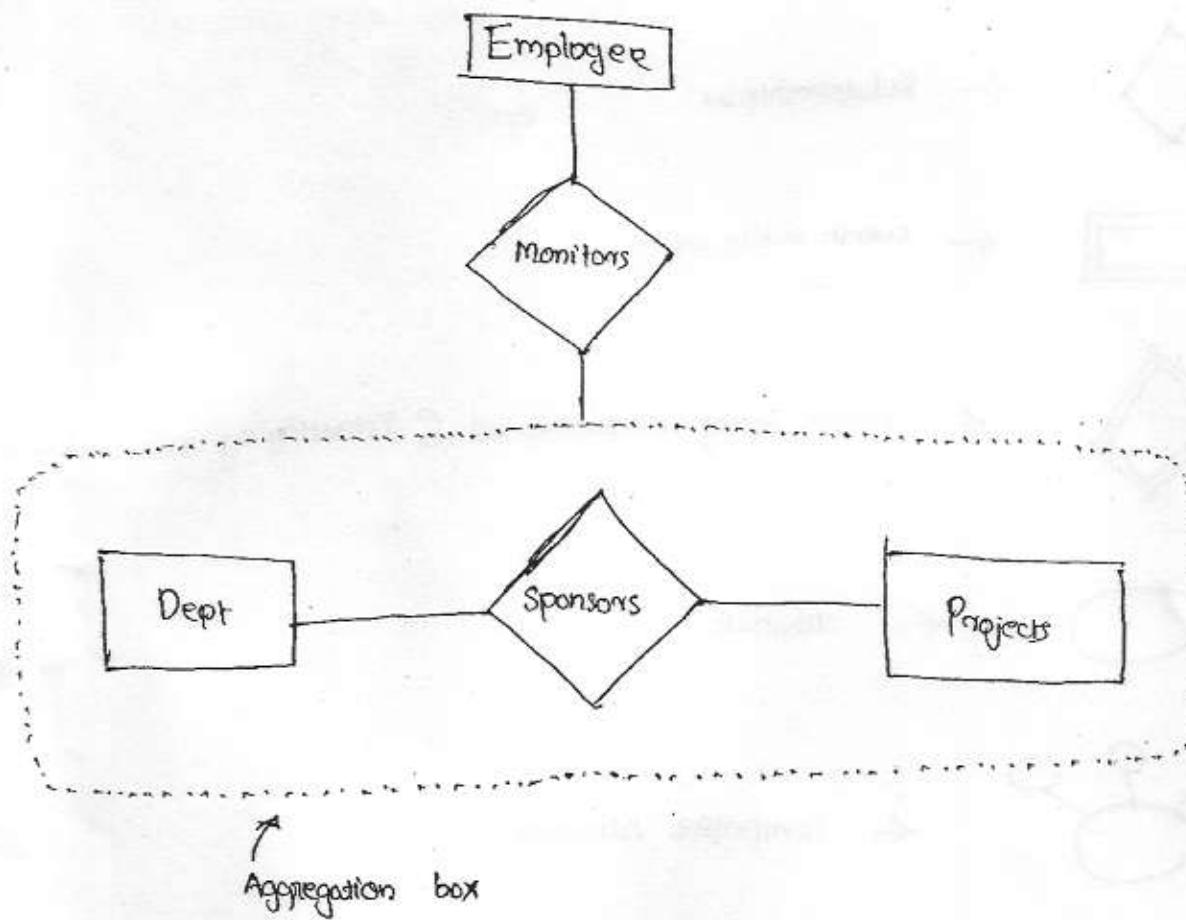
Class Hierarchy



Aggregation

11

Aggregation allows us to indicate that a relationship set participates in another relationship set.



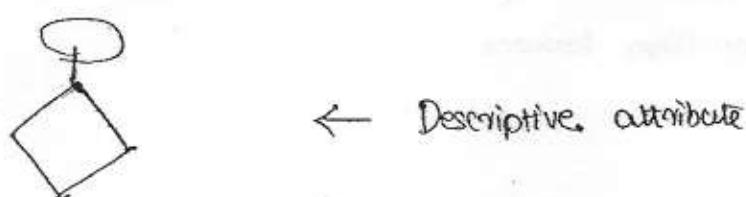
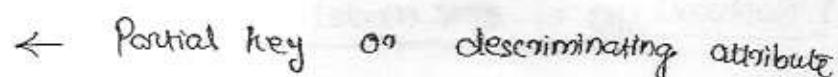
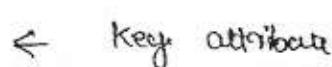
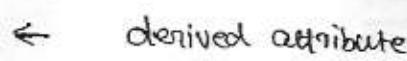
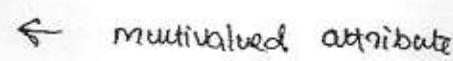
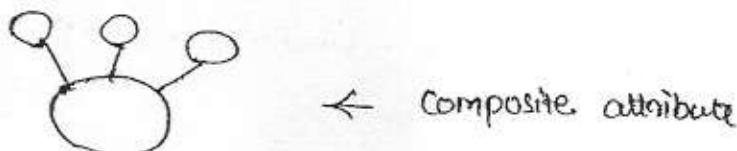
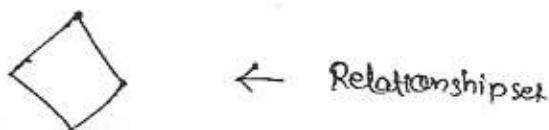
Advantages of E-R Model

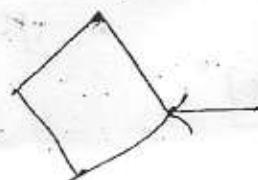
- 1) Easy to understand
- 2) It is an effective communication tool

Disadvantages of E-R Model

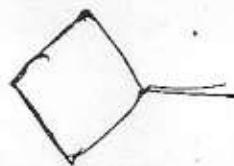
- 1) Limited constraint capability.
- 2) Loss of information content

E-R Components

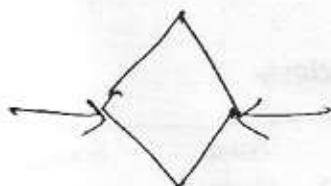




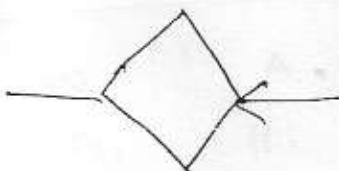
← Key constraint



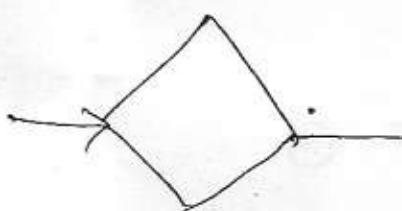
← total participation



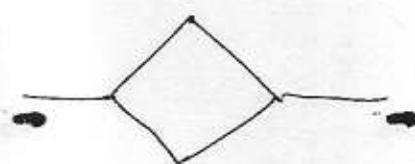
1:1 (One to one)



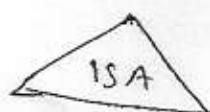
1:m (One to many)



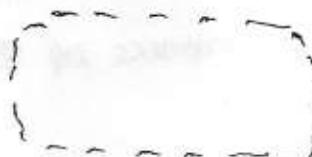
m:1 (many to one)



many to many (m:N)



← class hierarchy



← Aggregation box

Logical Database Design using "relational Model"

14

Relation :- → Rows & Columns
(Table) Records Attributes
 Tuples

described using

Relation Schema :- Structure

Relation instance :- tuples

Student			
Rno	Name	Branch	
Number(2)	char(10)	char(3)	
Primary key	Not null		
1	A	CSE	
2	B	IT	

Degree of a relation :-

Degree specifies No. of columns present in a tuple ③

Cardinality of a relation :-

Specifies no. of rows ②

RDBMS :- Collection of relations

Integrity Constraints :-

Is a condition specified on a database schema and restricts the data that can be stored in an instance of the database.

Ex:- PRIMARY KEY, NOT NULL, UNIQUE

Legal Instance :- The instance which satisfies all the integrity constraints specified on a database schema.

→ Otherwise such an instance is called Illegal instance.

Key Constraints

15

It is a set of fields of a relation has a unique identifier for a tuple. That is each tuple in a relation is identified using a set of attributes.

Student (RNo, Name, father, Branch, Passport)

- 1) $RNo \leftarrow \text{key}$
- 2) $(Name, father) \leftarrow \text{key}$
 - A. P
 - B. Q
- 3) $\text{Passport} \leftarrow \text{key}$

- 4) $(RNo, Name) \leftarrow \text{key}$
- 5) $(RNo, Passport) \leftarrow \text{key}$

I) Candidate key :-

It is a minimal set of attributes which uniquely identifies a tuple in a relation

ex:-

RNo , (Name, father) , Passport

II) Super key :-

It is a set of attributes which contains a key (candidate key)

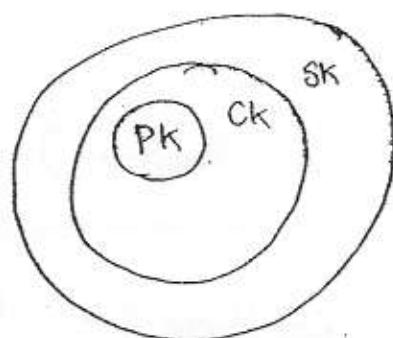
ex:- (RNo, Passport) , (RNo, Name), RNo, Passport, Name, f

→ Every candidate key is called a super key, but every super key need not be a candidate key

III) Primary key :-

Among all the available candidate keys one can be identified as primary key.

Ex:- Rno.



IV) Foreign key constraint (Referential Integrity Constraint)

Student			Registration		
Rno	Name	Branch	C.No	Cname	Rno
1	A	CSE	101	DBM'S	1
2	B	IT	102	CD	1
3	C	CSE	103	TOC	3
			104	PL	

(Referenced relation) (Parent) (Referencing relation) (Child)

→ The values present in foreign key must be present in primary key of referenced relation. Foreign key may contain duplicates and null values.



Parent table

✓ Insert < 4 D ECE >

✗ Delete < 1 A CSE >

child table

✗ Insert < 105 GT 5 >

✓ Delete < 103 TOC 3 >

→ Deletion from the referenced relation and insertion into the referencing relation may violate foreign key constraint.

foreign key with ondelete cascade

17

When data from the parent table is deleted. The related data from the child table also to be deleted cascadedly. (both tables)

- A Relation can act as both parent and child, ie, a relation may contain a primary key and a foreign key that refers to the same relation.

Q4 Consider a relation $R(A, B)$ where A is primary key and B is foreign key referencing the same relation. Then which of the following row sequence can be inserted successfully into R .

- a) $(1, \text{null}) (2, 1) (2, 2) (3, 2)$
- b) $(\text{null}, 1) (1, 2) (2, 3) (3, 4)$
- c) $(1, \text{null}) (2, 1) (3, 2) (4, 2)$
- d) all

Q5 Consider a relation $R(A, B)$ where A is a primary key and B is a foreign key referencing A with on-delete cascade. Consider the following relation instance of R .

$R(\overset{\leftarrow}{A} \underset{\text{with casc}}{B})$

2 4

3 4

4 3

5 2

7 2

9 5

6 4

what are the tuples that must be additionally deleted to 18

preserve the referential integrity constraint when the tuple 2,4 deleted

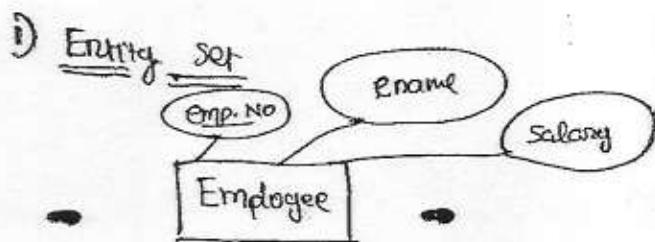
is.

- a) (3,4) (4,3)
- b) (3,4) (4,3) (6,4)
- c) (5,2) (7,2)
- ~~d) (5,2) (7,2) (9,5)~~

first which tuple is deleted?

- a) (5,2)
- b) (7,2)
- ~~c) (9,5)~~
- d) all at once

E-R to Relational Model

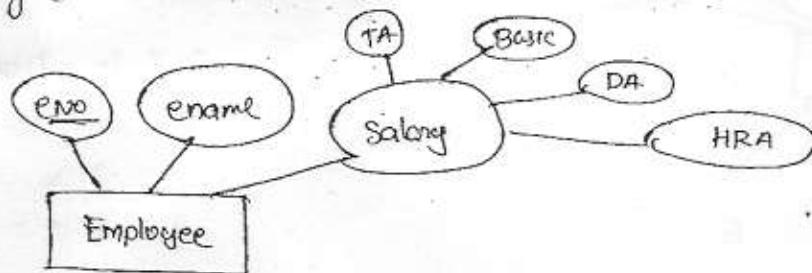


Employee

eno ename salary

- An entity set is mapped with a relation
- The attributes of the relation include the attributes of an entity
- The key attribute of an entity becomes primary key of the relation.

2) Entity Set with Composite attribute

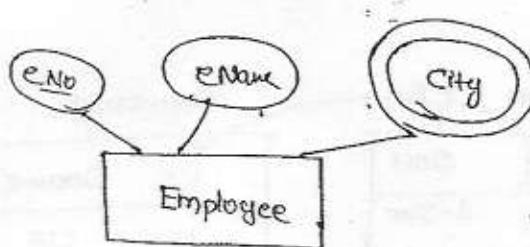


Employee

<u>eNo</u>	ename	Basic	TA	Basic	DA	HRA
1	A	5000	3000	5000	1200	

The attributes of a relation includes the simple attribute of an entity set

3) Entity Set with multivalued attribute

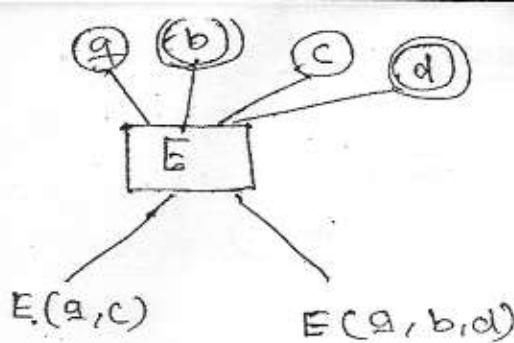


Employee

<u>e-no</u>	ename
1	A
2	B

FK Emp-addressing	
e-No	City
1	Hyd
1	Bang
2	Bang
2	Pune

Note :- If an entity contains multivalued attributes it is represented with two relations one - with all simple attributes and the other with key and multivalued attributes.

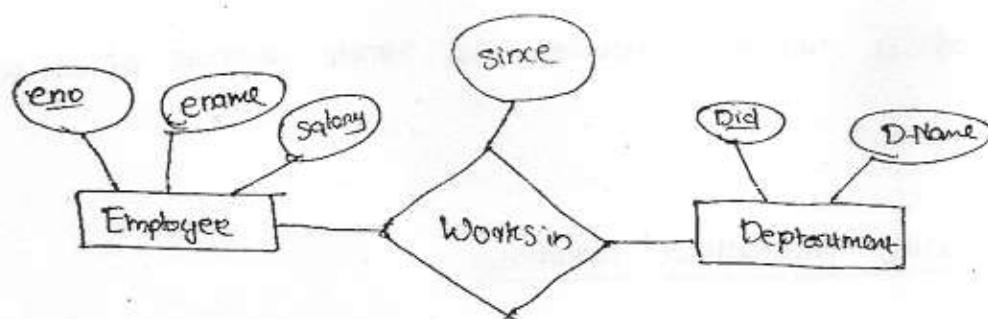


20
 6 - 8:30 - Digital
 9 - 1 - TOC
 2 - 5:30 TOC
 6 - 8:30 - PBMS } 372



04-12-13

Translating relationship set into a table



Three tables resulting from the translation:

- Employee**: Contains attributes e-no, e-name, and Salary. Data: 1 A 5k, 2 B 9k.
- Works_in**: Contains attributes eno, Did, and Since. Data: 1 101 1-Jan, 1 102 2-Feb, 2 102 1-Feb.
- Department**: Contains attributes Did and Dname. Data: 101 CSE, 102 IT.

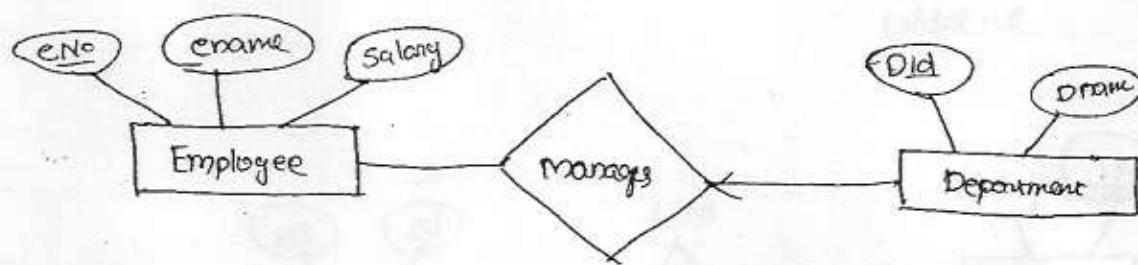
A Relationship set is mapped into a relation, the attributes of a relation includes

- (i) The key attributes of the participating relation and are declared as foreign keys to the respective relation
- (ii) Descriptive attributes (if any)
- (iii) Set of Non descriptive attributes is the primary key of the relation.

Relationship set with key constraint

21

Each dept is required to have almost one employee as a manager.



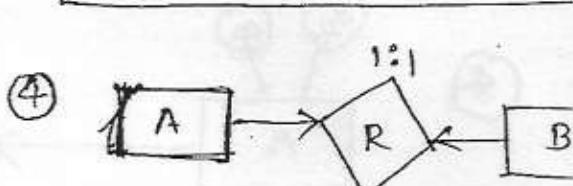
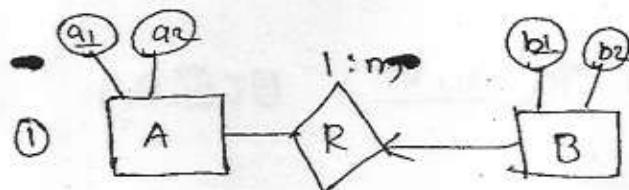
Employee			Manages		Department	
eno	ename	salary	eno	Did	Did	Dname
1	A	5k	1	101	101	CSE
2	B	9k	1	102	102	IT
			2	103	103	ECE

merge these tables

Foreign key

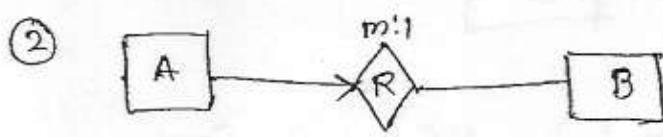
Department

Did	Dname	eno
101	CSE	1
102	IT	1
103	ECE	2



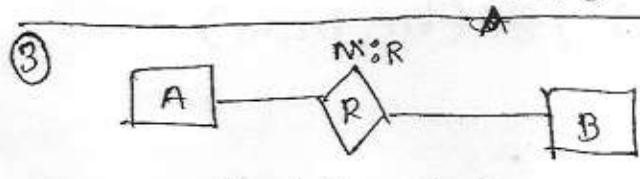
2-table $\frac{A, BR}{A(a_1, a_2), BR(b_1, b_2, a_1)}$

FK
↑

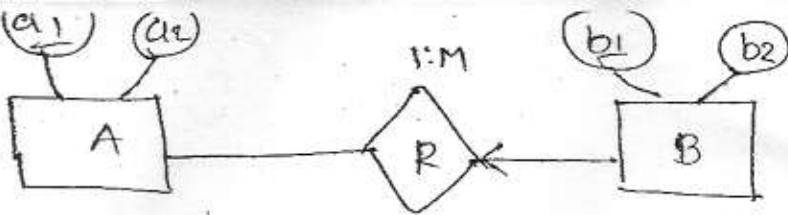


2-table AR, B

2-tables AR, B
or
A, BR

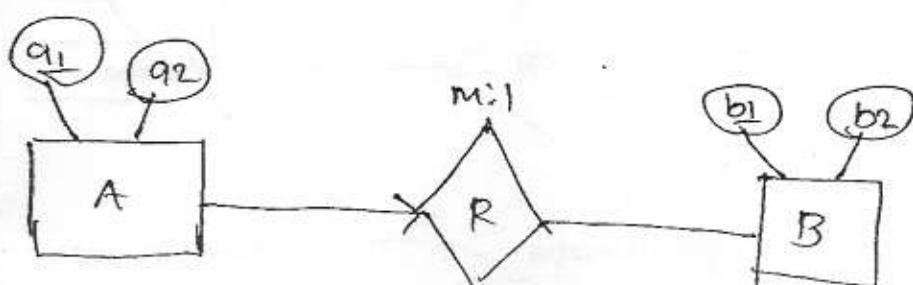


3-table A, R, B



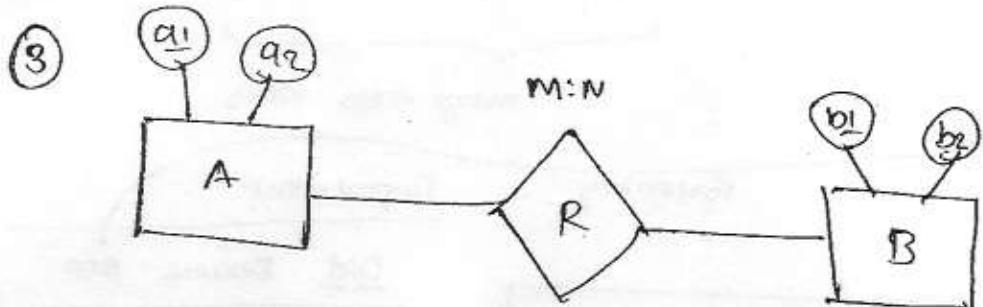
$A(a_1, a_2), B(b_1, b_2)$, $R(a_1, a_2, b_1)$

2-table



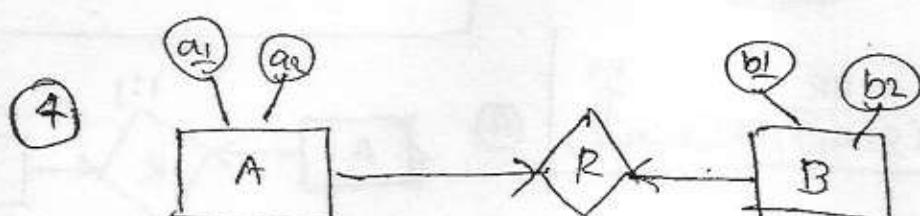
2-table

$A(a_1, a_2), B(b_1, b_2)$, $R(a_1, a_2, b_1)$



3-table

$A(a_1, a_2), R(a_1, b_1), B(b_1, b_2)$



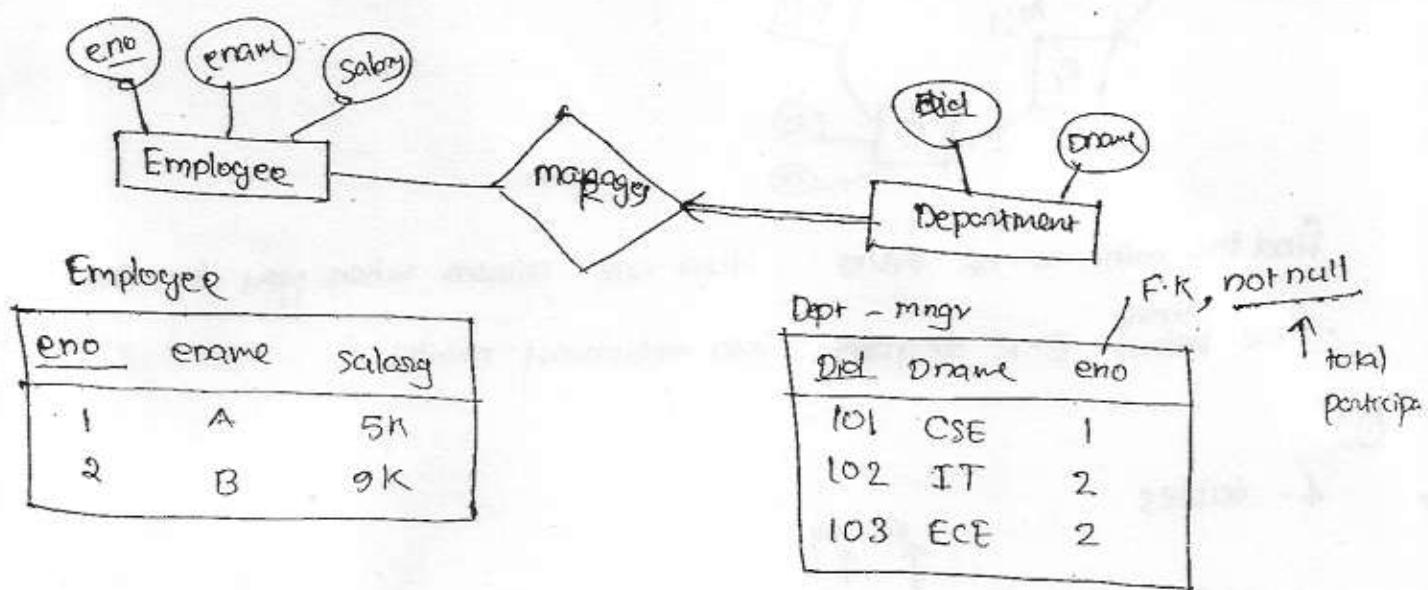
2-table.

$A(a_1, a_2), B(b_1, b_2)$, $R(a_1, b_1)$

$A(a_1, a_2), R(a_1, b_1), B(b_1, b_2)$

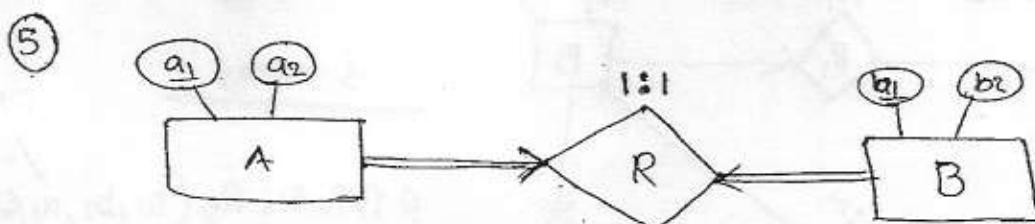
Relationship set with key constraint & Participation constraint.

- " Each Dept is required to have exactly one employee as a manager"



1) If there is a key constraint merge the relationship set table with an entity set table

2) If the entity set totally participating with relationship set then foreignkey with not null constraint

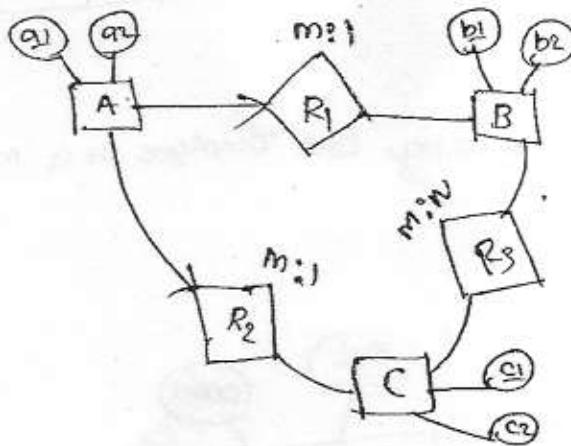


1 table. ARB (a_1, a_2, b_1, b_2)

Note: If there is a key constraint from both the sides of an entity set with total participation then we represent that binary relationship using single table.

Ques

24



Find the min. no. of tables that are possible when you translate the above E-R diagram into relational model.

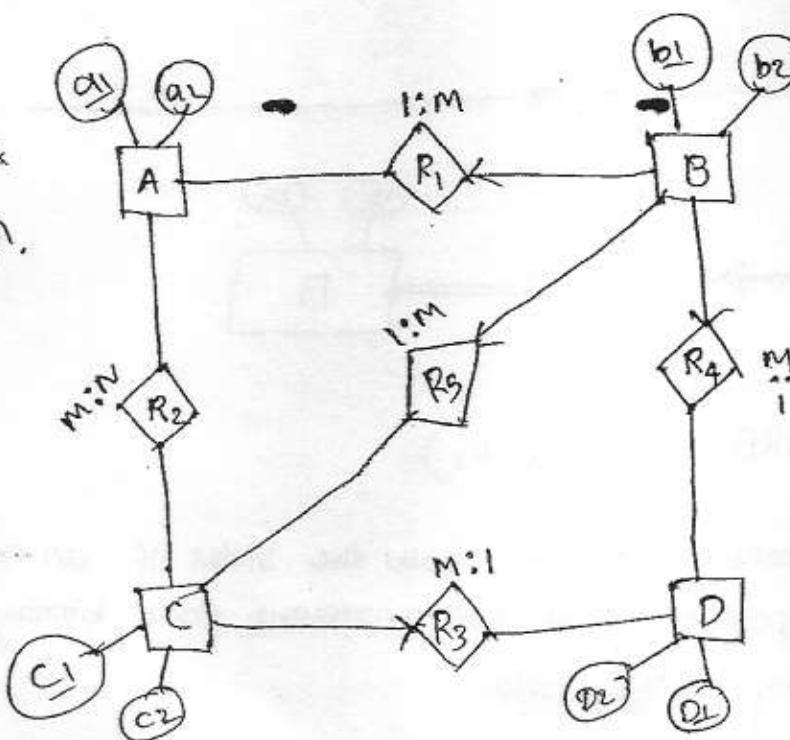
Ans

4-tables

- 1) AR₁, R₂ (a₁, a₂, b₁, c₁)
- 2) BC (b₁, b₂)
- 3) R₃ (b₁, c₁)
- 4) C (c₁, c₂)

Ques

What min. no. of tables = ?

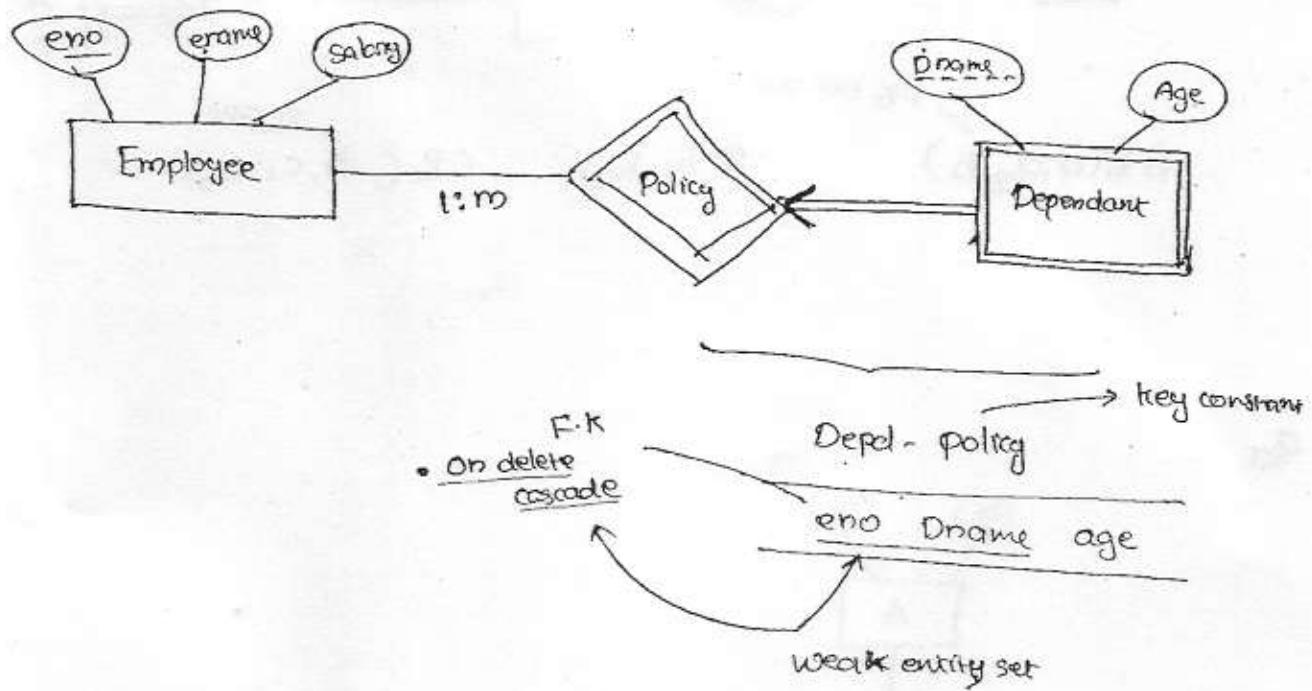


5-tables

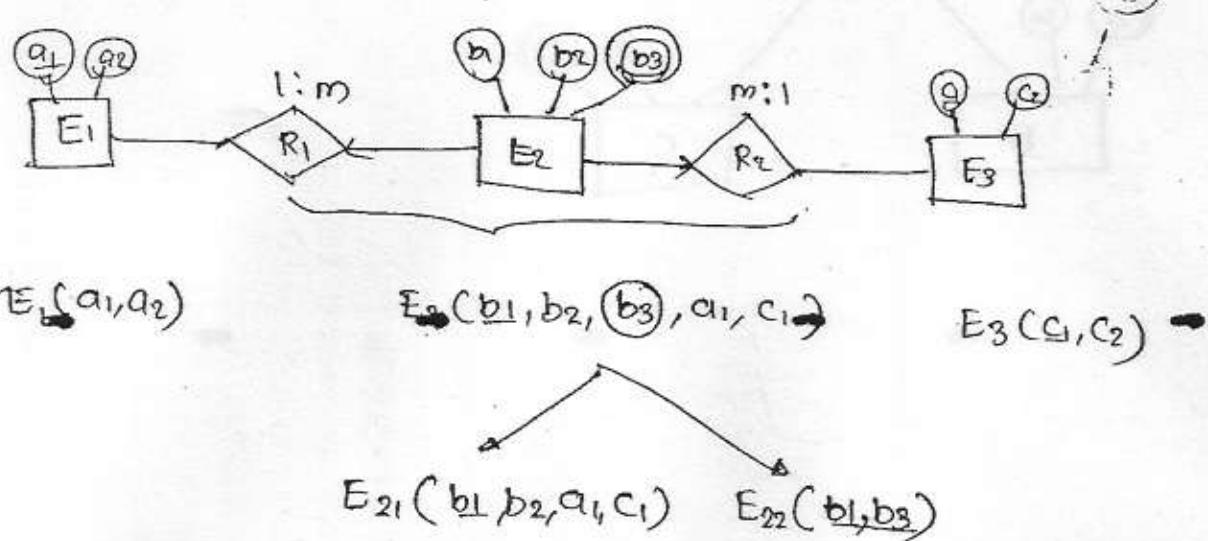
- 1) BR₁, R₄, R₅ (b₁, b₂, a₁, c₁, d₁)
- 2) A (a₁, a₂)
- 3) R₂ (a₁, c₁) FK
- 4) D (d₁, d₂)
- 5) CR₃ (c₁, c₂, d₁) FK

Weak entity Set

25

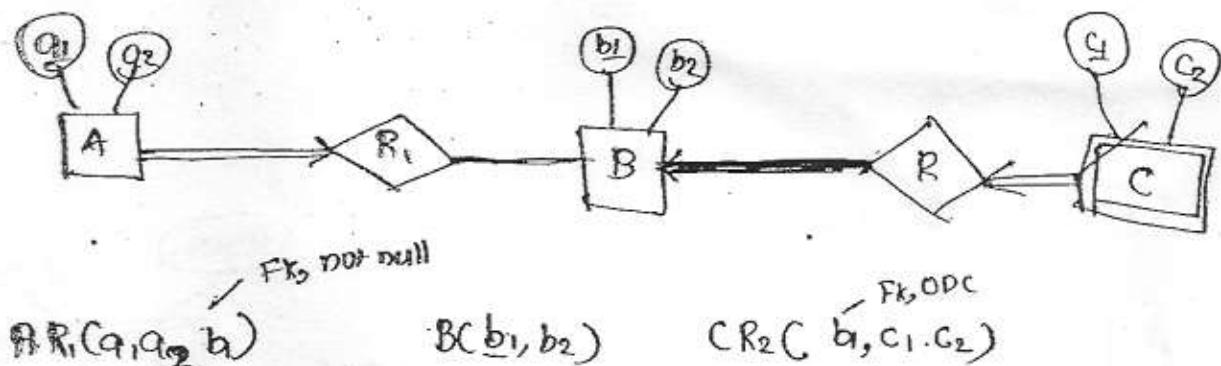


Ques

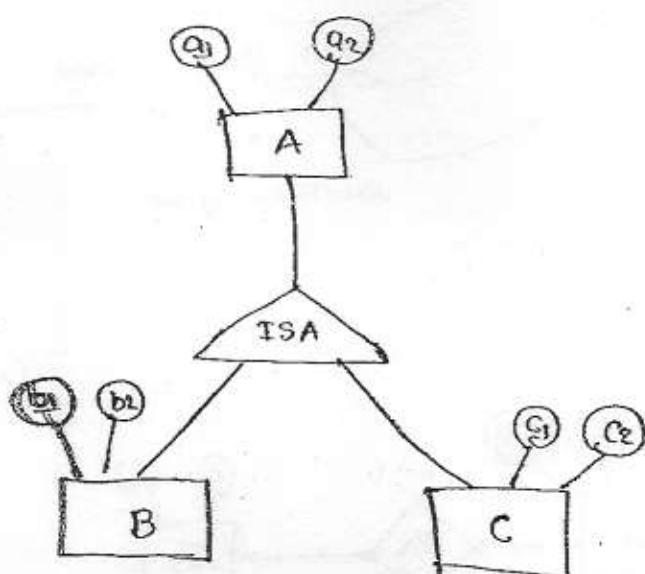


Total no. of tables (min) = 3

Q4 min. no. of tables = ?



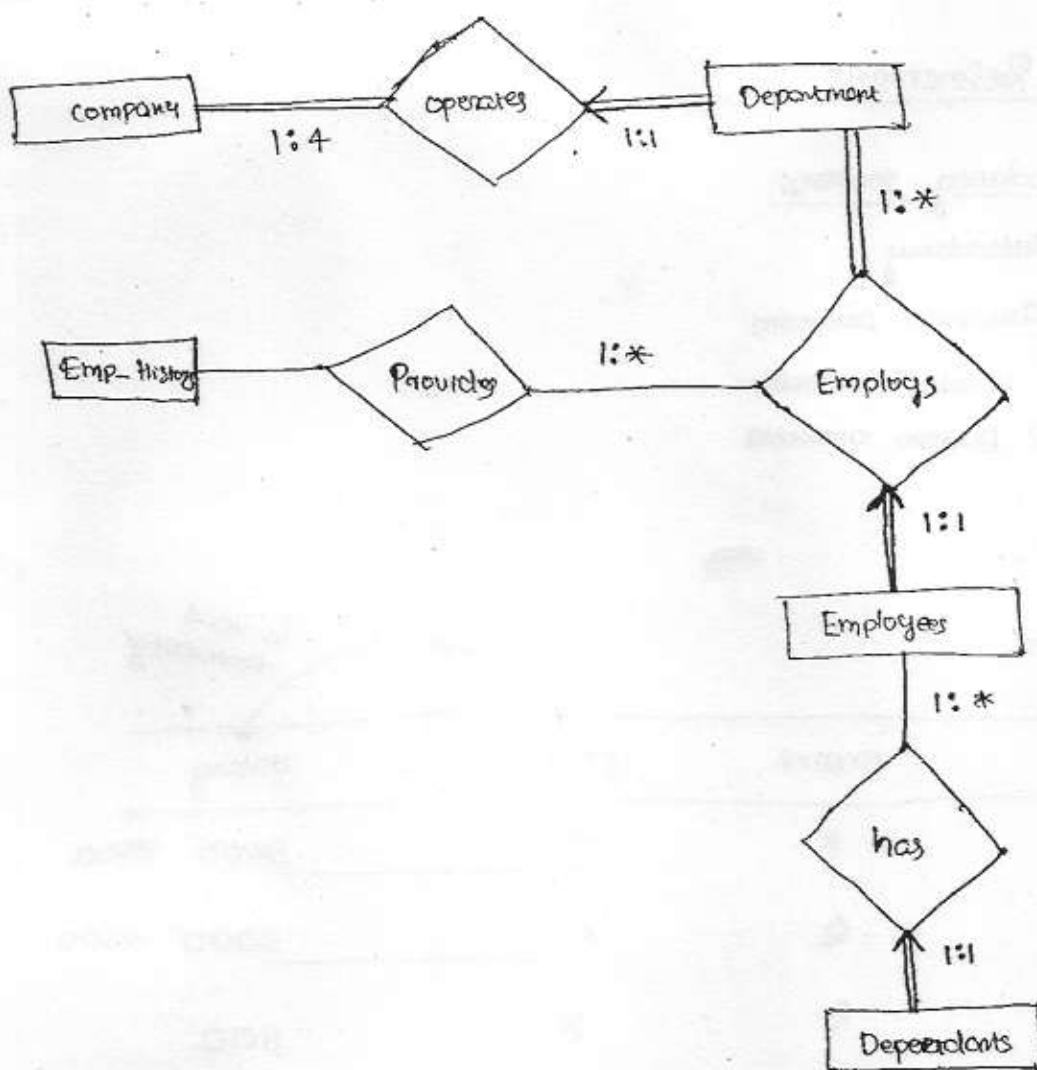
Q5



Level-2

(Q1)

(a)(b)



NORMALIZATION

Schema Refinement

- Redundancy problems

- 1) Redundancy
- 2) Insertion anomalies
- 3) Update anomalies
- 4) Deletion anomalies

Employee

<u>eno</u>	<u>ename</u>	<u>grade</u>	<u>salary</u>
1	P	A	9000 9500
2	Q	A	9000 9500
3	R	B	600
4	S	B	600
5	T	A	9000 9500
6	R	A	9800 X

functional dependency

- 1) Wastage of space
- 2) Multiple updation/insertion needed
- 3) Deletion causes loss of data.

→ Decomposition is the solution for these problems.

eno	ename	grade
1	P	A
2	Q	A
3	R	B
4	S	B
5	T	A
6	R	A ✓

Emp-salary	
Grade	salary
A	9000 → 9500
B	600

All the problems arising due to redundancy is resolved using decomposition.

Functional Dependancy (F.D)

$$\begin{array}{c} X \rightarrow Y \\ \text{determinant} \qquad \qquad \text{dependant} \end{array}$$

$X \leftarrow Y$ can be one or many attributes

$$\begin{array}{l} X \rightarrow Y \\ \hline X_1 Y_1 \checkmark \\ X_2 Y_1 \checkmark \\ X_1 Y_1 \checkmark \\ X_2 Y_3 \checkmark \\ X_2 Y_3 \checkmark \\ \boxed{X_2 Y_2} X \\ X_3 Y_1 \checkmark \\ X_4 Y_4 \checkmark \\ \hline \end{array} \quad \begin{array}{c} X \rightarrow Y \\ \hline X_1 Y_1 \\ X_2 Y_3 \\ X_3 Y_1 \\ X_4 Y_4 \\ \hline \end{array}$$

- The functional dependency is the generalization of the concept of key.
- $X \rightarrow Y$ says that if two tuples agree on the value of attribute X they must agree on the value in attribute Y .

Ques Consider a relation R (A B C) having the tuples

30

R (A B C)

1 2 3

4 2 3

5 3 3

Then which of the following dependencies can you infer does not hold over R.

a) $A \rightarrow B$.

b) ~~$BC \rightarrow A$~~

c) $B \rightarrow C$

d) $AC \rightarrow B$

Ques Consider the following relation instance

$\begin{array}{c} X \ Y \ Z \\ \hline 1 \ 4 \ 3 \end{array}$

1 5 3

4 6 3

3 2 2

which of the following functional dependencies are satisfied by the above instance. ?

a) $X \ Y \rightarrow Z, \ Z \rightarrow Y$

b) ~~$X \ Z \rightarrow X, \ Y \rightarrow Z$~~

c) $X \ Y \rightarrow Z, \ Z \rightarrow X$ Trivial

d) $X \ Z \rightarrow Y, \ Y \rightarrow Z$

- 1) Trivial functional Dependencies
- 2) Non-Trivial functional Dependencies.

1) Trivial F.D :-

A functional dependency $X \rightarrow Y$ is said to be trivial iff $Y \subseteq X$. In other words if R.H.S of some dependency is the subset of L.H.S of the dependency then it is called trivial F.D.

Ex:-

$$\begin{aligned}AB &\rightarrow A \\AB &\rightarrow B \\AB &\rightarrow AB\end{aligned}$$

2) Non-Trivial F.D :-

If there is atleast one attribute in the R.H.S i.e, not part of the L.H.S such dependency is called non-trivial functional dependency

Ex:-

~~because~~
 $AB \rightarrow BC$: Non-trivial
 $AB \rightarrow CD$: Completely non-trivial

[6-830 - Digital
9-536 - DBMS
@312]

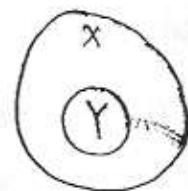
Closure properties of f.D's (closure)

1) Reflexivity

if $X \supseteq Y$ then $X \rightarrow Y$ is a dependency

Ex:-

$$AB \rightarrow A$$



2) Augmentation

if $X \rightarrow Y$ is a dependency

$XZ \rightarrow YZ$ is a dependency.

3) Transitivity

if $X \rightarrow Y$ is a dependency and $Y \rightarrow Z$ is a dependency
then $X \rightarrow Z$ is a dependency

4) Union property

if $X \rightarrow Y$ is a dependency and $X \rightarrow Z$ is a dependency.

then $X \rightarrow YZ$ is a dependency

5) Decomposition

if $X \rightarrow YZ$ is a dependency then $X \rightarrow Y$ and $X \rightarrow Z$ is a dependency.

It is the set of all F.D's. that can be determined using the given set of functional dependencies 'F.' and is denoted by F^+ (F-closure)

Ex:-

$R(ABC)$

$$F: \{A \rightarrow B, B \rightarrow C\}$$

$$F^+: \{A \rightarrow B, B \rightarrow C, A \rightarrow C, AC \rightarrow BC, AB \rightarrow BC, AB \rightarrow AC, \dots\}$$

Ques Find the no. of F.D's in F-closure (F^+) for a relation with 2-attributes?

A/w

$R(A, B)$

$$\begin{array}{c} \overbrace{X} \\ \phi \\ A \\ B \\ AB \end{array} \longrightarrow \begin{array}{c} \overbrace{Y} \\ \phi \\ A \\ B \\ AB \end{array}$$

$$\begin{array}{c} \phi \\ A \\ B \\ AB \end{array}$$

$4 \times 4 = 16$			
$\phi \rightarrow \phi$	$A \rightarrow \phi$	$B \rightarrow \phi$	$AB \rightarrow \phi$
$\phi \rightarrow A$	$A \rightarrow A$	$B \rightarrow A$	$AB \rightarrow A$
$\phi \rightarrow B$	$A \rightarrow B$	$B \rightarrow B$	$AB \rightarrow B$
$\phi \rightarrow AB$	$A \rightarrow AB$	$B \rightarrow AB$	$AB \rightarrow AB$

$$F^+ \# \text{dependencies} = \underline{\underline{16}}$$

Ques Consider $R(A, B, C)$

$$F^+ = \text{64 combinations}$$

$$X \rightarrow X$$

$$2^3 \times 2^3 = 64$$

Ques $R(n\text{-attribute})$

$$F^+ = \frac{2^n \rightarrow 2^n}{2^n \times 2^n = \underline{\underline{2^{2n}}}}$$

Closure Set of attributes (X^+)

Ex:- $R(A, B, C)$

$$F: \{A \rightarrow B, B \rightarrow C\} \quad A^+ = \{A, B, C\}$$

The set of all attributes that can be determined using the given set X of attributes is called attribute closure and is denoted by X^+

Ques $R(A, B, C, D, E, F)$

$$F: \{AB \rightarrow C, BC \rightarrow AD, D \rightarrow E, CE \rightarrow B\}$$

$$\text{Find } (AB)^+ = ?$$

$$\text{Ans} \quad AB^+ = \{A, B, C, D, E\} \quad CE^+ = \{B, A, D, C, E\}$$

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

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

Ques Consider a relation with F.D.'s

35

$$F: \{ A+B \rightarrow CD, AF \rightarrow D, DE \rightarrow F, C \rightarrow G, F \rightarrow E, G \rightarrow A \}$$

first $CF^+ = \{ C, F, G, E, A, D \}$

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

$$AF^+ = \{ A, F, D, E \}$$

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

Ques Consider a relation $R(A, B)$ with $f: \{ A \rightarrow B \}$

Find all dependancies in F^+

$$F^+ = \{ A \rightarrow B, AB \rightarrow B, A \rightarrow A, B \rightarrow B, A \rightarrow AB, \phi \rightarrow \phi, A \rightarrow \phi, B \rightarrow \phi, AB \rightarrow \phi, AB \rightarrow AB, AB \rightarrow A \}$$

$$\phi^+ = \phi = 1$$

$$A^+ = A, B = 4$$

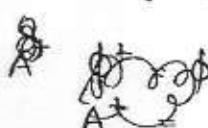
$$B^+ = B = 2$$

$$AB^+ = A, B = 4$$

$$\underline{\underline{11}}$$

Ques $R(A, B, C)$

$$f: \{ A \rightarrow B, B \rightarrow C \} \quad \text{find } F^+ = \underline{\underline{\quad}} ?$$



$$\phi^+ = \phi^- = 1$$

$$A^+ = A, B, C = 8$$

$$B^+ = B, C = 4$$

$$C^+ = C = 2$$

$$AB^+ = A, B, C = 8$$

$$AC^+ = A, B, C = 8$$

$$BC^+ = B, C = 4$$

$$ABC^+ = A, B, C = 8$$

43

Ques $\mathbb{R}(A, B, C)$

$$f : \{ A \rightarrow B, B \rightarrow C, C \rightarrow A, B \rightarrow A \}$$

$$f^+ = \underline{\quad ? \quad}$$

$$\phi^+ = \phi^- = 1$$

$$A^+ = A, B, C = 8$$

$$B^+ = A, B, C = 8$$

$$C^+ = A, B, C = 8$$

$$AB^+ = A, B, C = 8$$

$$AC^+ = A, B, C = 8$$

$$BC^+ = A, B, C = 8$$

$$ABC^+ = A, B, C = 8$$

57

Applications of attribute closure

37

- ① To find additional functional dependencies
- ② To find key's of a relation
- ③ To find equivalences of functional dependencies
- ④ To find minimal set of functional dependencies.

- 1] To find additional functional dependencies

Ex:- $R(A, B, C, D)$

$$f: \{ A \rightarrow BC, B \rightarrow CD, D \rightarrow AB \}$$

then find $AD \rightarrow C$ is possible?

$$\overbrace{AD^+}^{\uparrow} = \{ A, D, B, C \}$$

Qn Consider a Relation $R(ABCDE)$ with FD's

$$f: \{ A \rightarrow B, A \rightarrow C, CD \rightarrow E, B \rightarrow D, E \rightarrow A \}$$

which of the following F.D's is not implied by the above set?

a) $CD \rightarrow AC$

$$\overbrace{CD^+}^{\uparrow} = \{ C, D, E, A, B \}$$

b) $BD \rightarrow CD$

$$\times \overbrace{BD^+}^{\uparrow} = \{ B, D \}$$

c) $BC \rightarrow CD$

$$\overbrace{BC^+}^{\uparrow} = \{ B, C, D, E, A \}$$

d) $AC \rightarrow BC$

$$\overbrace{AC^+}^{\uparrow} = \{ A, C, B, D, E \}$$

② To find keys of a relation

The set of all attributes that can be determined using the given set of attributes is called attribute closure and is denoted by X^+ . If X^+ contains all the attributes of a relation then X is called superkey of R , where R is a relation.

If X is a minimal set then X is called Candidate key of R .

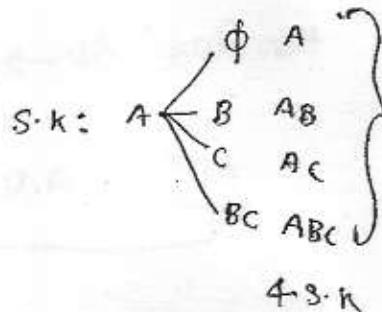
① $R(ABC)$

$$F.D: \{A+B, B+C\}$$

$$A^+ = \{A, B, C\} \leftarrow \text{key}$$

$$B^+ = BC$$

C.K: A



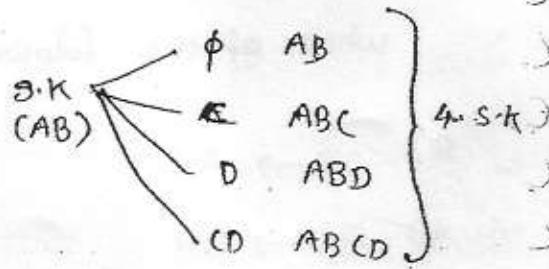
② $R(ABCD)$

$$f: \{AB \rightarrow C, B \rightarrow D\}$$

$$AB^+ = \{A, B, C, D\}, \text{key}$$

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

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



③ R(A,B,C,D)

39

$$f: \{ AB \rightarrow CD, CD \rightarrow AB \}$$

$$\cancel{CD} \rightarrow A$$

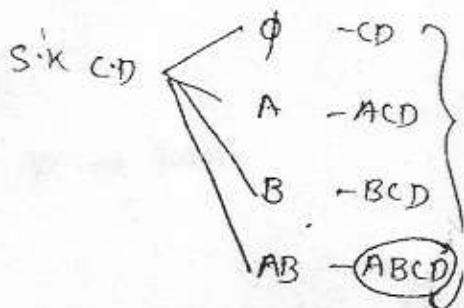
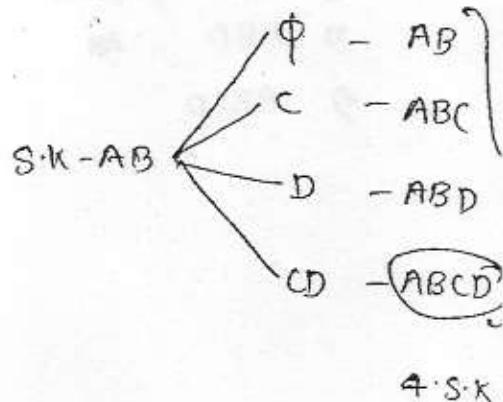
$$\cancel{CD} \rightarrow B$$

$$AB \rightarrow C$$

$$AB \rightarrow D$$

$$AB^+ = \{ AB, CD \} : C.K$$

$$CD^+ = \{ CD, AB \} : C.K$$

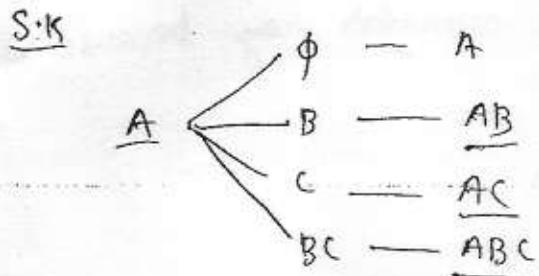


of superkeys = 7

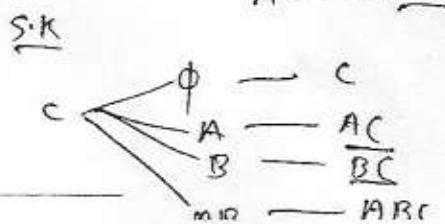
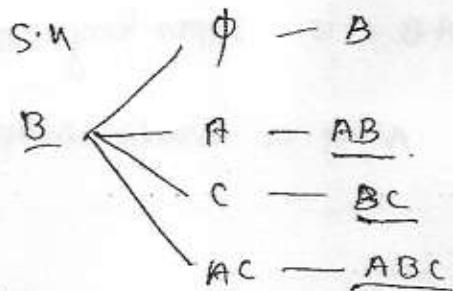
④ R(A,B,C)

$$f: \{ A \rightarrow B, B \rightarrow C, C \rightarrow A \}$$

$$\begin{aligned}
 A^+ &= \{ A, B, C \} : C.K \\
 B^+ &= \{ B, C, A \} : C.K \\
 C^+ &= \{ C, A, B \} : C.K
 \end{aligned}
 \left. \begin{array}{l} \\ \\ \end{array} \right\} 3 \cdot C.K$$

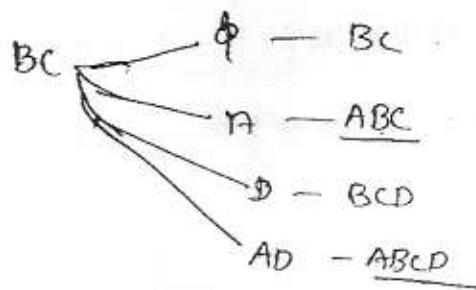
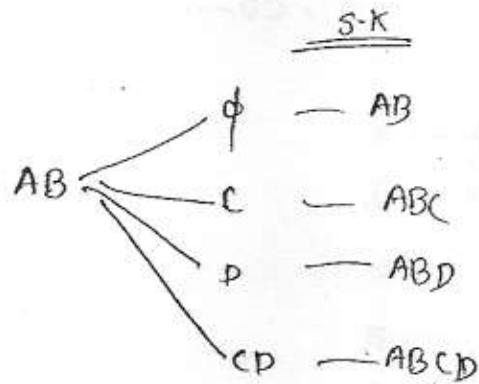


of superkeys = 7



5) Consider $R(ABCD)$ with C.R: AB, BC find the no. of Superkeys? (40)

- 1) AB ~~BC~~
- 2) ABC ~~BCD~~
- 3) ABD ~~A~~
- 4) ABCD



Total no. of Superkeys = 6

6) $R(ABCD)$,

f: $\{AB \rightarrow CD, A \rightarrow B\}$ find is AB a candidate key or only superkey?

$AB^+ = \{A, B, C, D\}$: key - Superkey

$A^+ = \{A, B, C, D\}$: key - ~~candidate key~~ candidate key.

$B^+ = \{B\}$

\therefore AB is superkey but not candidate key because A is a candidate key.

⑦ R(A B C D) with F: D $\{ A \rightarrow B, B \rightarrow C \}$ find C.K? 41

Ans

$$A^+ = \{ A, B, C \} \quad x$$

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

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

$$D^+ = \{ D \} \quad x$$

$$AD^+ = \{ A, B, C, D \} : \underline{\text{candidate key}}$$

hidden attribute: The attributes that are not part of RHS of dependence

Hidden candidate key.

⑧ R(A B C D E)

F: $\{ AB \rightarrow C, C \rightarrow D \}$ final C.K?

Ans

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

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

$$ABE^+ = \{ A, B, C, D, E \} \quad \checkmark$$

$$C.K = \underline{\underline{ABE}}$$

⑨ R(A B C), F: $\{ AB \rightarrow C, C \rightarrow A \}$

~~A B C~~

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

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

C.K: AB

CB ($C \rightarrow A$)

Note :- The attributes of a key are called key attributes or prime attribute. If prime attribute appeared on the R.H.S of some dependency that can be replaced with its L.H.S to get additional candidate keys.

(16)

$R(ABCD) \quad F: \{ AB \rightarrow C, B \rightarrow D, C \rightarrow B, D \rightarrow B \}$

find all C.K's of R. ?

$$\begin{array}{l} \text{C.K} \\ \hline \begin{array}{c} AB \\ AC \\ AD \end{array} \end{array} \quad \begin{array}{l} AB^+ = \{ A, B, C, D \} \\ \left[\begin{array}{l} \because C \rightarrow B \\ \because CD \rightarrow B \end{array} \right] \end{array}$$

Candidate keys

$$\begin{array}{l} AB \\ AC \\ AD \end{array}$$

3 Candidate keys.

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

$F: \{ A \rightarrow B, BC \rightarrow D, D \rightarrow AE \}$ find C.N's of R?

Ans

$$\begin{array}{l} 1) BC^+ = \{ B, C, D, A, E \} : \text{C.K} \\ 2) \underline{AC = \text{C.K.}} \end{array}$$

$$D \rightarrow AE$$

$$D \rightarrow A, D \rightarrow E$$

$$3) \underline{DC \in \text{C.K}}$$

(12) $R(C A B C D)$

$F: \{ AB \rightarrow C, C \rightarrow A, B \rightarrow D, D \rightarrow B \}$ $C.K = ?$

Ans

1) $AB : CK$

2) $CB : CK$

3) $CD : CK$

4) ~~AD~~ $: CK$

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

$\underline{2) CB} \quad C \rightarrow A \Rightarrow$

$3) AD \quad D \rightarrow B \Rightarrow$

~~4) CD~~

4) CD

(13) $R(C A B C D E)$

$F: \{ AB \rightarrow C, CD \rightarrow E, DE \rightarrow B \}$

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

$CD^+ = \{ C, D, E, B \}$

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

$AB^+ = \{ A, D \}$

$ABD^+ = \{ A, B, C, D, E \} : CK$

$ACD^+ = \{ A, B, C, D, E \} : CK$

$ADE^+ = \{ A, B, C, D, E \} : CK$

(14) RC

44

(14) ABCDEH)

$$f: \{ A \rightarrow B, BC \rightarrow D, E \rightarrow C, D \rightarrow A \} \quad c \cdot K = ?$$

~~ANSWER~~

- 1) AEH : C.K
- 2) DEH : C.K
- 3) BEH : C.K

~~ANSWER~~

$$\begin{aligned} A \cdot E \cdot H^+ &= \{A, E, H, B, C, D\} \\ \underline{D \cdot E \cdot H^+} &\quad D \rightarrow A \\ B \cdot C \cdot E \cdot H &\quad BC \rightarrow D \\ \underline{B \cdot E \cdot H} &\quad E \rightarrow C \end{aligned}$$

3) To find Equivalence of F.D's

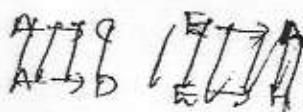
To sets of F.D's ' F ' and ' G ' are said to be equivalent iff $F^+ = G^+$, hence equivalence means that every F.D's in F can be inferred using G . and Every F.D in G can be inferred using F . ie, $F = G$ iff both

f covers G and G covers F holds.

Ex:- Consider the following two sets of F.D's

$$f: \{ A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H \}$$

$G: \{ A \rightarrow CD, E \rightarrow AH \}$ Is there two one equivalent?

Ans:

F covers G₁A⁺ $A \rightarrow CD$ Compute A^+ using F

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

↑↑

 $E \rightarrow AH$ compute E^+ using F

$$\underline{E^+ = E, ADHC}$$

↑ ↑

G₁ covers F $A \rightarrow C$ compute A^+ using G₁

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

↑

 $AC \rightarrow D$

$$\underline{AC^+ = \{A, C, D\}}$$

↑

 $E \rightarrow AD$ $E \rightarrow H$ E^+ using G₁

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

↓ ↓ ↑ ↑

F-covers G₁ and G₁ covers F \Rightarrow Both F and G₁ are equivalent.② F: { $A \rightarrow B, B \rightarrow C, C \rightarrow D$ }G₁: { $A \rightarrow BC, C \rightarrow D$ }F-covers G₁ X

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

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

F-covers G₁ ✓

$A^+ = \{ABCD\}$

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

$C^+ = \{CD\}$

③ F: $\{ A \rightarrow B, AB \rightarrow C, D \rightarrow AC, D \rightarrow E \}$

G: $\{ A \rightarrow BC, D \rightarrow AB \}$

$$\begin{array}{l} F \text{ covers } G \\ \hline A^+ = \{ A, B, C \} \end{array}$$

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

$$\begin{array}{l} G \text{ covers } F \\ \hline \end{array}$$

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

$$D^+ = \{ D, A, B, C \} \times E \text{ not covered}$$

④

F: $\{ A \rightarrow B, B \rightarrow C, C \rightarrow A \}$

G: $\{ A \rightarrow BC, B \rightarrow A, C \rightarrow A \}$

$$\left. \begin{array}{l} F \text{ covers } G \\ G \text{ covers } F \end{array} \right\} \text{ equal.}$$

④

To find minimal set of functional dependencies

f : given set
 f^+ : minimal set

Ex:- ① f: $\{ A \rightarrow B, B \rightarrow C, A \rightarrow C \}$ possible by finding A^+ in G.

G: $\{ A \rightarrow B, B \rightarrow C \}$ - minimal set

② f: $\{ A \not\rightarrow C, A \rightarrow B \}$ $AB \rightarrow C$ is possible by finding AB^+ in G.

G: $\{ A \rightarrow C, A \rightarrow B \}$ - minimal set

A minimal cover for a set of F.D's 'F' is a set of F.D 'G₁' 47
that satisfies the property that every dependency in F. is
in G₁⁺. Then G₁ is called minimal set.

Procedure to find minimal set

Step

- ① Put the F.D's in the standard forms.

Ex:-

$$A \rightarrow BC \Rightarrow A \rightarrow B \wedge A \rightarrow C$$

Step

- ② Remove redundant F.D's

Ex:- $\{A \rightarrow B, B \rightarrow C, A \rightarrow C\} \Rightarrow \{A \rightarrow B, B \rightarrow C\}$

Step

- ③ Minimize L.H.S of each F.D

$$AB \rightarrow C$$

A - can be deleted when B⁺ contains A

B - can be deleted when A⁺ contains B

if there was any removal of variables in step ③

repeat ② after that ③ - repeat this till there is
no removal. \Rightarrow it will be minimal.

X:- find the minimal set of P.D. for the following

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

Soln:-

Step 1:

$$1) A \rightarrow C$$

$$2) AC \rightarrow D$$

$$3) E \rightarrow A$$

$$4) E \rightarrow D$$

$$5) E \rightarrow H$$

Step 2:-

$$\checkmark A \rightarrow C$$

Compute A^+ using 2, 3, 4, 5

$$A^+ = A \Rightarrow 1 \text{ needed.}$$

$$\checkmark AC \rightarrow D$$

Compute AC^+ using 1, 3, 4, 5

$$AC^+ = AC$$

$$\checkmark E \rightarrow A$$

$$E^+ \text{ (using 1, 2, 4, 5)}$$

$$= E, DH$$

$$E \rightarrow D$$

$$E^+ \text{ (using 1, 2, 3, 5)}$$

$$= E, A, H, CD$$

} you can get $E \rightarrow D$ without 4
 => delete it.

$$\checkmark E \rightarrow H \text{ (using 1, 2, 3, 4)}$$

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

$A \rightarrow C \rightarrow D$ if A deleted: $C^+ = \{C\} \Rightarrow A$ can not be deletedif C deleted: $A^+ = \{A, C, D\} \Rightarrow C$ can be deleted. $\Rightarrow A \cancel{\times} \rightarrow D \Rightarrow A \rightarrow D$ Ans Set = after step 31) $A \rightarrow C$ 2) $A \rightarrow D$ 3) $E \rightarrow A$ 4) $E \rightarrow H$ (repeat step 2) \Rightarrow Nothing will be eliminatedminimal set or minimal cover $= \{A \rightarrow C, A \rightarrow D, E \rightarrow A, E \rightarrow H\}$ Canonical cover $\{A \rightarrow CD$
 $E \rightarrow AH\}$

L.H.S of all the dependencies should be unique.

②

f: $\{A \rightarrow B, C \rightarrow B, D \rightarrow A, BC \rightarrow D, AC \rightarrow D\}$ Step 1

- 1) $A \rightarrow B$
- 2) $C \rightarrow B$
- 3) $D \rightarrow A$
- 4) $D \rightarrow B$
- 5) $D \rightarrow C$
- 6) $AC \rightarrow D$

Step 2

- ① ✓
 ② ✓
 ③ ✓
 ④ ✗

Step 3

- $A \rightarrow B$
-
- $C \rightarrow B$
-
- $D \rightarrow A$
-
- $D \rightarrow C$
-
- $AC \rightarrow D$

③

 $AB \rightarrow C$ $C \rightarrow B$ $A \rightarrow B$ Step 1 $AB \rightarrow C$ $C \rightarrow B$ $A \rightarrow B$ Step 2 $AB \rightarrow C$ $C \rightarrow B$ $A \rightarrow B$ Step 3 $A \rightarrow C$ $C \rightarrow B$

④ $A \rightarrow BC$
 $B \rightarrow C$
 $AB \rightarrow D$

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

$\{ A \rightarrow B$
 $B \rightarrow C$
 $A \rightarrow D \}$

$\{ A \rightarrow BD$
 $B \rightarrow C \}$

$A \rightarrow BD$
 $B \rightarrow C$

50
minimal canonical cover

P-160

Q-11

R(ABCDEFG)

F: $\{ A \rightarrow BC, AB \rightarrow DE, D \rightarrow EF, F \rightarrow A \}$

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

$AG^+ : C \cdot K$

$B^+ =$

$F \cdot G_1 : C \cdot K$

$F \rightarrow A$

$D \cdot G_1 : C \cdot K$

$D \rightarrow EF =$

$D \rightarrow E$

$D \rightarrow F$

P-163

L-2

Q-1

$A \rightarrow C \times$

$A \rightarrow B \vee$

P-163

Q-05

$ABD \rightarrow AC$ $C \rightarrow BE$ $AD \rightarrow BF$ $B \rightarrow E$ ~~$A \rightarrow D \rightarrow AX$~~ ~~$A \rightarrow D \rightarrow C \checkmark$~~ ~~$C \rightarrow B \checkmark$~~ ~~$C \rightarrow E X$~~ ~~$AD \rightarrow B \checkmark$~~ ~~$AD \rightarrow F \checkmark$~~ ~~$B \rightarrow E \checkmark$~~ ~~$AD \rightarrow C \& F$~~ ~~$C \rightarrow B$~~ ~~$B \rightarrow E$~~ ~~$AD \rightarrow CF$~~ ~~$C \rightarrow B$~~ ~~$B \rightarrow E$~~ $\left. \begin{array}{l} \\ \\ \end{array} \right\}$ Canonical cover. $AD \rightarrow C$ $AD \rightarrow F$ $C \rightarrow B$ $B \rightarrow E$ $\left. \begin{array}{l} \\ \\ \end{array} \right\}$ minimal cov $A \rightarrow BC$ ~~$A \rightarrow E \rightarrow G \& H$~~ ~~$C \rightarrow D$~~ ~~$D \rightarrow G$~~ ~~$E \rightarrow F$~~

$$\left. \begin{array}{l} A \rightarrow C \\ A \rightarrow B \\ AE \rightarrow H \\ C \rightarrow D \\ D \rightarrow G \\ E \rightarrow F \end{array} \right\}$$

minimal cover

 ~~$BCD \rightarrow A$~~ $BC \rightarrow E$ $A \rightarrow F$ ~~$F \rightarrow D$~~ $F \rightarrow G$ $C \rightarrow D$ $A \rightarrow G X$

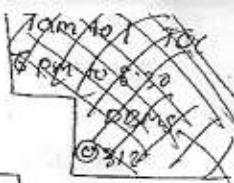
$$\left. \begin{array}{l} BC \rightarrow A \\ BC \rightarrow E \\ A \rightarrow F \\ F \rightarrow G \\ C \rightarrow D \\ A \rightarrow G \end{array} \right\}$$

minimal cover.

~~exists~~~~uniques~~

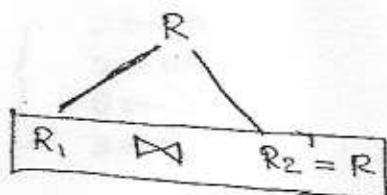
Properties of decomposition

52



06-12-13

① Loss-less Join decomposition

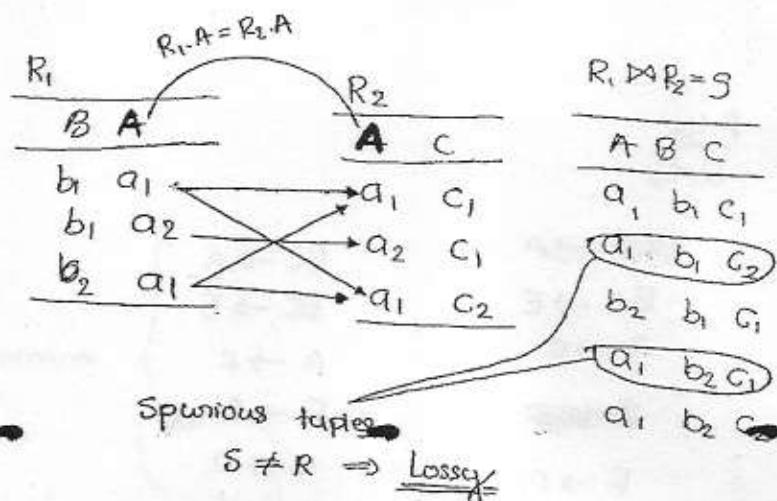


Let R be a relationship schema and F be a set of F.P's over R , the decomposition of R into R_1 and R_2 is said to be lossless decomposition w.r.t F iff $R_1 \bowtie R_2 = R$

* \bowtie - natural join

Ex:-

$R:$
A B C
$a_1 b_1 c_1$
$a_2 b_1 c_1$
$a_1 b_2 c_2$



R_1	R_2	$R_1 \bowtie R_2 = S$
A B	B C	A B C
$a_1 b_1$	$b_1 c_1$	$a_1 b_1 c_1$
$a_2 b_1$	$b_2 c_1$	$a_2 b_1 c_1$
$a_1 b_2$	$b_2 c_2$	$a_1 b_2 c_2$

$S = R \Rightarrow$ Lossless

if

$$R_1 \cap R_2 \rightarrow R_2 - R_1 \text{ (OR) } R_1 \cap R_2 \rightarrow R_1 - R_2$$

Note: The decomposition of R into R_1 and R_2 is said to be lossless if the attributes that is common in R_1 and R_2 is a key in either of the relations. 53

Let R be a relation and F be a set of functional dependencies over R . The decomposition of R into R_1 and R_2 is lossless iff f^+ contains either the functional dependency

$$R_1 \cap R_2 \rightarrow R_2 - R_1$$

(or)

$$R_1 \cap R_2 \rightarrow R_1 - R_2$$

Ques Consider a Relation $R(ABC)$ with FD $A \rightarrow B$ is decomposed into $R_1(AB)$ & $R_2(BC)$ check whether the decomposition is lossy or lossless.

Ans

$$R_1 \cap R_2 \rightarrow R_1 - R_2 \Rightarrow B \rightarrow A \times$$

$$\{AB\} \cap \{BC\} = B$$

$$R_1 \cap R_2 \rightarrow R_2 - R_1 \Rightarrow B \rightarrow C \times$$

$$R_1 - R_2 = \{AB\} - \{BC\} = A$$

\therefore the decomposition is lossy

$$R_2 - R_1 = \{BC\} - \{A\} = C$$

Ques $R(ABC)$

$f: \{A \rightarrow B\}$ decomposed into $R_1(AB)$, $R_2(AC)$

Sol:-

$$R_1 \cap R_2 \rightarrow R_1 - R_2$$

$$\{AB\} \cap \{AC\} \rightarrow \{AB\} - \{AC\}$$

$$A \rightarrow B \Rightarrow \underline{\text{losskey}}$$

Ques $R(ABCD)$

$F: \{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$ decomposed into

$R_1(AB) R_2(BC) R_3(CD)$

the decomposition is lossless or lossy?

Ans

$(R_1 \bowtie R_2) \rightarrow B$ key in R_2

$((R_1 \bowtie R_2) \bowtie R_3) \rightarrow C$ is key in R_3

}

loss less

Ques

$R(ABCDE)$

$F: \{A \rightarrow B, C \rightarrow DE, AC \rightarrow F\}$

$R_1(AB) R_2(CDE) R_3(ACF)$

the decomposition is _____



$$\underbrace{[R_1 \bowtie (R_2 \bowtie R_3)]}_{} = R$$

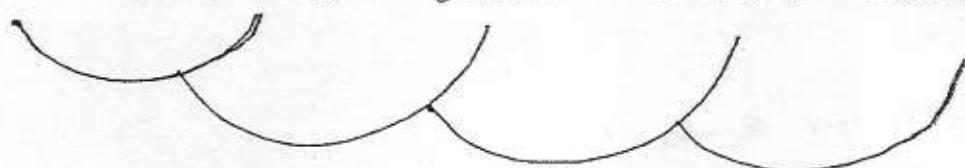
loss less

Ques

$R(ABCDEFGHIJ)$

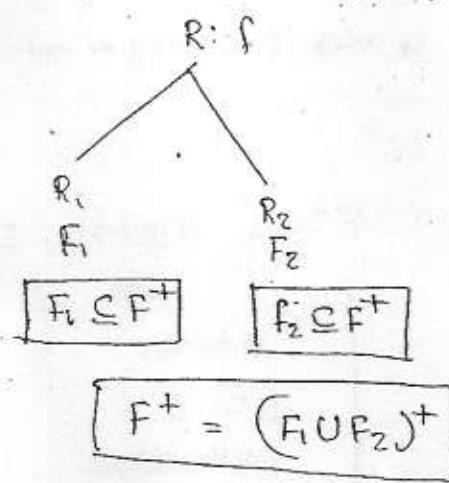
$F: \{AB \rightarrow C, A \rightarrow DE, B \rightarrow F, F \rightarrow GH, D \rightarrow IJ\}$

$R_1(ABC) R_2(ADE) R_3(BF) R_4(CFGH) R_5(DIJ)$



$$\underbrace{[[[(R_1 \bowtie R_2) \bowtie R_3] \bowtie R_4] \bowtie R_5] = R}$$

Loss less decomposition.



The decomposition of a relation R with F.D's ' F ' into R_1 and R_2 with F.D's F_1 and F_2 respectively is said to be dependency preserving iff $F^+ = (F_1 \cup F_2)^+$

Ex:- $R(ABC)$, $f:D \{ A \rightarrow B, B \rightarrow C, C \rightarrow A \}$ is decomposed into $R_1(CAB) \leftarrow R_2(BC)$ is the decomposition is dependency preserving or not?

Soln:-

$$\begin{array}{c} R_1 : F_1 \\ \hline A \rightarrow B \\ B \rightarrow A \end{array} \quad \begin{array}{c} R_2 : F_2 \\ \hline B \rightarrow C \\ C \rightarrow B \end{array}$$

$$\begin{aligned} (F_1 \cup F_2) &= \{ A \rightarrow B, B \rightarrow C, C \rightarrow B, B \rightarrow A \} \\ (F_1 \cup F_2)^+ &= \{ A \rightarrow B, B \rightarrow C, C \rightarrow A, \dots \} \\ \Rightarrow (F_1 \cup F_2)^+ &= F \end{aligned}$$

\Rightarrow dependency preserving decomposition

$$\begin{aligned} F^+ &\left\{ \begin{array}{l} A^+ = ABC \{ A \rightarrow A, A \rightarrow B, A \rightarrow C \} \\ B^+ = BCA \{ B \rightarrow A, B \rightarrow B, B \rightarrow C \} \\ C^+ = CAB \{ C \rightarrow A, C \rightarrow B, C \rightarrow C \} \end{array} \right. \end{aligned}$$

Ques RCABCD with F.D F: { AB → C, D → A }

56

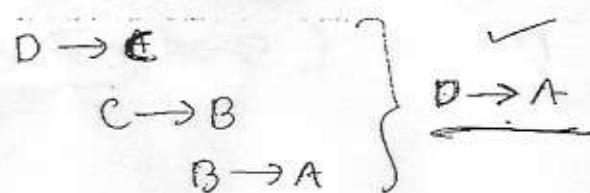
R ₁ (AP)	R ₂ (BCD)	↳ Decomp. is dep. preserving or not
F ₁ : D → A	F ₂ : DB → C DB → C	$F^+ = \{ AB \rightarrow C, D \rightarrow A \}$ $A^+ = A$ $B^+ = B$ $AB^+ = A, B, C$ $D^+ = D, A$ $DB^+ = D, B, A, C$ $C^+ = C$ $BC^+ = B, C$ $CD^+ = C, D, A$

(F₁ ∪ F₂) { D → A, DB → C }
 Compute AB⁺ = A, B from (F₁ ∪ F₂)
 AB → C is lost
 hence not preserving dependency.

Ques RCABCD

F: { A → B, B → C, C → D, D → A }

R ₁ (AB)	R ₂ (BC)	R ₃ (CD)
A → B ✓	B → C ✓	C → D ✓
B → A	C → B	D → C



⇒ Dependency is preserved

Q1 R(ABCDEG)

57

$f: \{AB \rightarrow G, AC \rightarrow B, AD \rightarrow E, B \rightarrow D, BC \rightarrow A, E \rightarrow G\}$

$R_1(ABC)$	$R_2(ABDE)$	$R_3(EG)$
$AB \rightarrow C$	$AD \rightarrow E$	
$AC \rightarrow B$	$B \rightarrow D$	
$BC \rightarrow A$		$E \rightarrow G$

$$(F_1 \cup F_2 \cup F_3)^+ = F^+$$

Dependancy is preserved

Q2 R(ABCDE)

$f: \{A \rightarrow BC, C \rightarrow DE, D \rightarrow E\}$

$$(R_1 \bowtie R_2) = R$$

$R_1(ABC)$	$R_2(DE)$
$A \rightarrow BC$	
$C \rightarrow D$	$D \rightarrow E$

I lossy ~~II~~ lossless

~~III~~ D.P ~~IV~~ not D.P

P-166

Q34

$$D.C.K = DB \quad \underline{D.P}$$

~~As~~ BC & AD is not a decomposition because its lossy

2) C.K : BC, AB

lossless bcs C is key in R1

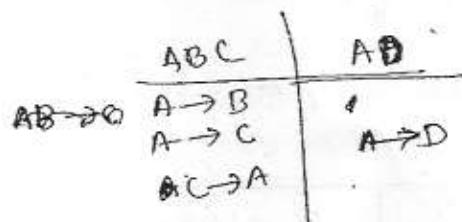
It is not preserving dependency ($AB \rightarrow C$ lost).

3)

) $A \rightarrow BC, C \rightarrow AD$

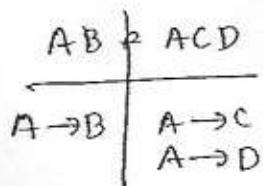
C.K : C, A

D.P. & loss less.



) $A \rightarrow B, B \rightarrow C, C \rightarrow D$

C.K : A



loss

loss less but Not D.P.

$B \rightarrow C$ is loss

) $A \rightarrow B, B \rightarrow C, C \rightarrow D$

AB, AD, CD

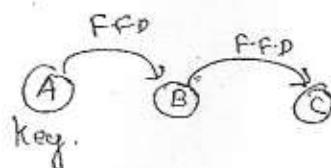
lossy ~~loss~~ & Not D.P.

$B \rightarrow C$

Q: R(ABC)

$$F: \{ A \rightarrow B, B \rightarrow C \}$$

normal form forms : 2NF



R is in 2NF and also in 1NF

Q: R(ABCD) with f: { AB → C, A → D } decompose the above relation into 2NF?

Ans

$$R_1(ABC) \text{ & } R_2(AD) -$$

C.K: AB

R is in 1NF but not in 2NF.

$$R_1(\overline{ABC}) - 2NF$$

$$R_2(AD) - 2NF$$

Q R(ABCDE)

$$F: \{ AB \rightarrow C, A \rightarrow D, B \rightarrow E \}$$

decompose the above relation

into 2NF.

$$\left\{ \begin{array}{l} R_1(ABC) - 2NF \\ R_2(AD) - 2NF \\ R_3(BE) - 2NF \end{array} \right\}$$

C.K: AB

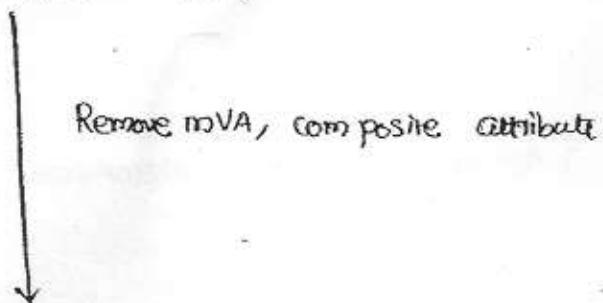
$$PFD = \frac{A \rightarrow D}{B \rightarrow E}$$

R is in 1NF but not in 2NF

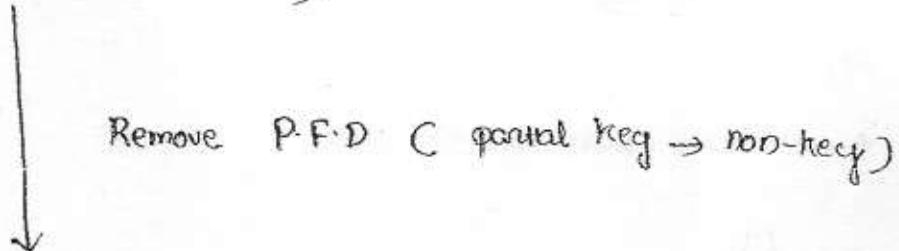
Note:- The decomposition required the closures of the violating dependencies into separate relations and remaining attributes (if any) and key attributes of decomposed relations forms another relation.

3NF

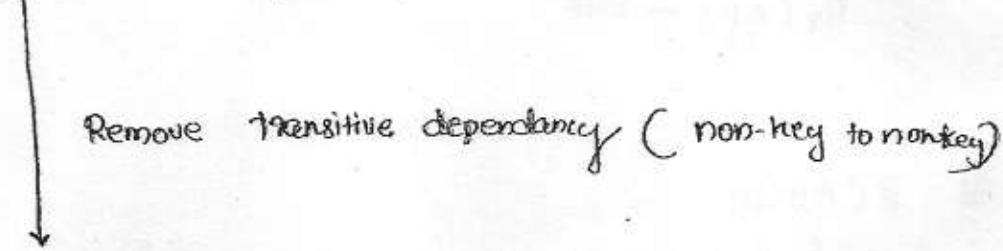
Un-normalized relation



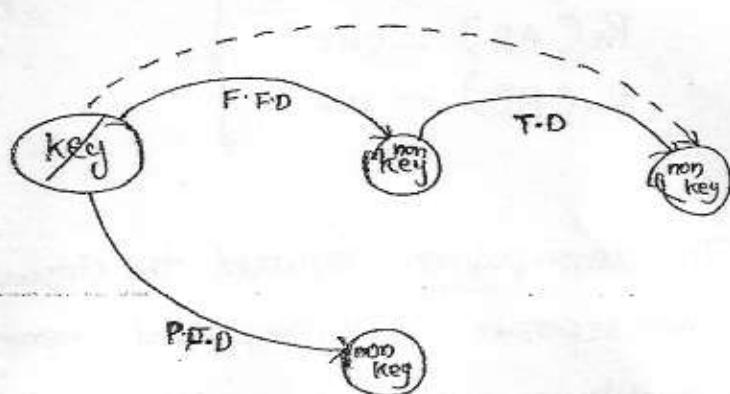
1NF (atomic values)



2NF (allows only F.F.D)



3NF



Normalization of data can be looked upon as a process of analyzing the given relation schema based on their F.D.'s and primary key's to achieve the desired properties of minimizing redundancy and minimizing the insertion deletion and update anomalies. Several normal forms have been proposed. Each normal form minimizes the redundancy up to some extent. The higher normal forms are only of theoretical interest but not practically applicable.

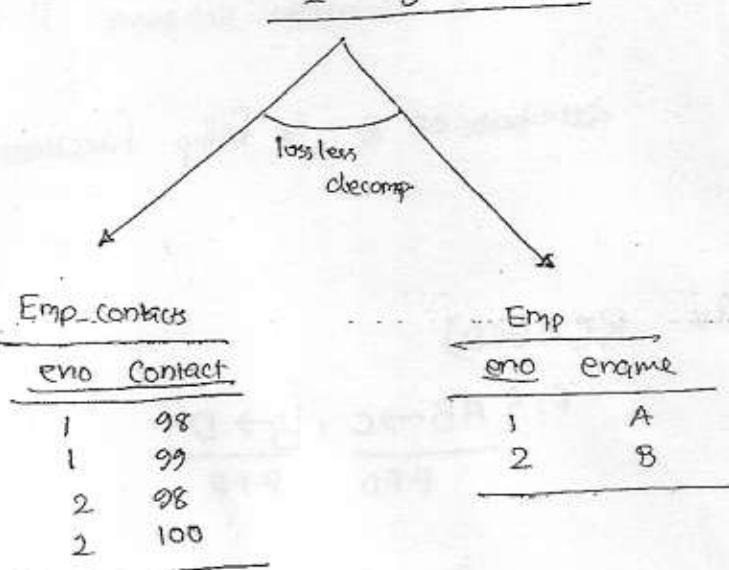
Most DB systems uses normalization upto 3NF.
(upto BCNF is recommended.)

1NF :-

Un-normalized relation

Employee		
eno	ename	Contact
1	A	{98, 99}
2	B	{98, 100}

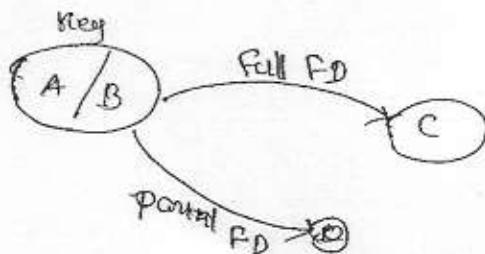
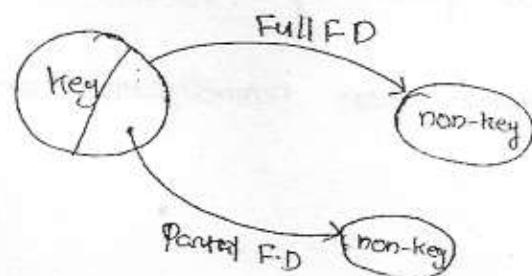
1NF		
eno	ename	Contact
1	A	98
1	A	99
2	B	98
2	B	100



A relation is in 1NF if every field contains only atomic values
 ie, The attribute of any tuple must be a single value or null value
 from its domain.

- Every relation in RDBMS must satisfy 1NF.

2NF :-



Ex: $R(ABCD)$
 $F: \{AB \rightarrow C, B \rightarrow D\}$

2NF is based on the concept of full Functional dependency and it disallows partial functional dependencies.

A Relation Schema R is in 2NF. if every non-key attribute of R is fully functionally dependant on the key of R .

Ex: $R(ABCD)$

$f: \{ \underbrace{AB \rightarrow C}_{\text{FFD}}, \underbrace{B \rightarrow D}_{\text{P.F.D}} \}$ C.K : AB

R is in 1NF but not in 2NF (because PFD: $B \rightarrow D$).

$R(ABCDEF\bar{GHIJ})$

- $AB \rightarrow C$ - PFD
- $BD \rightarrow EF$ - PFD
- $AD \rightarrow GH$ - PFD
- $A \rightarrow I$ - PFD
- $H \rightarrow J$ - TD

C.K: ABD

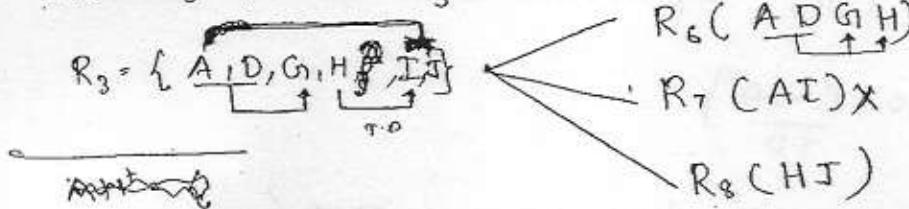
$$AB^+ = \{A, B, C, I, \}$$



$$BD^+ = \{B, D, E, F\}$$

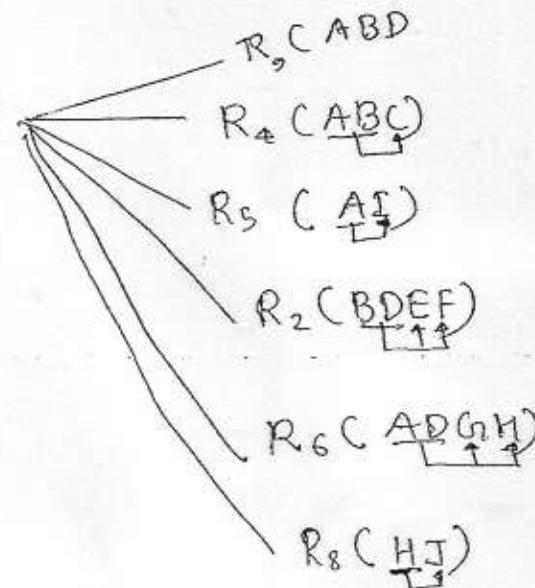
$$R_2(\underline{BDEF})$$

$$AD^+ = \{G, H, A, D, I, J\}$$



\Rightarrow 3NF tables

$R(ABCDEF\bar{GHI})$

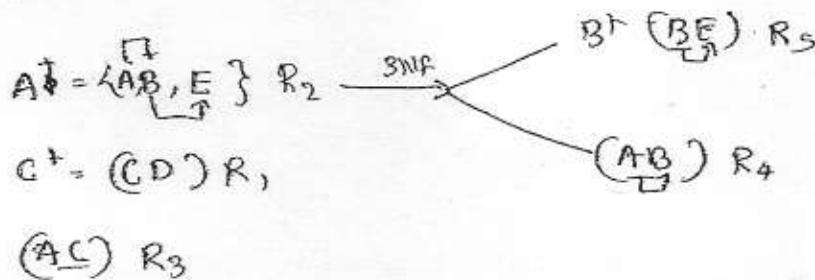


R(ABCDE)

C.K: AC

$f: \{A \rightarrow B, B \rightarrow E, C \rightarrow D\}$

R is in 1NF, not in 2NF



(ii) $R(A B C D)$ with primary key AB .

$$f: \{AB \rightarrow C, \quad \frac{B \rightarrow D}{P.F.D}\} \quad 1NF \text{ not in } 2NF$$

*) R(ABCD) with key AB and 2NF not in 3NF

$$f: \{AB \rightarrow C, C \xrightarrow{T,D} \}$$

(e) $R(ABCD)$ with key AB and 3NF

$$f: \{AB \rightarrow CD\}$$

3NF is designed to disallow transitive dependencies.
(OR)

A Relation schema R is in 3NF if it satisfies 2NF and no non-prime attribute of R is transitively dependent on key attributes of R.

Ex:- ① R(ABC)

$$f: \left\{ \begin{array}{l} A \rightarrow B, \\ \text{F.F.D} \end{array}, \begin{array}{l} B \rightarrow C, \\ \text{F.F.D} \end{array} \right\}$$

T.D

C.K: A

R is in 2NF but not in 3NF. because transitive dependency $B \rightarrow C$

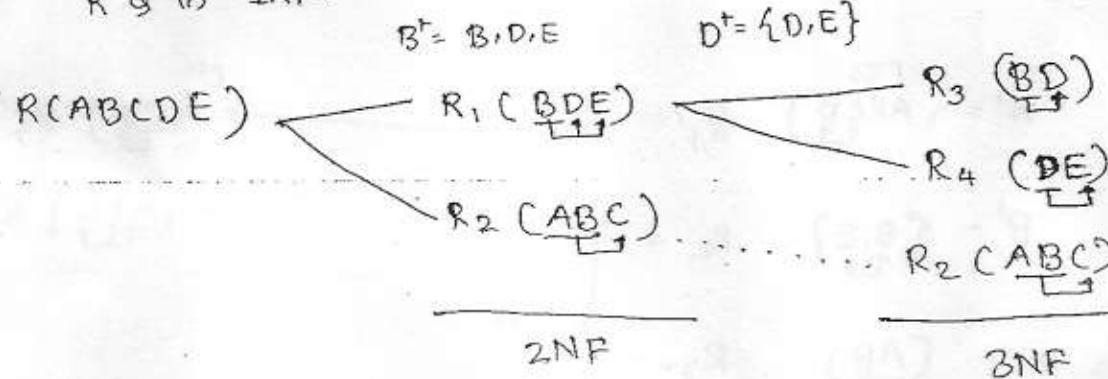
Decompose the above relation into 3NF

$$\begin{aligned} B^+ &= \{B, C\} & R_1 & \{B, C\} \\ A^+ &= \{A, B\} & R_2 & \{A, B\} \end{aligned} \quad \left. \begin{array}{c} \\ \\ \end{array} \right\} \quad \underline{\text{3NF}}$$

Ex: ② Decompose R(ABCDE), $f: \left\{ \begin{array}{l} AB \rightarrow C, \\ \text{F.F.D} \end{array}, \begin{array}{l} B \rightarrow D, \\ \text{P.F.D} \end{array}, \begin{array}{l} D \rightarrow E \end{array} \right\}$ into 3NF?

C.K: AB

R is in 1NF.



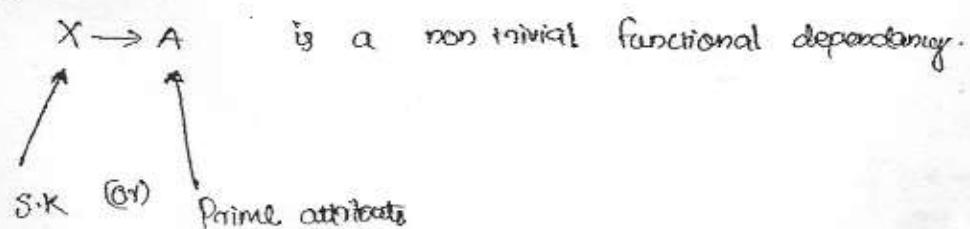
~~R(ABC)~~

$$f \models \{ \underbrace{AB \rightarrow C}_{\text{F.F.D}}, \underbrace{C \rightarrow A}_{\text{P.F.D}} \}$$

C.K: AB
CB

R is in 3NF and also in 2NF & 1NF

NF



Note:-

- A Relation Schema R is in 3NF if whenever a non-trivial functional dependency $X \rightarrow A$ holds in R then either X is a S.K of R or A is a prime attribute of R.

P-165
Q-25

B (B.T, A.N, B.Tg, L.I.P, A.g,

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

$$F: \{ \underbrace{A \rightarrow FC}_{\text{P.F.D}}, C \rightarrow D, \underbrace{B \rightarrow E}_{\text{P.F.D}} \}$$

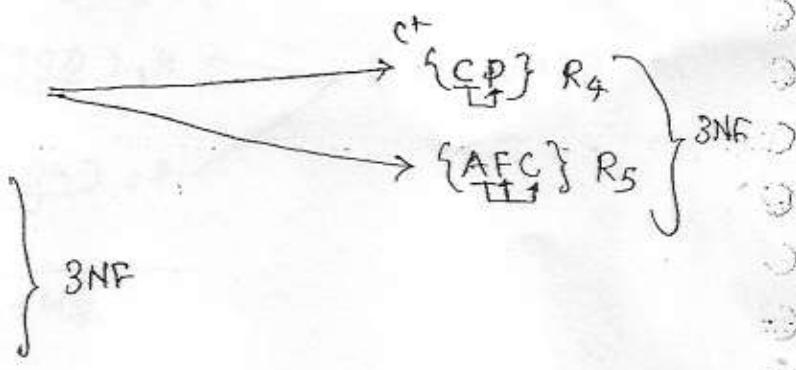
C.K: AB

R is in 1NF

$$A^+ = \{ \overbrace{AFCD}^T \} \quad R_1 \quad \checkmark$$

$$B^+ = \{ \overbrace{B,E}^T \} \quad R_2 \quad \checkmark$$

$$(AB) \quad R_3 \quad \checkmark$$



2NF

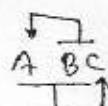
Boyce Codd Normal form (BCNF)

Ex:-

 $R(ABC)$

$$f: \{ AB \rightarrow c, c \rightarrow A \}$$

C.H : AB
CB



3NF but not in BCNF

 \overline{ABC}

overlapping candidate key

A Relationship Schema R is in BCNF if whenever a non-trivial FD $X \rightarrow A$ holds in R then X is a Superkey of R . i.e., the determinants of all functional dependencies must be superkeys.

Q2 $R(ABC)$

$$f: \{ A \rightarrow B, B \rightarrow c, c \rightarrow A \}$$

The relation is in BCNF \therefore every attribute is key

C.H: A, B, C

It is also in 3NF, 2NF, 1NF

Note: If every determinant is superkey of R , the relation is in BCNF

Q3 find normal form of a two attribute relation is _____

 $R(AB) \sim \text{BCNF}$

- i) $f: \{ A \rightarrow B \}$ C.H: A - BCNF
- ii) $f: \{ B \rightarrow A \}$ C.H: B - BCNF
- iii) $f: \{ A \rightarrow B, B \rightarrow A \}$ C.H: A, B - BCNF
- iv) $f: \{ \emptyset \}$ C.H: AB - BCNF

Q: $R(ABC) : f: \{\emptyset\}$

the Nor. form = BCNF

Note:- A Relation with only trivial dependancies is always in BCNF

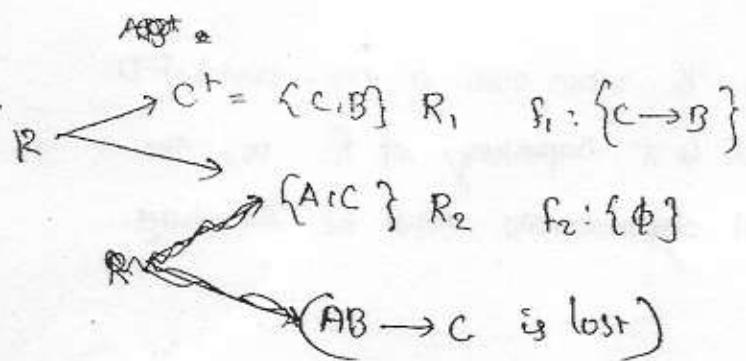
Q2

$R(ABC)$

$f: \left\{ \begin{array}{l} \underbrace{AB \rightarrow C}_{S.K}, C \rightarrow \underline{B} \\ \text{Prime} \end{array} \right\}$ Decompose the above Relation into BCNF.

CK : AB, AC

R is in 3NF but not in BCNF.



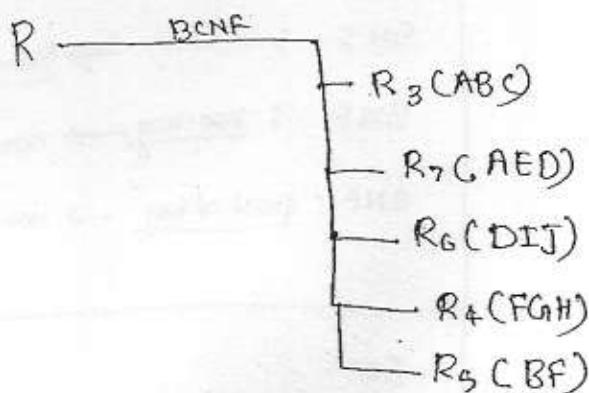
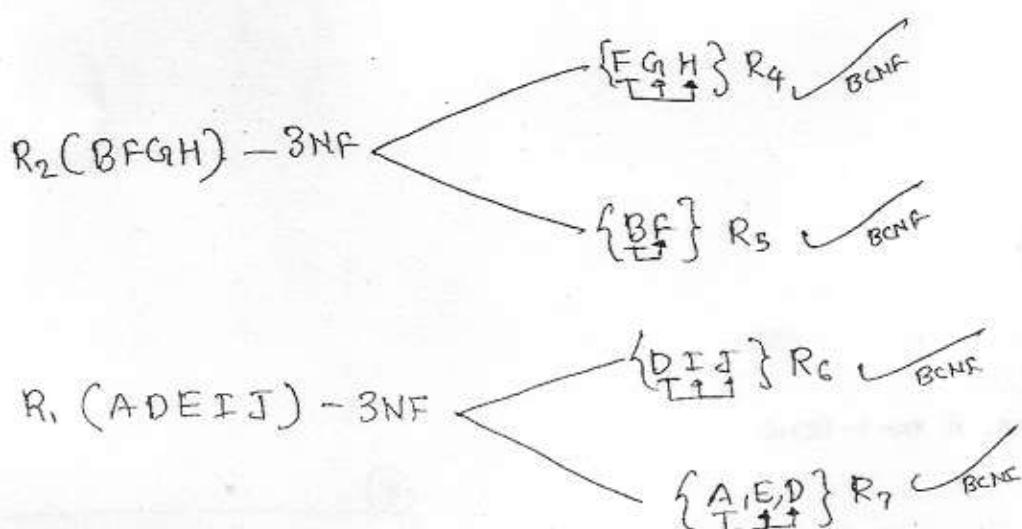
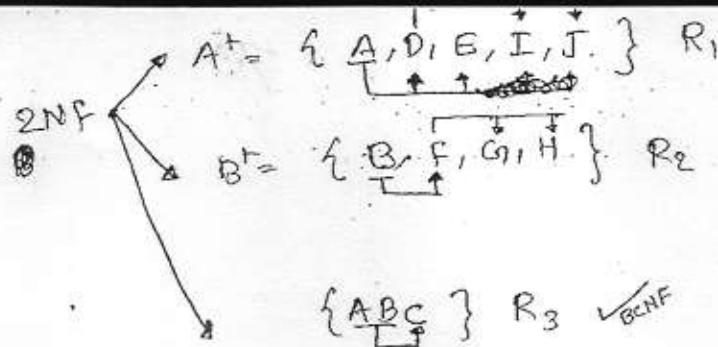
The BCNF decomposition may-not ensures dependency preserving decomposition.

Q3 $R(ABCDEFGHIJ)$

$f: \left\{ \begin{array}{l} AB \rightarrow C, \underbrace{A \rightarrow DE}_{P.F.D}, \underbrace{B \rightarrow F}_{P.F.D}, E \rightarrow GH, D \rightarrow IJ \end{array} \right\}$ decompose.

the above relation into BCNF.

CK AB



Q4 $R(ABCD)$ is a relation. Which of the following does not have a lossless join and D.P BCNF decomposition

- | | |
|---|---|
| a) $A \rightarrow B, B \rightarrow CD$
b) $A \rightarrow B, B \rightarrow C, C \rightarrow D$
c) $AB \rightarrow C, C \rightarrow AD$
d) $A \rightarrow BCD$ | a) $(AB) R_1, (BCD) R_2 \quad \checkmark$
b) $(AB) R_1, (BC) R_2, (CD) R_3 \quad \checkmark$
c) $(\overbrace{ACD}) R_1, (BC) R_2 \quad \times \text{ not D.P}$
d) $(ABCD) R_1 \quad \checkmark \rightarrow (CD) R_3, (CA) R_4$ |
|---|---|

R(ABCDE)

$$F: \left\{ \begin{array}{l} A \rightarrow B, BC \rightarrow E, ED \rightarrow A \\ \hline PA & PA & PA \end{array} \right\}$$

C.K: ACD

C.R: ECD

C.N: BCD

R is in 3NF

70

Q-30

R(ABCD)

(i) C.K - ?

(ii) N.F - ?

(iii) decompose if not in BCNF

$$\begin{array}{c} C \rightarrow D, C \rightarrow A, B \rightarrow C \\ \hline 2NF & 2NF & BCNF \end{array}$$

(i) C.K: B

(ii) 2NF

(iii)

$$f: \{B \rightarrow C, C \rightarrow A, C \rightarrow D\}$$

$$C^+ = \boxed{\{A, D\}} \quad R_1 \quad - \text{BCNF}$$

$$\boxed{\{B, C\}} \quad R_2 \quad - \text{BCNF}$$

~~$$f: \boxed{B \rightarrow C, D \rightarrow A}$$~~

(i) C.K: BD

(ii) BCNF

~~$$f: \{ABC \rightarrow D, D \rightarrow A\}$$~~

(i) C.K: ABC

DBC

(ii) 3NF

(iii)

*)

BCNF: Superkey \rightarrow Any3NF: \rightarrow Prime attributes2NF: non-key \rightarrow non-key1NF: part of key \rightarrow non-key

Ex:-

R(ABCDE) C.K: AB

$$f: \left\{ \begin{array}{l} AB \rightarrow C, B \rightarrow D, D \rightarrow E \\ \hline BCNF & 1NF & 2NF \end{array} \right\}$$

R is in 1NF

Ex:- R(ABCDE) C.K: E

$$f: \left\{ \begin{array}{l} A \rightarrow B, BC \rightarrow D, E \rightarrow AC \\ \hline 2NF & 2NF & BCNF \end{array} \right\}$$

R is in 2NF

$$2) \quad \frac{B \rightarrow C}{1NF}, \frac{D \rightarrow A}{1NF}$$

(i) C.K = BD

(ii) 1NF

(iii)

$$B^t = \{B, \underline{C}\} R_1 \checkmark_{BCNF}$$

$$D^t = \{D, \underline{A}\} R_2 \checkmark_{BCNF}$$

$$\{\underline{BD}\} R_3 \checkmark_{BCNF}$$

$$3) \quad \frac{ABC \rightarrow D}{BCNF}, \frac{D \rightarrow A}{3NF}$$

(i) ABC : CK

DBC : CK

(ii) 3NF

$$D^t = \{D, \underline{A}\} R_1 \checkmark_{BCNF} \Rightarrow \text{Lossy decomposition} \quad ABC \rightarrow D \text{ is lost}$$

$$\{\underline{DBC}\} R_2 \checkmark_{BCNF}$$

$$4) \quad \frac{A \rightarrow B}{BCNF}, \frac{BC \rightarrow D}{2NF}, \frac{A \rightarrow C}{BCNF}$$

(i) C.K : A

(ii). 2NF

$$(iii) \quad BC^t = \{B, \underline{C}, D\} R_1 \checkmark$$

$$\{\underline{ABC}\} R_2 \checkmark$$

Ques

Q-31* R(ABCDEFCH)

 $f_1: \{AB \rightarrow C\} \cap \{AB \rightarrow H\}$ ~~A~~ $\rightarrow G$ $F \rightarrow G$ $FB \rightarrow H$ $HBC \rightarrow A \cap \{F \rightarrow H\}$ $FBC \rightarrow ADE \}$ \Rightarrow minimal cover $AB \rightarrow CH$ $F \rightarrow G$ $FB \rightarrow H$ $HBC \rightarrow F$ $FBC \rightarrow ADE$ $AB^+ = ABCHFDE - ; \text{key}$ $F^+ = F \cdot G$ $FB^+ = F \cdot B \cdot H \cdot G$ $HBC^+ = HBCFGADE - ; \text{key}$ $FBC^+ = FBCADEGH - ; \text{key.}$

$$\frac{AB \rightarrow CH}{BCNF}, \frac{F \rightarrow G}{1NF}, \frac{FB \rightarrow H}{3NF}, \frac{HBC \rightarrow F}{BCNF}, \frac{FBC \rightarrow ADE}{BCNF}$$

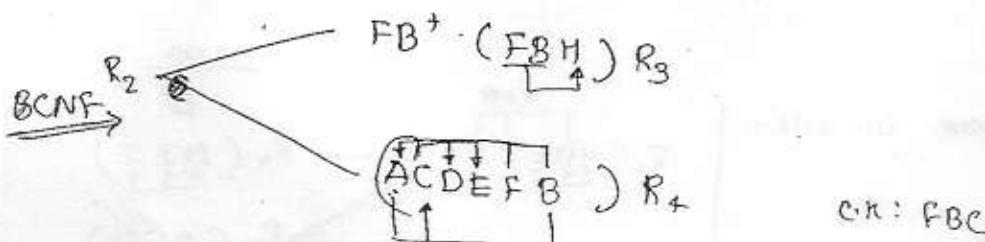
$2NF \Rightarrow$

$F^+ = (FG)R, \checkmark$

$\boxed{ABCDEFH}$

$$\begin{array}{c}
 \xrightarrow{\text{AB} \rightarrow \text{CH}} \text{BCNF} \\
 \xrightarrow{\text{FB} \rightarrow \text{H}} \text{3NF} \\
 \xrightarrow{\text{HBC} \rightarrow \text{F}} \text{BCNF} \\
 \xrightarrow{\text{FBC} \rightarrow \text{ADE}} \text{BCNF} \\
 \end{array}
 \quad \left. \right\}$$

GK: AB, HBC, FBC

 $\text{HBC} \rightarrow \text{F}$ lost $\text{AB} \rightarrow \text{H}$ lost $f_4: \{ \text{FBC} \rightarrow \text{ADE}, \text{AB} \rightarrow \text{C} \}$

(7-1) DBMS

P-165

Q4

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

$$F: \{ \begin{array}{l} AB \rightarrow C, A \rightarrow DE, B \rightarrow F, F \rightarrow GH, D \rightarrow IJ \\ \text{FFD} \quad \text{P-FD} \quad \text{P-FD} \quad \text{P-FD} \quad \text{T-D} \\ \text{BCNF} \quad \text{1NF} \quad \text{2NF} \quad \text{2NF} \quad \text{2NF} \end{array} \}$$

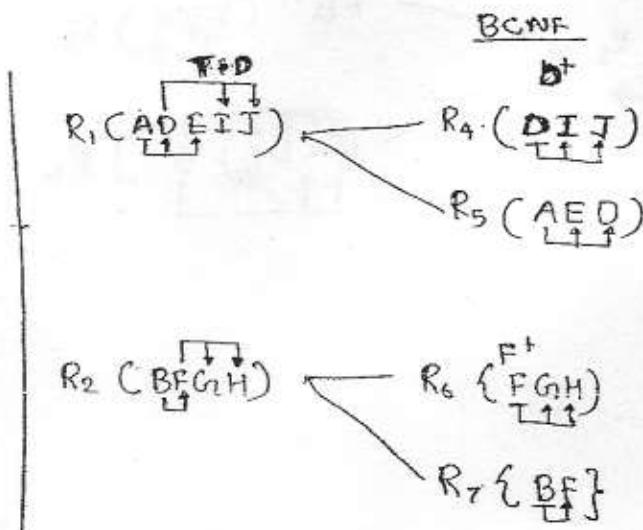
$$AB^+ = \{ A, B, C, D, E, F, G, H, I, J \} : C.K$$

R is in 1NF \therefore Decompose to 2NF

$$A^+ = \{ \begin{array}{l} A, D, E \\ T \uparrow \uparrow \end{array} \} \boxed{I, J} R_1 : 2NF$$

$$B^+ = \{ \begin{array}{l} B, F, G, H \\ T \uparrow \uparrow \end{array} \} R_2 : 2NF$$

$$\{ \begin{array}{l} AB \\ \hline C \end{array} \} R_3 : BCNF$$



If it is to decompose into

2NFR₁ (ADEIJ)R₂ (BFGH)R₃ (ABC)3NF

also in BCNF

R₃ (ABC)R₄ (DIJ)R₅ (AED)R₆ (FGH)R₇ (BF)

P-168

Q25

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

$$f: \{ \begin{array}{l} A \rightarrow FC, C \rightarrow D, B \rightarrow E \\ \text{P-FD} \quad \text{P-FD} \quad \text{P-FD} \end{array} \}$$

$$C.K = AB^+ = \{ A, B, C, F, D, E \}$$

2NF

$$A^+ = \{ \underline{\underline{A, F, C, D}} \} R_1 \text{ } \cancel{\text{2NF}}$$

$$\cancel{B^+ = \{ \underline{\underline{B, E}} \} R_2 \text{ } \checkmark \text{BCNF}}$$

$$\{ \underline{\underline{AB}} \} R_3 \text{ } \checkmark \text{BCNF}$$

R (2NF)

$$R_1(C, A F C D)$$

$$R_2(B E)$$

$$R_3(A B)$$

3NF

$$A^+ C^+ = \{ \underline{\underline{C, D}} \} R_4$$

$$\{ \underline{\underline{A F C}} \} R_5$$

R (3NF)

$$R_2(B E)$$

$$R_3(A B)$$

$$R_4(C D)$$

$$R_5(A F C)$$

Q-26

R(A B C D E)

PK: AC

$$f: \{ \begin{array}{l} B \rightarrow E, C \rightarrow D, A \rightarrow B \\ \text{P.F.D} \quad \text{P.F.D} \quad \text{P.F.D} \end{array} \}$$

R is in 1NF

2NF

$$A^+ = \{ \underline{\underline{A, B, E}} \} R_1 \text{ } \cancel{\text{2NF}}$$

3NF

$$B^+ = \{ \underline{\underline{B, E}} \} R_4$$

$$C^+ = \{ \underline{\underline{C, D}} \} R_2 \text{ } \checkmark \text{BCNF}$$

$$\{ \underline{\underline{A, B}} \} R_5$$

$$\{ \underline{\underline{A, C}} \} R_3 \text{ } \checkmark \text{BCNF}$$

R (2NF)

$$R_1(A B E) \quad R_2(C D) \quad R_3(A C)$$

R (3NF)

$$R_2(C D) \quad R_3(A) \rightarrow R_4(B E) \quad R_5(A)$$

Q-27

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

$$f: \{ \begin{array}{l} AB \rightarrow C, BD \rightarrow EF, AD \rightarrow GH, A \rightarrow I, H \rightarrow J \\ \text{P.F.D} \quad \text{P.F.D} \quad \text{P.F.D} \quad \text{P.F.D} \quad \text{T.D} \end{array} \}$$

$$C.K = ABD^+ = \{ A, B, C, E, F, G, H, I, J \}$$

$$AB^+ = \{ \underline{\underline{A, B, C, I}} \} R_1 \text{ } \checkmark \text{2NF}$$

$$R_1(A B C I) \xrightarrow{2NF} A^+ = \{ \underline{\underline{A, I}} \} R_4 \text{ } \checkmark \text{BCNF}$$

$$BD^+ = \{ \underline{\underline{B, D, F, E}} \} R_2 \text{ } \checkmark \text{BCNF}$$

$$\{ \underline{\underline{A, B, C}} \} R_5 \text{ } \checkmark \text{BCNF}$$

$$AD^+ = \{ \underline{\underline{A, D, G, H, I, J}} \} R_3 \text{ } \checkmark \text{2NF}$$

$$\{ \underline{\underline{A, B, B D A D}} \} = \{ \underline{\underline{A, B D}} \} R_0 \text{ } \checkmark \text{BCNF}$$

$$R_3(\underline{ADGHIJ}) \xrightarrow{3NF} H^+ = \{\underline{H}\} R_6 \quad \checkmark_{BCNF}$$

$$\{\underline{ADGHI}\} R_7 \quad \checkmark_{BCNF}$$

76

BCNF of R

$$R_0(\underline{CABD})$$

$$R_1(\underline{ADGHI})$$

$$R_6(\underline{CHI})$$

$$R_5(\underline{ABC})$$

$$R_4(\underline{AI})$$

$$R_2(\underline{BDFE})$$

Ex-29

$$R(ABCDE) \quad f: \{ \begin{array}{l} A \rightarrow B \\ BC \rightarrow E \\ ED \rightarrow A \end{array} \}$$

$$\text{Keep: } \left. \begin{array}{l} ACD \\ ECD \\ BCD \end{array} \right\} R \text{ is in 3NF}$$

Ex-30

$$(1) R(ABCD)$$

$$\begin{array}{c} C \rightarrow D, C \rightarrow A, B \rightarrow C \\ \xrightarrow[2NF]{FD} \quad \xrightarrow[2NF]{FD} \quad \xrightarrow[BCNF]{FD} \\ R: B \end{array} \quad 2NF \text{ but not in 3NF}$$

$$R(ABCD) \xrightarrow{3NF} \left. \begin{array}{l} C^+ \{ \underline{C}, \underline{D}, \underline{A} \} R_1 \\ \{ \underline{BC} \} R_2 \end{array} \right\} \quad \checkmark_{BCNF}$$

$$(2) \quad \begin{array}{c} B \rightarrow C, D \rightarrow A \\ \xrightarrow[2NF]{FD} \quad \xrightarrow[2NF]{FD} \\ R: BD \end{array} \quad R \text{ is in 1NF but not in 2NF}$$

$$R \xrightarrow{} \left. \begin{array}{l} B^+ \{ \underline{BC} \} R_1 \quad \checkmark_{BCNF} \\ \{ \underline{DA} \} R_2 \quad \checkmark_{BCNF} \\ \{ \underline{BD} \} R_3 \quad \checkmark_{BCNF} \end{array} \right.$$

R(ABCDEFCH) f: { AB } → C H

A → D

F → G

FB → H

HBC → A F

FBC → E

}

77

K. AB, HBC, FBC

f: { AB } → CH BCNF

A → D (PFD INC)

F → G (PFD INC)

FB → H (PFD INC)

HBC → AF BCNF

FBC → E BCNF

}

AB^r = { A, B, C, H, D, F, G, E }

HBC^r = { H, B, C, F, A, G, D, E }

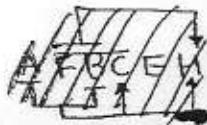
FBC^r = { F, B, C, E, H, A, G, D }

} keep

R is now in 1NF not in 2NF

→ A^r = { A, D } R₁ ✓ BCNF

→ F^r = { F, G } R₂ ✓ BCNF



FB^r = { FBH } R₃ ✓ BCNF

→ { A FBC E } R₄ ✓ BCNF

Preserved dep's

AB → C

FBC → E

A → D

F → G

FB → H

Not preserved dep's

AB → H

HBC → AF

(3) $\frac{ABC \rightarrow D, D \rightarrow A}{\text{FFD } 3NF \quad \text{P.P.D } 3NF}$

GK : ABC
: DBC

R is in 3NF but not in BCNF
BCNF

$R \xrightarrow{\cdot} D^+ = \{\underline{D/A}\}, R_1 \checkmark_{BCNF}$
 $\xrightarrow{\cdot} \{DBC\}, R_2 \checkmark_{BCNF}$

inexplicable decompositions

Not. D.P

$ABC \rightarrow D$ lost

(4) $\frac{A \rightarrow B}{\text{BNF}}, \frac{BC \rightarrow D}{T \cdot D \in 2NF}, \frac{A \rightarrow C}{\text{BCNF}}$

C.K : A

R is in 2NF but not in BCNF & 3NF

$BC^+ = \{\underline{BCD}\}, R_1 \checkmark_{BCNF}$

$\{\underline{ABC}\}, R_2 \checkmark_{BCNF}$

(5) $\frac{AB \rightarrow C}{\text{BCNF}}, \frac{AB \rightarrow D}{\text{BCNF}}, \frac{C \rightarrow A}{\text{P.P.D } 3NF}, \frac{D \rightarrow B}{\text{P.P.D } 3NF}$

GK = AB
CB
AD
CD

R is in 3NF but not in BCNF

$e^+ = \{\underline{CA}\}, R_1 \checkmark_{BCNF}$

$b^+ = \{\underline{D,B}\}, R_2 \checkmark_{BCNF}$

$\{\underline{CD}\}, R_3 \checkmark_{BCNF}$

Not D.P

$AB \rightarrow CD$ lost

R(A B C D E F G H)

f: {
 BC → A
 BC → E
 A → F
 F → G
 C → D
 A → G}

Key: BC^t = {B, C, E, D, A, F, G}

f: {
 BC → A ✓ - BCNF
 BC → E ✓ - BCNF
 A → F ✓ - 1NF (2NF)
 F → G ✓ - 1NF (2NF)
 C → D ✓ - P.F.D (1NF)
 A → G ✓ - 1NF (2NF)}

R is in 1NF but not in 2NF

1NF to 2NF

C^t = {C, D} R₁ ✓
 T ↑

{C A B E F G H} R₂ ✓
 T ↑ ↑
 2NF $\xrightarrow{3NF}$ A^t = {A F G H} R₃ $\xrightarrow{2NF}$

{A C B E} R₄ ✓
 T ↑
 BCNF

R₃ (A F G H) $\xrightarrow{(F G)} R_5$ ✓
 T ↑
 BCNF
 R₃ (A F G H) $\xrightarrow{(A F)} R_6$ ✓
 T ↑
 BCNF

D.P & lossless

Q: 33]

R(A B C D E F G H)

f: f A → BC
 AE → H

C → D

D → G

E → F

}

1NF $\xrightarrow{2NF}$

f: {
 A → BC (1NF)
 AE → H (BCNF)
 C → D (2NF)
 D → G (2NF)
 E → F (1NF)}

Key: AE^t = {A, B, C, D, E, F, G}

A^t = {
 A, B, C, D, G } R₁ ✓
 T ↑ L ↑

{A, E, F, A, D, G, H}

$E^+ = \{ E, F \} R_2 \quad \checkmark_{BCNF}$

80

$\{ A, E, H \} R_3 \quad \checkmark_{BCNF}$

2NF to 3NF

$R_1: \underbrace{ABC}_{\uparrow \uparrow} \underbrace{D, G, H}_{\uparrow \uparrow}$

$C^+ = \{ C, D, G \} R_4 \quad \checkmark_{2NF}$

$\{ ABC \} R_5 \quad \checkmark_{BCNF}$

$D^+ = \{ D, G \} R_6$

$\{ C, D \} R_7$

BCNF

Q: 35

$R(CABCD)$

$f: \underbrace{AB \rightarrow DE}_{BCNF}, \underbrace{A \rightarrow C}_{2NF}, \underbrace{D \rightarrow E}_{2NF}$

C.K: AB

R is in 1NF but not in 2NF

1NF to 2NF

$A^+ = \{ A, C \} R_1 \quad \checkmark_{BCNF}$

$(\underbrace{ABDE}_{\uparrow \uparrow}) R_2 \quad \checkmark_{2NF}$



2NF to 3NF

$D^+ = \{ D, E \} \quad \checkmark_{BCNF} R_3$

$\{ \underbrace{ABD}_{\uparrow \uparrow} \} R_4 \quad \checkmark_{BCNF}$

R is decomposed into $R_1(A, C)$ $R_2(D, E)$ $R_4(A, B, D)$ (BCNF)
form

Q: 36

$R(CABCD)$ $f: \underbrace{AB \rightarrow DE}_{BCNF}, \underbrace{A \rightarrow C}_{2NF}, \underbrace{C \rightarrow D}_{2NF}$

C.K: AB,

$A^+ = \{ A, C, D \} R_1 \quad \checkmark_{BCNF}$

$\{ \underbrace{ABE}_{\uparrow \uparrow} \} R_2 \quad \checkmark_{BCNF}$

$C^+ = \{ C, D \} R_3 \quad \checkmark_{BCNF}$

$\{ AC \} R_4 \quad \checkmark_{BCNF}$

R is decomposed to $R_2(ABE)$, $R_3(C, D)$, $R_4(AC)$ and it is ~~not~~ D.P.

Qn: A



$$\begin{array}{c}
 E^t = \{EBF \underset{\substack{\uparrow \\ \downarrow}}{CD}\} \xrightarrow[2NF]{\quad} B^t = \{ \underset{\substack{\uparrow \\ \downarrow}}{BCD} \} \xrightarrow[3NF]{\quad} \{ \underset{\substack{\uparrow \\ \downarrow}}{CD} \} R \\
 \text{R} \leftarrow \{ \underset{\substack{\uparrow \\ \downarrow}}{EBF} \} R_3 \xrightarrow[3NF]{\quad} \{ \underset{\substack{\uparrow \\ \downarrow}}{BC} \} R \\
 \{ \underset{\substack{\uparrow \\ \downarrow}}{AE} \} R_4 \xrightarrow[BCNF]{\quad}
 \end{array}$$

Q-38

R{ABCDE}

$$f: \{ \underset{\substack{\uparrow \\ \downarrow}}{AB} \rightarrow \underset{\substack{\uparrow \\ \downarrow}}{CD} \} \xrightarrow[BCNF]{\quad}, \underset{\substack{\uparrow \\ \downarrow}}{C} \rightarrow A, \underset{\substack{\uparrow \\ \downarrow}}{D} \rightarrow E \}$$

$$C.K = AB \\ CB$$

$$D^t = \{ \underset{\substack{\uparrow \\ \downarrow}}{DE} \} R_1 \xrightarrow[BCNF]{\quad}$$

$$(\underset{\substack{\uparrow \\ \downarrow}}{AB} \underset{\substack{\uparrow \\ \downarrow}}{CD}) R_2 \xrightarrow[2NF]{\quad} C^t = \{ \underset{\substack{\uparrow \\ \downarrow}}{CA} \} R_3 \xrightarrow[BCNF]{\quad}$$

$$(AB \rightarrow CD \text{ is lost})$$

$$(\underset{\substack{\uparrow \\ \downarrow}}{CB}) R_4 \xrightarrow[BCNF]{\quad}$$

Q-39

R = CAB(D)

$$C.K: A$$

$$f: \{ \underset{\substack{\uparrow \\ \downarrow}}{A} \rightarrow B, \underset{\substack{\uparrow \\ \downarrow}}{B} \rightarrow C, \underset{\substack{\uparrow \\ \downarrow}}{C} \rightarrow D \}$$

$$B^t = \{ \underset{\substack{\uparrow \\ \downarrow}}{B} \underset{\substack{\uparrow \\ \downarrow}}{CD} \} \xrightarrow[3NF]{\quad} C^t = \{ \underset{\substack{\uparrow \\ \downarrow}}{CD} \} R_1 \xrightarrow[BCNF]{\quad}$$

$$\{ \underset{\substack{\uparrow \\ \downarrow}}{BC} \} R_2 \xrightarrow[BCNF]{\quad}$$

$$\{ \underset{\substack{\uparrow \\ \downarrow}}{AB} \} R_3 \xrightarrow[BCNF]{\quad}$$

Q-40

R{ABC(D)}

$$C.K: AB \\ CB$$

$$f: \{ AB \rightarrow CD, \underset{\substack{\uparrow \\ \downarrow}}{C} \rightarrow A, \underset{\substack{\uparrow \\ \downarrow}}{A} \rightarrow C \}$$

$$C^t = \{ \underset{\substack{\uparrow \\ \downarrow}}{CA} \} R_1 \xrightarrow[BCNF]{\quad} f_1: \{ \underset{\substack{\uparrow \\ \downarrow}}{C} \rightarrow A, \underset{\substack{\uparrow \\ \downarrow}}{A} \rightarrow C \}$$

$$\{ \underset{\substack{\uparrow \\ \downarrow}}{ABD} \} R_2 \xrightarrow[BCNF]{\quad} f_2: \{ AB \rightarrow D \}$$

$$f_1: \{ \underset{\substack{\uparrow \\ \downarrow}}{A} \}$$

$$(ABD) \cap (CBD)$$

Database language

— Data Definition language (DDL) :- "Structure" :- Create, Alter, Drop, ...
 SQL Examples.

— Data manipulation language (DML) :- "Data" :- Insert, Select, Update, Delete, ...

Basic queries

- > Create table Student (Rno number (2), name char(10)),
- > insert into Student Values (1, 'Ram');
- > Update Student Set Name = 'Rahul' where Rno=1;
- > Delete Student where rno=1;

Query Evaluation Process

1) from (cartesian product of all tables)

A	B
P Q	R S
1 2	5 6
3 4	7 8

2) Where (selects the tuples according to the 'where' condition)

AxB	
P Q R S	
1 2	5 6
1 2	7 8
3 4	5 6
3 4	7 8

$\left. \begin{matrix} 1 & 2 & 5 & 6 \\ 1 & 2 & 7 & 8 \\ 3 & 4 & 5 & 6 \\ 3 & 4 & 7 & 8 \end{matrix} \right\} 4$

$(m \times n)$

3) Group by (Divides the rows into groups)

4) Having (selects the group)

5) Expressions in select clause are evaluated (if any)

6) Distinct (Duplicates are eliminated)

7) Set operations (union, intersect, ...)

7) Set Operations (Union, intersect, except)

83

8) Order by (Sorts the rows of result)

Simple Select

> Select rno from Student;

> Select rno, name from Student;

> Select * from Student;

> Select rno, marks+5 from Student;

> Select distinct branch from Student;

> Sele

Select - Where

- and
- or
- not
- in
- not in
- between .. and
- not between .. and
- is null
- is not null
- like %
- like _

• <, >, <=, >=, <>

↑
not eq

Student (eno, name, branch, email, marks, city, percpnt);

1) Find Students of CSE living in Hyderabad (HYD)

Select * from student where branch = 'CSE' and City = 'HYD';

(OR)

> Select *

from student

where branch = 'CSE' and City = 'Hyd';

2) Find all Students of CSE and IT

> Select *

from student

where branch = 'CSE' OR branch = 'IT' ;

3) Find all Students ~~of~~ except of CSE.

> Select *

from student

~~where branch != 'CSE'~~

where not branch = 'CSE';

(OR)

branch != 'CSE';

(OR)

branch <> 'CSE';

4) find all students who are living in 'Hyd', 'Ban' & 'Chennai'

85

> Select *
from student

where city IN ('Hyd', 'Ban', 'Chennai');

(Same as):-

City = 'Hyd' or City = 'Bang' or City = 'Chenn'
Dis-adv:- * long list

Note:-

The IN operator is used to compare a value or column with a list of values.

5) find all students who scored the marks from 10 to 20

> Select *
from student

where marks between 10 AND 20

(Same as):-

inclusive
 $\text{marks} \leq 10$ and $\text{marks} < 20$; exclusive

Note:-

The between..and operator is used to compare a column with range of values.

find all students who have no passport

86

? Select *

from student

where passport is null;

NULL is

- not zero
- Value does not exist
- Value is unknown
- Even if known, but Value not specified.

Ans R - 10 tuples

■ Select *

from R

where I = 1;

{ 10 rows returned.

■ Select *

from R

where null is null

■ Select *

from R

where O is null

{ 0 rows returned

■ Select *

from R

where I > 2

> Update student

Set marks = marks + 5;

Select * from student;

	(before) marks	(after) marks
1	A 10	A 15
2	B	B
3	C 9	C 14

Q. Find all students whose name starts with 's'.

87

Select *
from student

where name like 's%';

→ Starts with S

'%A';
→ ends with A

'%.I.%';
→ containing I

'____';
→ All 3-length name

'S__';
→ length 3 & starts with S

'S__%';
→ starts with S and minimum length 3.

Note:-

The 'Like' condition is used to specify certain search condition for a pattern in a column.

A percentile (%) sign can be used to define wildcards (missing char.) both before and after the pattern. % sign can be used to replace an arbitrary number of zero and more characters.

Underscore (_) replaces a single character.

Order-by

Order-by clause is used to sort the rows.

* Company (name, invoice_no)

Display all the company in alphabetical order of their names.

> Select *
from company

Order by name asc;

Display all the company in reverse alphabetical order of their names.

> Select *
from Company

Order by name desc;

> Display all the companies in reverse alphabetical order of their names
and numerical order of their invoice no.

I/p	name,	invoice_no	qty
A	10	100	
B	7	101	
A	10	102	
C	9	103	
A	2	104	

> Select *

from Company

Order by name desc, invoice_no asc;

%p	name	invoice_no	qty
C	9	103	
B	7	101	
A	2	104	
A	10	100	
A	10	102	

Note:- When first order by clause fails it sorts the rows using
2nd order by clause when 2nd fails it sorts the rows by 3rd order
by clause and continues. If all order by clause fails it display

Aggregate functions

89

Avg

min

max

sum

Count

- > Select sum(marks), avg(marks)
from student;

%		Sum (marks)	avg (marks)	marks
		30	10	5
				10
				15

(OR) aliasing

- > Select sum(marks) as Total, avg(marks) as average
from student;

O/p. Total Average
 30 10

Q3 find a query to find diff between max. and min marks of the student

- > Select (max(marks) - min(marks)) as difference.
from student

O/p Difference
 10

b3 find the no. of students in a class.

> Select Count (*) as Strength
from student

O/p Strength

3

Student

Rno	name	m ₁	m ₂
1	A	1	2
2	B	2	4
3	C	4	6
4	D	4	8
5	E	4	null
6	F	null	6

> select max(name) from student;

% : F

Note:- min and max functions can be used over ~~textfields~~
numbers, strings, dates

> Select avg(name) from student; X
Error:

Note :- Avg and sum functions can only be used with numbers.

> select count(*) from Student;

O/p: 6 returning no. of rows in table.

> select count(m₁) from Student;

O/p: 5 returning no. of non-null values

> select count(4) from Student;

6

→ Count (constant) ; - returns no. of rows

> select count('Ramesh') from Student;

6

> select count(m₁+m₂) from Student;

4

> select count(distinct m₁) from Student;

3

Note:- with count function we can use any kind of data.

> select sum(m₁,m₂) from Student; X

Error

> select sum(m₁+m₂) from Student;

O/p: 31

> Select name

from student

where $m_2 = \max(m_2);$

Note:- Aggregate functions can not be used in where clause.

Error

> Select name, $\max(m_2)$

from student

~~where m2~~

Error

Name $\max(m_2)$

{ A,B,C,D,E,P } 8

Violates 1NF

> Select name

from student

order by $\max(m_2);$

Note:-

Order by - expects a column name

Error

> Select name

from student

order by $m_2;$

where $m_2 >= 4;$

order by Error

Should be none.

Q:- Consider a table T with following tuples

T	
Rno	marks
1	10
2	20
3	30
4	null

The following sequence of SQL statements was successfully executed on table T.

- > update T set marks = marks + 5;
- > select avg(marks) from ~~student~~ T;

What is the output of the select ~~statement~~ T?

- a) 18.75 b) 20 c) 25 d) Error

Group by - clause and Having - clause

Group by clause is used to compute aggregate functions on a group.

- > select count(*), branch
from student
where branch = 'CSE';

= 'IT';
= 'ECE';
- > select count(*), branch
from student
group by branch;

Note:- we can reduce the burden on the query execution engine by using groupby instead of by 1

Rno.	Name	Branch	Year	Gender
1	A	CSE	I	M
2	B	CSE	II	F
3	C	IT	I	F
4	D	IT	I	F
5	E	CSE	II	
6	F	ME	I	F

Q1 Write a query to find number of students in each branch?

> select Branch, Count(*)

from student

group by Branch

O/P: Branch Count (*)

Branch	Count (*)
CSE	3
IT	2
ME	1

Q2 find the no. of students in each branch and year?

> select Branch, Year, count(*)

from student

group by Branch, Year;

Branch	Year	Count (*)
CSE	I	1
CSE	II	2
IT	I	2
ME	I	1

- ⑥ If the "year" is not included in the group by clause
group by Branch;

95

<u>O/P</u>	<u>Branch</u>	<u>Year</u>	<u>Count (*)</u>
CSE	{I, II}	3	Violates INF, Error.

Note :-

All the columns that appear in the select clause must appear in the group by clause

6

Find no. of female Students in each branch. ?

<u>%P</u>	<u>Branch</u>	<u>Count (#)</u>
	CSE	1
	IT	2
	ME	1

Q4 find no. of female students in each branch and display the result if there are more than one student in the branch.

(P.T.O.)

> Select Branch, count(*)
 from Student
 where gender = 'F'
 group by Branch
 having count(*) > 1;

(execution order)

96

⑤

①

②

③

④

%		Branch	Count (*)
		IT	2

Ques:-

Having clause is used to write group conditions

Aggregate functions can be used in having class.

Find the no. of students in each branch except of CSE

> Select Branch, count(*) ④

from Student ①

where branch <> 'CSE' ②

group by branch ; ③

%		Branch	Count (*)
		IT	2
		ME	1

Q2 > Select Branch, count(*) ... ④
 from student ... ①
 group by branch ... ②
 having branch <> 'CSE' ; ... ③

%p	Branch	Count(*)
	IT	2
	ME	1

Q₁ is more efficient than Q₂.

Note:-

The column's that appear in having clause must be an aggregate function or must be part of a group by clause.

i.e., The column appearing in having clause must be a single value per the group.

Where Vs Having

⇒ 'where' is used to select the rows whereas 'having' is used to select the group

⇒ Aggregate functions cannot be used in where clause, but can be used in having clause.

Q. Which of the following statements are true about an SQL query.

P: An SQL query can contain a having clause even if it does not have a group by clause.

Q: An SQL query can contain a having clause only if it has a group by clause.

R: All the attributes used in the group by clause must appear in the select clause.

S: Not all attributes used in the group by clause must appear in the select clause.

a) P and R

~~b)~~ P and S

c) Q and R

d) Q and S

Ans

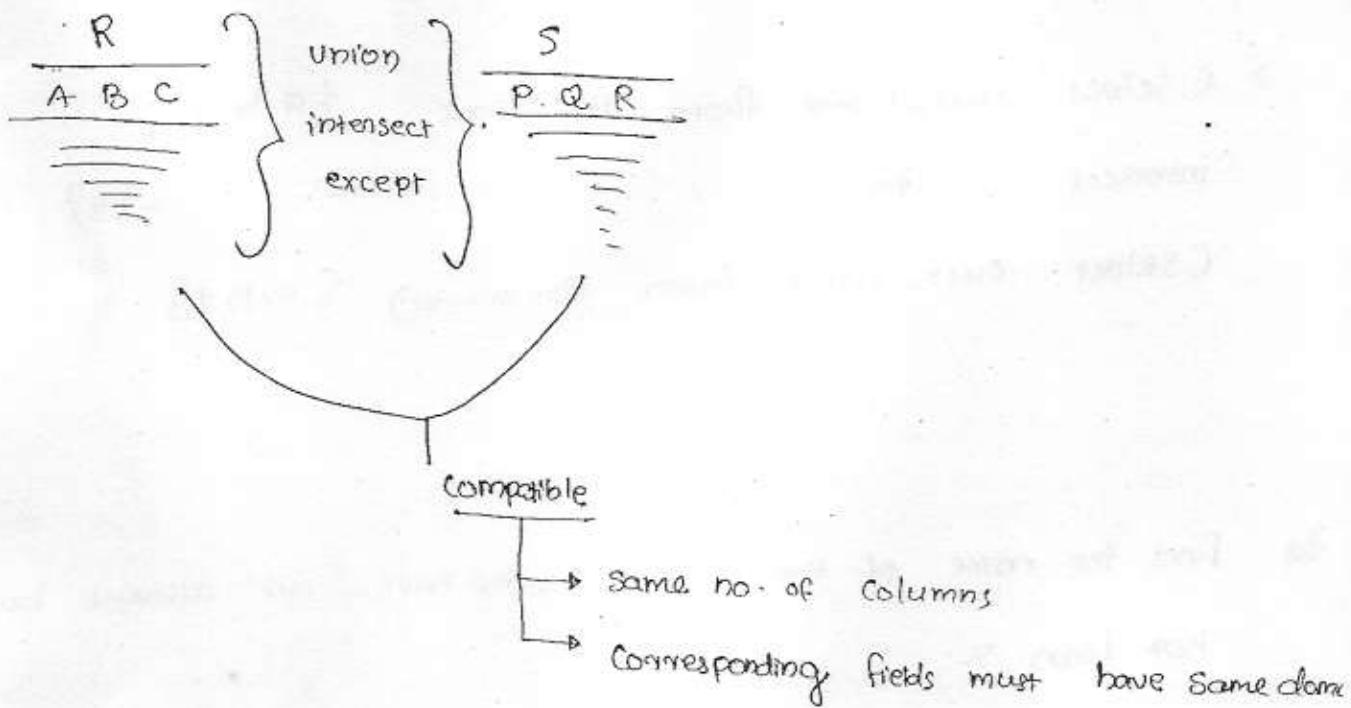
```
> Select count(*)
  from Student
  having count(*) > 2
```

Op
6

Note: In the absence of group by clause, the whole relation will be considered as one group.

Set manipulation operations

99



Since the answer to a query is multi set of rows it is natural to consider the set operations between two compatible relations ie, both must have same set of columns. and corresponding columns taken in order from left to right must have same domains.

SQL offers set manipulation under the names 'Union', 'Intersect' and 'Except'.

Ex:-

Depositor (acc-no, cust-name)

Borrower (loan-no, cust-name)

Q. Write a query to find name of the customers who have an account or loan or both at the bank?

> (Select cust-name from Depositor) {a,b,c}
 union = {a,b,c,p,q}

C Select cust-name from Borrower) {q,p,r}

100

Ques find name of the customer who have both an account and loan ?

> (Select cust-name from Depositor) {a,b,c}

intersect
= {a}

(Select cust-name from Borrower) {a,p,q}

Ques find the name of the customer who have an account but not loan. ?

> (Select cust-name from Depositor) {a,b,c}

except
{b,c}

(Select cust-name from Borrower) {p,q}

JOIN

The join operation is used to combine related tuples from two relations into a single ~~table~~ tuple.

Employee (eno, ename, city, deptno);

Dept (dno, dname);

Ques find name of the employee's working in research department.

> Select ename

from Employee, Dept

where depno = dno and dname = 'R';

O/P: ename

A
C

Employee		Dept	
ename	Deptno	Dno	dname
A	99	99	R
B	100	100	S
C	99		

Employee X Dept

A	99	—	99	R	✓
A	99	—	100	S	✗
B	100	—	99	R	✗
B	100	—	100	S	✓
C	99	—	99	R	✓
C	99	—	100	S	✗

Join ... on

> Select ename

from employee join Dept on depno = dno

where dname = 'R';

O/P

ename

A
C

Natural join

When we have a cartesian product with equality condition using columns having the same name in the where condition it is called natural join.

Employee(eno, ename, city, dno)

102

Dept(dno, dname);

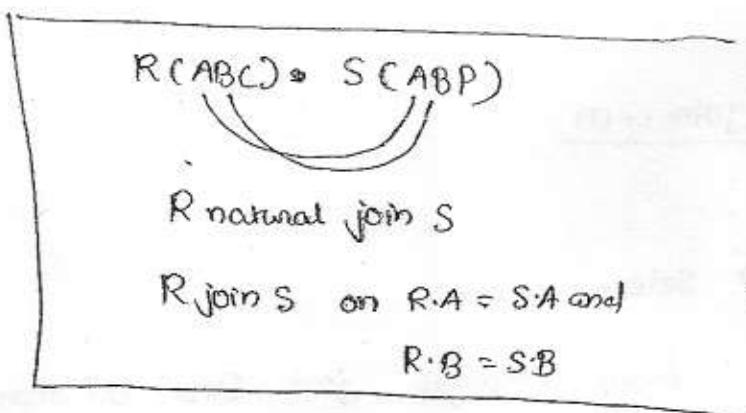
>

Select ename

from Employee natural join Dept
where dname = 'Research';

Implicit
Employee join dept on
Employee.dno = Dept.dno

O/p
 \Rightarrow ename
A
C



Outer Join

Faculty

Fid	Fname
1	A
2	B
3	C

Courses

cid	cname	fid
99	DBMS	1
100	CD	1
101	TOC	3
104	CDS	

<u>Fname</u>	<u>C-name</u>	<u>(O/P)</u>	<u>Cname</u>	<u>Fname</u>
A	DBMS		DBMS	A
A	CD		CD	A
B	null		TOC	C
C	TOC		CDS	null

Outer join is used when we want to output all rows from one table even if it does not have a corresponding row in another table.

- Left outer join
- Right outer join
- Full outer join

Left outer join:- Returns all rows from the first table even if there are no-matches in the second table.

Q find names of all faculty and courses they teaches (if any).

> Select Fname, Cname
from faculty Natural left outer join Courses;

<u>Fname</u>	<u>Cname</u>
A	DBMS
A	CD
B	null
C	TOC

Right - Outer join :- Returning all the rows from the second table even if there are no matches in the first table.

104

Q find all courses and name of the faculty if teaches the courses.

> Select course_name, faculty_name
from Faculty Natural right outer join Courses;

$\mathbb{C}(P)$

<u>Cname , Fname</u>	
DBMS	A
CD	A
TOC	C
CDS	null

Full outer join :- Returns all the rows from both the tables even if there is no matching row appearing in another table.

$$FOJ = \{10J\} \cup \{R0J\}$$

> Select cname, fname
from faculty Natural full outer join courses;

<u>Cname</u>	<u>Fname</u>
DBMS	A
CD	A
TOC	C
CDS	null
null	B.

Self JOIN

105

It is a join in which a table can be joined by itself.

Employee

<u>eno</u>	<u>ename</u>	<u>salary</u>	<u>mgreno</u>
1	Kiran	9000	—
2	John	6000	1
3	Rajesh	6000	1
4	Mathesh	4000	2

Q. Find em-name and his manager?

(Qp)

- > Select e.ename emname, m.ename manager
from Employee e, Employee m
where e.mgreno = m.eno;

<u>e-name</u>	<u>manager</u>
John	Kiran
Rajesh	Kiran
Mathesh	John

Q. Write a query to find no. of employees working under Kiran?

- > Select ~~emname~~ count(*)
from Employee e, Employee m
where m.eno = e.mgreno and m.ename = 'Kiran';

(Qp)

Count(*)
2

Note:-

Table aliases are mandatory in self joins.

Nested Query

A query inside a query is called nested query

Supplier (sid, Sname, City, Turnover)

Supply (sid, partid, qty)

Catalog (Partid, Pname, Color)

Q. Find name of the supplier who is supplying part-id 99.

> Select Sname

from Supplier, Supply

where Supplier.sid = Supply.sid and Supply.partid = 99;

<u>Supplier</u>		<u>Supply</u>		<u>%p Sname</u>
<u>sid</u>	<u>Sname</u>	<u>sid</u>	<u>partid</u>	<u>A</u>
1	A	1	98	
2	B	1	99	
		2	98	

m n

(m x n) - condition product is generated.

⇒ The memory is utilized more.

so we have to go for nested queries

> Select sname

107

from supplier

where sid in (Select sid

from supply

where partid = 99) ;

% sname
A

here $(m+n)$ tuples are considered which is very few compared with (mn) of previous joins.

In nested queries the inner query is evaluated first and the result is supplied to its outer query. Where inner query is independent and outer query depends on result of inner query.

Q

Write a query to find name of the suppliers who supplies blue color parts.

> Select sname

from supplier

where ~~Supply~~ sid

in (Select sid
from supply.

where partid in (Select partid

from catalog

where pa color = 'Blue'))

Note:-

Nested queries are evaluated from bottom to top.

Q. Find name of the supplier who supplies only supplies called A. ?

Ans

> Select pname
from catalog
where partid in (Select partid
from Supply
where sid in (Select sid
from supplier
where sname = 'A'));

Q. Find name of the supplier who has maximum turn over.

> Select sname
from supplier
where Turn over = (Select max(Turn over)
from supplier);

Q. Consider the following SQL query

Select sname
from supplier
where sid not in (Select sid
from Supply
where partid not in (Select partid
from catalog
where color \leftrightarrow 'Blue'));

which of the following is the correct interpretation of the above query.

109

- find the names of all suppliers who have supplied a non-blue part
- find the names of all suppliers who have not supplied a non-blue part
- names of all suppliers who have supplied only blue part
- find the names of all suppliers who have not supplied only blue part.

Supplier			Supply		Catalog		
Sid	name	to	Sid	partid	partid	color	Part
1	A	SL	1	98	98	Red	P
2	B	SL	1	99	99	Blue	Q
3	C	2L	2	98			
			3	99			

a)

A
B

b)

A
C

c)

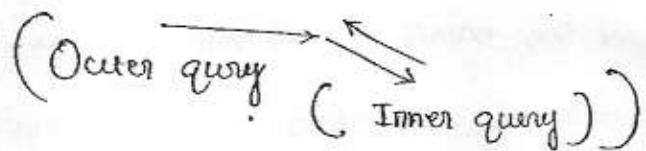
C

d)

B

{ 2 - 5.30 - TOC }
{ 6 - 9.00 - DBMS }

Correlated Subquery



In correlated subquery both outer and inner query are evaluated simultaneously, ie, for each row of outer query all the rows of inner query are evaluated, based on the result the tuple of the outer query is selected for the output.

The correlated sub-queries are executed

from TOP - BOTTOM - TOP in this order.

- Ex: find name of the suppliers who supplies part~~id~~ = 99

<u>Supplier</u>	
Sno	Sname
1	A
2	B

<u>Supply</u>	
Sno	partid
1	98
2	99
2	98

→ Select S.sname

from Suppliers

where 99 in (Select sp. partid

from Supply sp

where s.sno = sp.sno) ;

op s.sname
A

Student S1

Name	marks
A	100
B	500
C	300
D	400
E	200

Student S2

Name	marks
A	100
B	500
C	300
D	400
E	200

111

 $S1.\text{marks} \leq S2.\text{marks}$

> select $S1.\text{Name}$
 from Student S1

where $3 = C$ select count (distinct marks)
 from Student S2
 where $S1.\text{marks} \leq S2.\text{marks}$)

O/P $S1.\text{Name}$
C

IF there is
duplicate
marks

Q4 Consider the relation Book (Title, Price) contains the titles and prices of different books assuming that no two books have the same price. what does the following SQL query list?

Select title
 from Book B
 where (select count(*)
 from Book T
 where T.price > B.price) < 5;

0
 1
 2
 3
 4

- a) Titles of four most expensive books
- b) Titles of fifth most expensive book
- c) Titles of fifth most inexpensive book
- d) Titles of five most expensive books

Note :- When even the inner query uses the reference of outer query then both the queries are said to be correlated. 112

Book B

Title	Price
A	100
B	200
C	300
D	400
E	500
F	600

Book T

Title	Price
A	100
B	200
C	300
D	400
E	500
F	600

(d)

(a)

(b)

(c)

B

C

D

E

F

for A - 5

B	- 4	✓
C	- 3	
D	- 2	
E	- 1	
F	- 0	

5 most expensive books displayed.

Q : Title

B
C
D
E
F

Exists Operator

Exists operator is used for testing whether a given set is empty or not.

An exists operator on empty set returns false, while on non-empty set returning true.

exists (result) = true

exists () = false.

Q :- Find name of the supplier who supplies part-id 99.

(P.T.O)

> Select S.sname

from Supplier S

where exists (Select *

from Supply SP

where S.sno = SP.sno and SP.partid = 99);

Qp: S.sname

A

<u>A</u>		
P	Q	R
0	a	65
1	b	66
2	c	67
3	d	69

<u>B</u>		
P	S	T
0	A	82
1	A	81
2	S	82
5	A	80
1	S	83
3	A	84

Consider the above table of data and result of the following SQL query

> Select P

from B

where S='A' and exists (Select *

from A

where R>65 and B.P = A.P);

B Q/P

0 - x 1

1 - v ~~1~~

2 - x 3

5 - x ~~5~~

1 - ~~x~~ 1

3 - v ~~3~~

Set - comparison Operations

op any() ex:- $x < \text{any} (5, 10, 15)$

op all () ex:- $(x < 5) \text{ or } (5 < 10) \text{ or } (x < 15)$

..... ex:- $x < \text{all} (5, 10, 15)$

$(x < 5) \text{ and } (x < 10) \text{ and } (x < 15)$

SQL supports set comparison operators op any & op all

where 'op' is any valid arithmetic comparison operators.

Op-any

Compares a value with each value in a set and returns true if any value is compared according to given conditions.

Op-all

Compares a value with each value in a set and returning true if the given condition satisfied for every value in the set.

Q. find name of the suppliers whose turn over is better than the turn over of some ~~suppliers~~ of suppliers of Hyderabad

Supplier

Sno	Sname	City	Turnover
1	A	Hyd	3L
2	B	Bang	4L
3	C	Hyd	5L
4	D	Delhi	6L

> Select sname
from Suppliers

where City <> 'Hyd' and

turnover > any (Select turn-over
from Suppliers
where City = 'Hyd');

O/p: sname
B
D

Ques. Find name of the suppliers whose turnover is better than the turnover of all suppliers of 'Hyd'.

> Select sname

from Suppliers

where city <> 'Hyd' and turnover > all (Select turnover

from Suppliers
where City = 'Hyd');

O/p: sname
P

Ques. Consider the following tables

Table 1

T1A	T1B
a	aa
b	bb
c	cc

Table 2

T2A	T2B	T2C
a	a	a
b	a	null

Ques. Find the no. of rows returned by the each of the following SQL query.

(P.Q.O)

3) Select *

from Table1

where T1A = all (select T2B

from Table2

where T2B >= 'b');

$= \text{all } C;$

Ans: 2 - rows returned



b) Select *

from Table1

where T1B in (select T2A

from Table2

where T2C is null);

Ans: 0 - rows returned

Select *

from Table1

where T1A in (select T2C

from Table2);

Ans: 1 - row returned

Select *

from Table1

where exists (select count(*)

from Table2

where T2B = 'x');

Ans: 3 - rows returned

e) Select *

From Table1

Where not exists (Select *

From Table2

Where $T2C \geq 'a'$ and $T2C < T1A$);

Ans: 1 - rows returned

f) Select *

From Table2

Where $T1A = \text{all}$ C Select $T2C$

From Table2

Where $T2C \geq 'x'$);

Ans: 3 - rows returned

Ques

Select S.sname ~~from~~

From Sailors S

Where not exists ((Select B.bid from Boats B) $\{100, 200, 300\} - \{100, 300\}$)
except

(select R.bid from Reserves R) where R.sid = S.sid

Sailors (Sid, sname)		Reserves (Sid, bid)		Boats (Bid, bname)	
1	A	1	100	200	BB
2	B	1	200	200	BBB
3	C	2	300	300	BBBB
		2	200		

The above query returns _____

a) names of sailors who have reserved any boat;

b) names of sailors who have ^{not} reserved any boat

c) names of sailors who have reserved all boats

d) none.

consider the following relationschema where the primary keys are shown, underlined

118

Student (rollno, name, address) ----- 120

Enroll (rollno, courseno, course name) ----- 8

The no. of tuples in the Student and enrolled tables are 120 and 8 respectively. what are the maximum and minimum. no. of tuples that can be present in (student natural join enroll)

- a) 960, 8
- b) 960, 120
- c) 120, 8
- d) 8, 8

Note:- ~~Normal~~

when two tables are joined (natural join) w.r.t primary key and foreign key the no. of tuples present in the resulting relation is always equals to tuples in foreign key relation.

Q. Consider the following table

A	B	C
P	S	10
Q	R	5
R	S	7

A	B	C
P	S	10
Q	R	5
R	S	7

> Select count(*)
from C

(Select A,B from Table1) as X

natural join

(Select B,C from Table1) as Y
);

5

The result of the above query is —

119

- a) 3
- b) 5
- c) 6
- d) 9

P-102

Staff C (StaffNo, name, dept, Skill Code)

Skill C (SkillCode, description, chargeOutRate)

Project C (ProjectNo, startDate, end Date, budget, Project manager Staff no)

Booking C (StaffNo, ProjectNo, date WorkedOn, timeWorkedOn);

1) Select - *
from Skills

where chargeoutrate > 60

Order by Description;

2) Select *

from Staff

where dept = 'Special projects' and skill code =

(Select skillcode

from Skills

where description = 'programmer');

3)

2 pm - 5:30 pm TOR
6 pm - 9:00 DBMS

(3) > Select S.name, B.projectNo, B.dateWorkedOn, B.timeWorkedOn
from Staff S Booking B

where B.project no in

(Select projectno

from projects

where startdate <= 01 - July - 1995 and
enddate >= 31 - July - 1995)

and S.staffno = B.staffno

Order by S.name, B.project No, B.date Worked On;

4) Select count(*)

from Staff

where SkillCode in (Select SkillCode from

from skill

where Description = "Programmer");

5)

Select *

from Projects

where projectNo in (Select projectNo

from Booking

Group by ProjectNo

having Count(*) >= 2);

(OR)

Select *

from Projects P

where (select Count(*)

from Booking B

where P.project = B.projectno) >= 2);

(06) Select avg(ChargeOutRate)

from Skill;

(07) Select * From Staff where SkillCode

in (Select SkillCode
from Skill

where ChargeOutRate > (Select avg(ChargeOutRate)
from Skill));

Final

Examination

Relational Algebra

Relational algebra is a procedural language. Queries in Relational algebra are composed using a collection of operators and each query describes a step by step procedure for computing the desired answer.

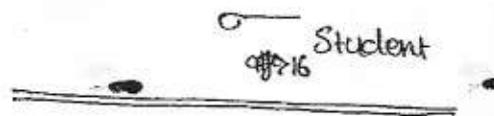
i.e., Relational algebra expression represents a query evaluation plan.

Operators in Relational Algebra

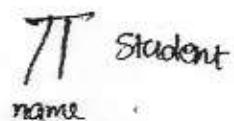
- ① Selection (σ) - "Rows"
- ② Projection (Π) - "Columns"

Student (eno, name, age);

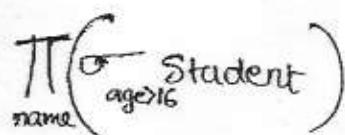
- ① find all students of age above '16' ?



- ② Display the names of all students



- ③ find names of all students of age above '16'.



Consider the following set domain Statement

123

S1: $\Pi_{\text{name}} (\sigma_{\text{age} \geq 16} \text{ Student})$

S2: Select name
from student
where age > 16;

Which of the following is true about the results of S1 + S2 always

- a) $S_1 = S_2$
- b) $S_1 \subseteq S_2$
- c) $S_2 \subseteq S_1$
- d) $S_1 \neq S_2$

Student	
name	age
A	17
B	18
A	20

Op $\frac{S_1}{A}$ $\frac{S_2}{A}$
 $\quad \quad B$ $\quad \quad B$
 $\quad \quad A$

Note: Relational Algebra assumes duplicate elimination is implicit

Q: Which of the following is true about an SQL Query?

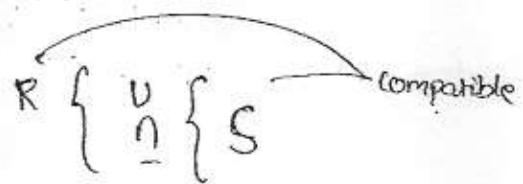
Select l from R where c;

- a) $\Pi_c (\sigma_l R)$
- b) $\sigma_l (\Pi_c R)$
- c) $\sigma_c (\Pi_l R)$
- d) $\Pi_l (\sigma_c R)$

3)

Set - Operations

124



Relational algebra supports ~~selection~~

- (i) Set union (U)
- (ii) Set intersection (N)
- (iii) Set difference (-)

between the results of 2 relational algebra expressions
if they are compatible

Depositor (cust-name, ac-no)

Borrower (cust-name, loan-no)

Loan (loan-no, branch-name, City, amount);

Find name of the customer who have an account or loan or both at the bank

$$(\pi_{\text{cust-name}} \text{Depositor}) \cup (\pi_{\text{cust-name}} \text{Borrower})$$

\cap \leftarrow who have both account and loan

$-$ \leftarrow who have an account but no loan

4)

Cross Product (X)

$R \times S$ - returns a relation instance whose schema contains all the fields of R followed by all the fields of S. The result of $R \times S$ contains a tuple $\langle R, S \rangle$ for each $R \in R, S \in S$.

Q) find name of the customer who have a loan in ABIDS branch.

125

$$\Pi_{\text{Cust-name}} \left(\sigma_{\begin{array}{l} \text{Borrower-loan-no} = \text{Loan-loan-no} \\ \wedge \text{Loan.branch} = 'ABIDS' \end{array}} (\text{Borrower} \times \text{Loan}) \right)$$

5) Rename (ρ_{rho})

ex:-

1. $\rho [A, C \text{ Borrower} \times \text{Loan}]$
2. $\rho [B, (\sigma_{\begin{array}{l} \text{B-loan-no} = \text{L-loan-no} \\ \wedge \text{L.branch-name} = 'ABIDS' \end{array}} A)]$
3. $\Pi_{\text{Cust-name}} B$

Rename operator is used to represent relational algebra expression with a shorter name.

JOINS

Join is defined as a cross product followed by selection then projection.

Variants of Joins

1) Conditional join (\bowtie_c)

Conditional join is a join in which the two relations are joined

based on some condition.

$$\Pi_{\text{Cust-name}} \left(\sigma_{\text{Branch-name} = 'ABIDS'} (\text{Borrower} \bowtie_c \text{Loan}) \right)$$

② Equi Join ($\bowtie_=_$)

Equi join is a join in which the join condition must be an equality of the form.

$R \bowtie S$

$R.A = S.P$

$$\frac{\begin{array}{c} R \bowtie S \\ R.A = S.P \end{array}}{\begin{array}{c} A B C \\ P Q R \end{array}} = \left(\Pi_{A, B, C, Q, R} \left(\sigma_{R.A = S.P} (R \times S) \right) \right)$$

Projected out

Note:-

Every equi join is called a conditional join, but reverse is not always.

③ Natural join (\bowtie)

Natural join is a join in which the join condition is implicit ~~as~~ ~~a~~ equality on all columns having the same name.

$$\frac{R}{\begin{array}{c} A B Q \\ \hline \end{array}} \quad \frac{S}{\begin{array}{c} A P Q \\ \hline \end{array}}$$

$$\frac{\begin{array}{c} R \bowtie S \\ A B Q \ A P Q \end{array}}{\begin{array}{c} A B Q \ A P Q \\ \hline \end{array}} \Rightarrow \Pi_{A B Q P} \left(\sigma_{R.Q = S.Q} (R \times S) \right)$$

projected out

ex:-

$$\Pi_{\text{cust_name}} \left[\sigma_{\text{branch_name} = \text{Abidg}} (\text{Borrower} \bowtie \text{Loan}) \right]$$

Ques.

Table 1 - T₁

P	Q	R
11	a	b
16	b	a
26	a	7

Table 2 - T₂

A	B	C
11	b	7
26	c	4
11	b	6

Consider the tables shown above. What is the no of tuples in the result of the given algebraic expression

i) $T_1 \bowtie T_2$ \Rightarrow No. of tuples = 3
 $T_1 \cdot P = T_2 \cdot A$

ii) $T_1 \bowtie T_2$ \Rightarrow No. of tuples = 2
 $T_1 \cdot Q = T_2 \cdot B$

iii) $T_1 \bowtie T_2$ \Rightarrow No. of tuples = 1
 $(T_1 \cdot P = T_2 \cdot A \text{ and } T_1 \cdot R = T_2 \cdot C)$

iv) $T_1 \bowtie T_2$ \Rightarrow No. of tuples = 9

Note:- If there is no column in common, Natural join results in a cross product

Ques. Consider the following table

A		
ID	Name	age
12	A	60
15	S	24
99	R	11

B		
ID	Name	age
15	S	24
25	H	40
98	T	20
99	R	11

C	
id	phone
12	2200
99	2100

$(A \cup B) \bowtie C$

$Age > 40 \vee C.ID < 15$

Ques. find the no. of rows returned by the above relational algebra expression. assume that the schema of $A \cup B$ is same as that of A.

- a) 7
b) 4
c) 5
d) 9

Q1

Let $R_1(\underline{ABC})$ and $R_2(\underline{CDE})$ be two relationship schemas where the primary keys are shown underlined. and let C be a foreign key in R_1 referring to R_2 . Suppose there is no violation of referential integrity constraint in the corresponding relation instances γ_1 and γ_2 which of the following relational algebra expressions would necessarily produce an empty result?

- a) $\Pi_b(\gamma_2) - \Pi_c(\gamma_1)$
 b) $\Pi_c(\gamma_1) - \Pi_b(\gamma_2)$ $\{F, K\} - \{P, H\} = \underline{\{ \}}$
 c) $\Pi_b(\gamma_1 \bowtie_{C \neq D} \gamma_2)$
 d) $\Pi_c(\gamma_1 \bowtie_{C \neq D} \gamma_2)$

2pm - 5:30 - TOC } 3/2

Division Operation (\div)

129

Find the name of students who have registered for all courses?

Student		Registration			Courses	
eno	name	free	eno	cno	cno	name
1	A	ok	1	99	99	DB
2	B	ok	2	99	100	TOC
			ok	100		

$$\pi_{\text{name}} \left[\text{Student} \bowtie \left[\begin{array}{l} (\pi_{\text{eno}, \text{cno}} \text{ Registration}) \div (\pi_{\text{cno}} \text{ Courses}) \\ \text{eno} \end{array} \right] \right]$$

~~copy:~~ name
B

Consider two relation instances A and B in which A has two fields X and Y and B has just one field Y with the same domain as in A. We define the division operator $A \div B$ as the set of all X values such that for every Y value in B there is a tuple X, Y in A.

<u>$A(XY)$</u>	<u>$B_1(X)$</u>	<u>$A \div B_1(X)$</u>	<u>$A \div B_2(X)$</u>	<u>$A \div B_3(X)$</u>
$x_1 y_1$	<u>y_1</u>	x_1	x_1	x_1
$x_1 y_2$	<u>$B_2(X)$</u>	x_2	x_2	x_2
$x_2 y_2$		x_3		
$x_3 y_1$	<u>y_1</u>			
$x_1 y_3$	<u>y_2</u>			
$x_2 y_1$	<u>$B_3(X)$</u>			
		y_1		
		y_2		
		y_3		

Ques

~~Ques~~
find name of the customers who have a loan in all branches of Hyd.

Borrower (cust-name, loan-no)

Loan (loan-no, branch-name, city, amount)

Ans

$$\left[\prod_{\substack{\text{City=Hyd} \\ \text{Customer, Branch}}} (\sigma_{\text{Borrower} \bowtie \text{Loan}}) \right] \div \left[\prod_{\text{Branch-name}} (\sigma_{\text{City=Hyd}} \text{Loan}) \right]$$

~~Ques~~ find name of the publisher who published all category of Books?

Books (ISBN, title, Category, Price, Pub-Id)

Publisher (Pub-Id, Pname, email)

Ans

$$\left[\prod_{\text{Pname, Category}} (\text{Books} \bowtie \text{Publisher}) \right] \div \left[\prod_{\text{Category}} (\text{Books}) \right]$$

Q1: Consider two tables R_1 and R_2 with N_1 and N_2 rows. where $N_2 > N_1$, find min and maximum rows for each of the following expressions

131

	<u>expression</u>	<u>min</u>	<u>max</u>
1)	$\sigma_{age \geq 30} R_1$	0	N_1
2)	$\pi_{name} R_2$	1	N_2
3)	$R_1 \cup R_2$	N_2	$N_1 + N_2$
4)	$R_1 \cap R_2$	0	N_1
5)	$R_1 - R_2$	0	N_1
6)	$R_1 \bowtie R_2$	0	$N_1 * N_2$

Q2 Consider the following Relations $R(\overbrace{ABC}^F)$ $S(\overset{PK}{BDE})$ with the following functional dependancies $A \rightarrow C$ $B \rightarrow A$, the relation R contains 200 tuples and S contains 100 tuples what is the maximum no. of tuples possible in $R \bowtie S$?

Ans

100

Q3 Let γ be a relation instance of schema $R(ABCD)$ we define $\gamma_1 = \pi_{ABC}(\gamma)$ and $\gamma_2 = \pi_D(\gamma)$. Let $S = \gamma_1 \bowtie \gamma_2$ assuming that the decomposition of γ into γ_1 and γ_2 is lossy then which of the following is true

- a) $S \subset R$
- b) $S = R$
- c) $R \subset S$
- d) $\gamma * S = S$

⑥ The decomposition generates spurious tuples if it is lossy

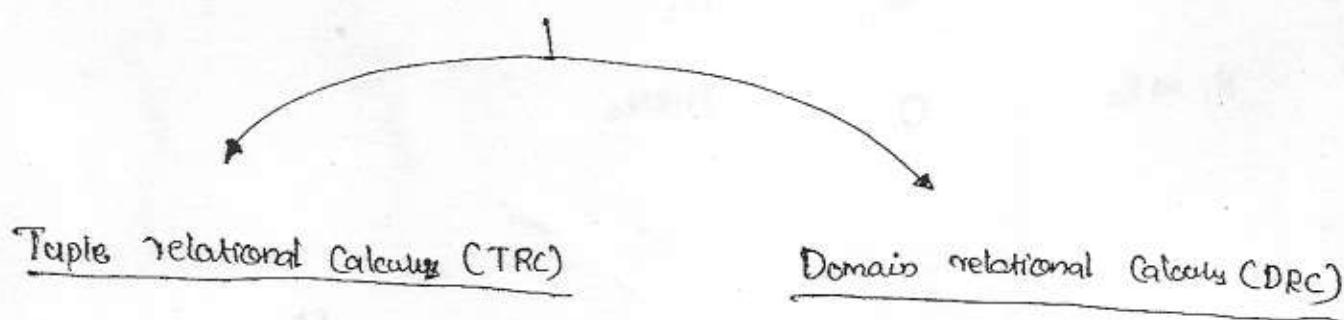
132

Relational Calculus → (Based on predicate calculus)

Relation Calculus :- In this query describes the result set without specifying how the answer is to be computed.

This non-procedural style of querying is called relational calculus.

Two-variants of Relational Calculus



→ Query describes results in the form of set of tuples

→ Query describes results in the form of set of columns (Domain)

→ Tuple Variable :-

It is a variable that takes on tuples of a Relation schema as values.

→ Domain Variable :-

It is a variable that ranges over the domain of some attributes.

→ Form of Query

$\{ T \mid P(T) \}$

formula which describes
Tuple Variable

→ form of Query

$\{ \langle x_1, x_2, \dots, x_n \rangle \mid P(x_1, x_2, \dots, x_n) \}$

Domain Variables

formula which describes the domain

Borrower (Cn, Lb
cust-name, loan-no)

133

Loan (Ln, Bn, Am
loan-no, branch-name, amount)

Q4 find all the loans of an amount above 5000

TRC: { T / T ∈ Loan (T.amount > 5000) }

DRC: { < loan-no, branch-name, amount > / < loan-no, branch-name, amount >
∈ Loan (amount > 5000) }

Q5 Display loan-no and amount of all loans

TRC: { T / ∃ L ∈ Loan (T.loan-no = L.loan-no ∧ T.amount = L.amount) }

DRC: { < loan-no, branch-name > / ∃ branch-name (< loan-no, branch-name, amount >
∈ Loan) }

Q6 find name of the customer who have a loan in ABIDS branch
and display the loan amount also.

TRC: { T / ∃ B ∈ Borrower (T.cust-name = B.cust-name) ∧
∃ L ∈ Loan (L.branch-name = 'ABIDS' ∧ B.loan-no = L.loan-no ∧
T.amount = L.amount) }

(OR)

$\{ T / \exists L \in \text{Loan} (L.\text{branch-name} = 'ABIDS' \wedge T.\text{amount} = L.\text{amount})$
 & $\exists B \in \text{Borrower} (B.\text{loan-no} = L.\text{loan-no})$
 & $\exists B \in \text{Customer} (T.\text{cust-name} = B.\text{cust-name})) \}$

DRC:

$\{ \langle C_n, A_m \rangle \mid \exists L_b (\langle C_b, L_b \rangle \in \text{Borrower} \wedge$
 $\exists L_n, B_n (\langle L_b, B_n, A_m \rangle \in \text{Loan} (B_n = 'ABIDS')$
 $\wedge L_n = L_b)) \}$

Q1 What is the result of the following TRC expression?

A1:

$\{ T / \exists S \in \text{Student} (T.\text{name} = S.\text{name} \wedge$
 $\exists C \in \text{Course} (C.\text{sno} = S.\text{sno} \wedge C.\text{course-name} = 'CS')) \}$

Ans

It finds the names of all students studying the course 'CS'.

Q2 Translate the following TRC expressions into relational algebra.

$\{ T / \text{TER} (T.A = 10 \wedge T.B = 20) \}$

Ans $\underline{\underline{(\sigma_{A=10} R) \wedge (\sigma_{B=20} R)}} \text{ (OR)} \underline{\underline{(\sigma_{A=10 \wedge B=20} R)}}$

① R(ABCDE)

$$f: \left\{ \begin{array}{l} A \rightarrow C \\ B \rightarrow E \end{array} \right. , \left\{ \begin{array}{l} C \rightarrow D \\ B \rightarrow E \end{array} \right. , \left\{ \begin{array}{l} B \rightarrow E \end{array} \right. \right\}$$

BCNF 2NF INF

R is in 1NF

C.K: AB

$$B^+ = \left\{ \begin{array}{l} B, E \\ \downarrow \end{array} \right\} R_1 \checkmark_{BCNF}$$

$$\left(\begin{array}{c} \square \\ A \\ B \\ C \\ D \end{array} \right) R_2 \checkmark_{2NF} \xrightarrow{3NF} C^+ = \left\{ \begin{array}{l} C, D \\ \downarrow \end{array} \right\} R_3 \checkmark_{BCNF}$$

$$\left\{ \begin{array}{l} A, B, C \\ \downarrow \\ T \end{array} \right\} R_4 \checkmark_{BCNF}$$

R in BCNF

$$R_1(BE) : f_1: \{B \rightarrow E\}$$

Decomposition is lossless & D.P

$$R_3(CD) : f_2: \{C \rightarrow D\}$$

$$R_4(ABC) : f_3: \{A \rightarrow C\}$$

② R(ABCDEF)

$$f: \left\{ \begin{array}{l} A \rightarrow BC \\ BC \rightarrow AD \\ B \rightarrow F \\ D \rightarrow E \end{array} \right. \right\}$$

BCNF BCNF 1NF 2NF

C.K = A, BC, F

$$B^+ = \left\{ \begin{array}{l} B, F \\ \downarrow \end{array} \right\} R_1 \checkmark_{BCNF}$$

$$\left(\begin{array}{c} \square \\ \square \\ A \\ B \\ C \\ D \\ E \\ F \end{array} \right) R_2 \checkmark_{2NF} \xrightarrow{3NF} D^+ = \left\{ \begin{array}{l} D, E \\ \downarrow \end{array} \right\} R_3 \checkmark_{BCNF}$$

$$\left\{ \begin{array}{l} A, B, C, D \\ \downarrow \\ T \end{array} \right\} R_4 \checkmark_{BCNF}$$

BCNF

+

Lossless - join

+

D.P

③ R(ABCDE)

$$f: \left\{ \begin{array}{l} A \rightarrow BC \\ \text{BCNF} \end{array}, \begin{array}{l} CD \rightarrow E \\ \text{BCNF} \end{array}, \begin{array}{l} B \rightarrow P \\ \text{3NF} \end{array}, \begin{array}{l} E \rightarrow A \\ \text{BCNF} \end{array} \right\}$$



R is in 3NF

$$B^t = \left\{ \begin{array}{l} BD \\ \text{3NF} \end{array} \right\} R_1, f_1: \{ B \rightarrow D \}$$

$$(ABCE) R_2 \quad f_2: \left\{ \begin{array}{l} A \rightarrow BCE \\ BC \rightarrow AE \\ E \rightarrow ABC \end{array} \right\}$$

$$\begin{aligned} & \text{BCNF} \\ & + \\ & \text{lossless join} \\ & + \\ & \left[\begin{array}{l} CD^t \text{ using } f_1, f_2 \\ CD^t = \{ CD \} \\ CD \rightarrow E \text{ lost} \end{array} \right] \\ & \text{Not D.P} \end{aligned}$$

Because $CD \rightarrow E$ is not in R_1 or R_2 then to preserve

$$\text{Dependencies take } R_3 \left(\begin{array}{l} CDE \\ \text{3NF} \end{array} \right) \quad f_3: \{ CD \rightarrow E \}$$

④ R(ABC(D))

$$f: \left\{ \begin{array}{l} AB \rightarrow CD \\ \text{BCNF} \end{array}, \begin{array}{l} D \rightarrow A \\ \text{3NF} \end{array} \right\}$$

CK AB
DB

$$D^t = \left\{ \begin{array}{l} DA \\ \text{3NF} \end{array} \right\} R_1 \quad \text{BCNF} \quad f_1: \{ D \rightarrow A \}$$

BCNF
+
Lossless join

$$(BCD) R_2 \quad \text{BCNF} \quad f_2: \{ BD \rightarrow C \}$$

take $AB \rightarrow C$ in $R_3(ABC)$ $f_3: \left\{ \begin{array}{l} AB \\ \text{3NF} \end{array} \rightarrow C \right\}$

$$\text{using } f_1, f_2: BD^t = \{ BDAC \}$$

$$AB^t = \{ AB \}$$

take $AB \rightarrow D$ in $R_4(ABD)$ $f_4: \left\{ \begin{array}{l} AB \\ \text{3NF} \end{array} \rightarrow D \right\}$

$AB \rightarrow CD$ is lost

R_4 is 3NF

$R_6(BD)$ BCNF $AB \rightarrow D$ is lost

Note:- Dependency preserving decomposition into BCNF, may not be possible always.

Observation

If ~~the~~ relation contains overlapping candidate keys. Dependency preserving BCNF decomposition may - not be possible.

Note:

- ① Relation schema R with no non-trivial functional dependencies are always in BCNF.
- ② A Relation is failed in BCNF only if there should be at least one non-trivial dependency $X \rightarrow Y$ other with X as not a Superkey.
- ③ Relation R with only 2-attribute is always in BCNF
- ④ If Relation R consists only of simple candidate keys then R always in 2NF.
- ⑤ If Relation R consists only of prime attribute then R - always in 3NF
- ⑥ If relation consists of Only simple candidate keys and R is in 3NF then R must be in BCNF

(S.A) 2-5-30 NA - 2
C-9. DBMS - 9

Concurrent execution

↑ (Interleaving execution of the operations of a Transaction)

139

Why?

- 1) For avoiding longer waiting time
- 2) If transaction consists of multiple steps. Some involves I/O activities and others involve CPU activities. In a computer system CPU and I/O operations can be done in parallel. Therefore I/O activity can be done in parallel with processing at the CPU.
- 3) The processor and Disk utilization increases.

Schedule

It represents the order in which instructions of a transaction are executed.

The problems due to concurrent execution of the transactions

① Lost Update problem. (W-W conflict)

		$A \xrightarrow{50} B$	$A \xrightarrow{4\%} A$
		T_1	T_2
	Read(A)		Read(A)
	$A = A - 50$		$x = A * 0.04$
	<u>Write(A)</u>		<u>Write(A)</u>
	Read(B)		
	$B = B + 50$		
	<u>Write(B)</u>		

A
1000

B
200

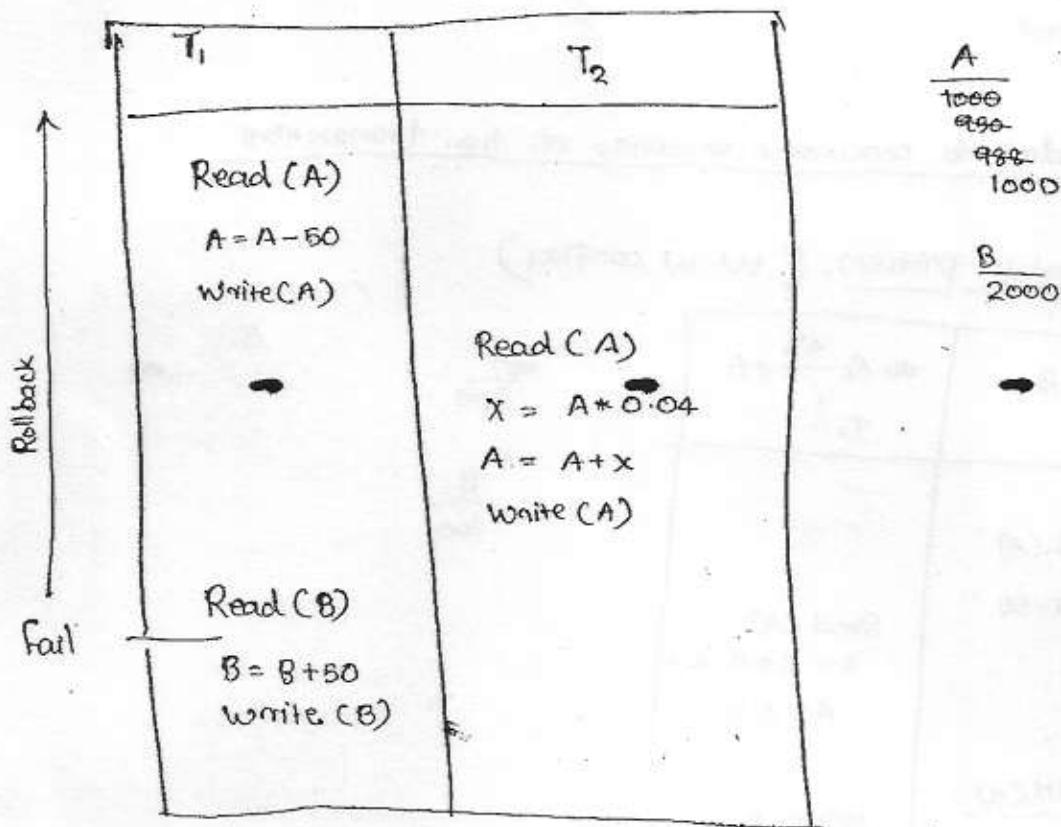
5%

Suppose that the operations of T_1 and T_2 are interleaved in such a way that T_2 reads the value of account A before T_1 updates, now when T_2 updates the value of account A in the data base, the value of account A updated by transaction T_1 is overwritten and hence is lost. This is known as lost update problem.

Test:-

If there are any two write operations of different transactions, between that there is no read operation the second write operation overwrites the first write. hence it causes loss of first updation.

2) Dirty Read problem (W-R conflict)



Reading on un-committed data is called dirty read operations.

Unrepeatable transactions

T ₁	T ₂	
		A 20,000
Read(A)	Read(A) A = A - 15,000 write(A)	
Read(A) A = A - 10,000 write(A)		
<u>fail</u>		

When a transaction tries to read the value of a data item twice and another transaction updates the same data item inbetween the two read operations of the first transaction, as a result the first transaction reads varied values of same data item during its execution this is known as un-repeatable reads.

④ phantom tuple (Phantom phenomenon)

	T ₁	T ₂
eno ename salary		
1 A 5000	select * from Emp where Salary > 3000	
3 C 4000		
		insert into Emp Values (4,D,3500);
eno - ename salary	Select * from Emp where salary > 3000;	
1 A 5000		
3 C 4000		
4 D 3500		

Employee		
eno	ename	salary
1	A	5000
2	B	2000
3	C	4000
4	D	3500

Transaction T_1 may read set of rows from a Table based on 142

Some condition, now suppose that a transaction T_2 inserts a new row that also satisfies the ~~where~~ clause condition of T_1 .

If T_1 is repeated then T_1 will see a row that previously did not exist called a phantom tuple.

3) Incorrect Summary Problem

T_1	T_2
Read (X) $X = X + 500$. write (X);	Sum = 0 Read (K) Sum = Sum + K ;
Read (Y) $Y = Y + 200$. write (Y);	Read (X) Sum = Sum + X ; Read (Y) Sum = Sum + Y ;

	Before T_1	After T_1
K =	50	50
X =	100	600
Y =	200	400
	<u>350</u>	<u>1050</u>
	↑ Correct o/p's	

O/P : 850 - incorrect

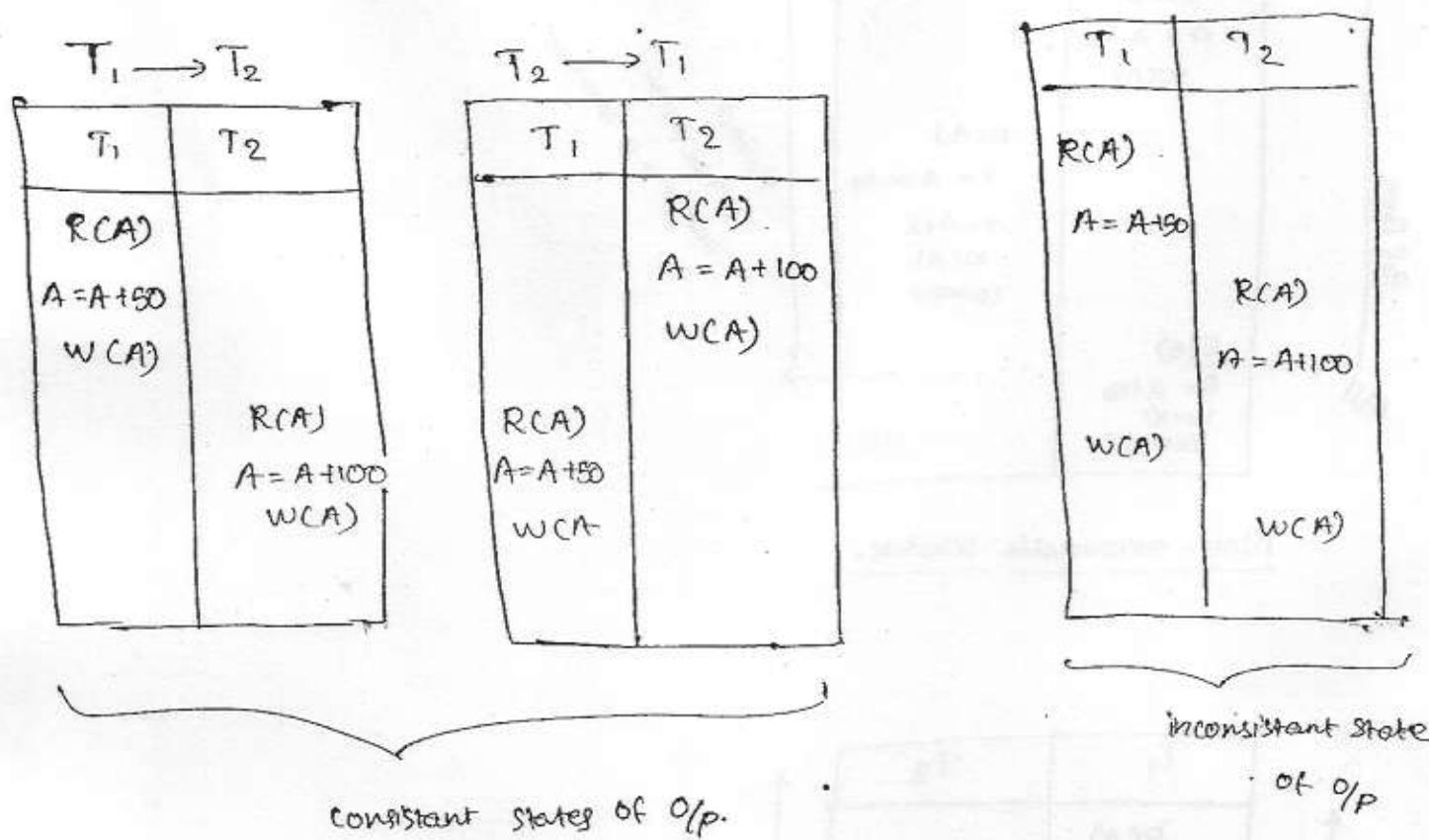
50 +
600 +
200

When a transaction tries to read the value of data

If one transaction is calculating an aggregate summary function on a number of records while other transaction updating some of them records the aggregate function may calculate some values before they are updated and others after they are updated results in incorrect summary.

Characterizing schedules

1) Serial schedule :-



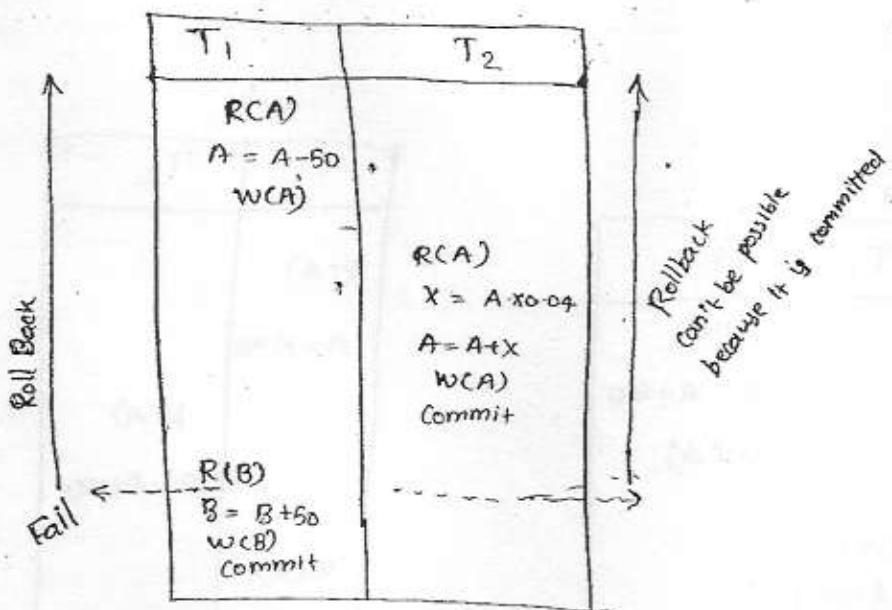
- Serial schedules that does not interleave the actions of any operations of different transaction.
- When transactions are executing serially, which always ensures a consistent state.
- If there are n-transactions in a schedule the possible no. of serial schedules are $n!$ ($n!$ consistent states are possible)

2) Complete schedule :-

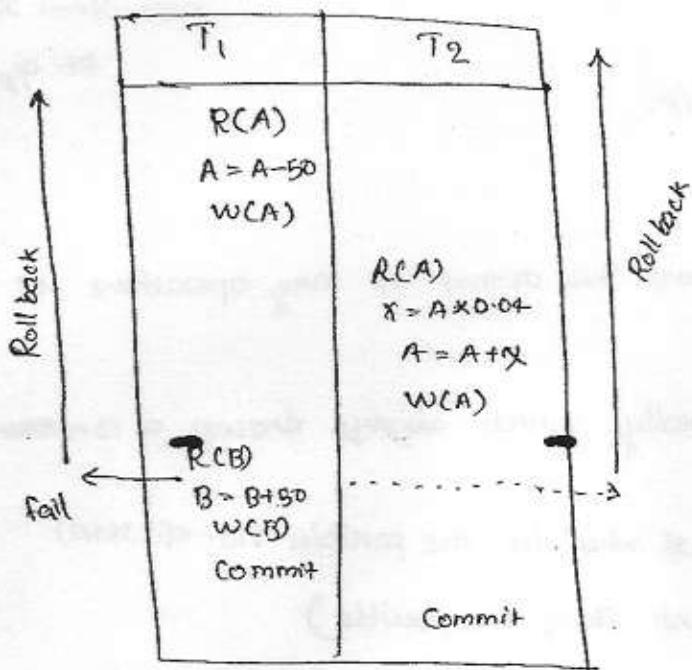
T ₁	T ₂
R(A)	
A = A - 50	R(A)
W(A)	
Commit	A = A + 50
	W(A)
	abort

A schedule is said to be complete if the last operation of each transaction is completed either abort or commit.

3) Recoverable schedule



Non-recoverable schedule.



Recoverable schedule

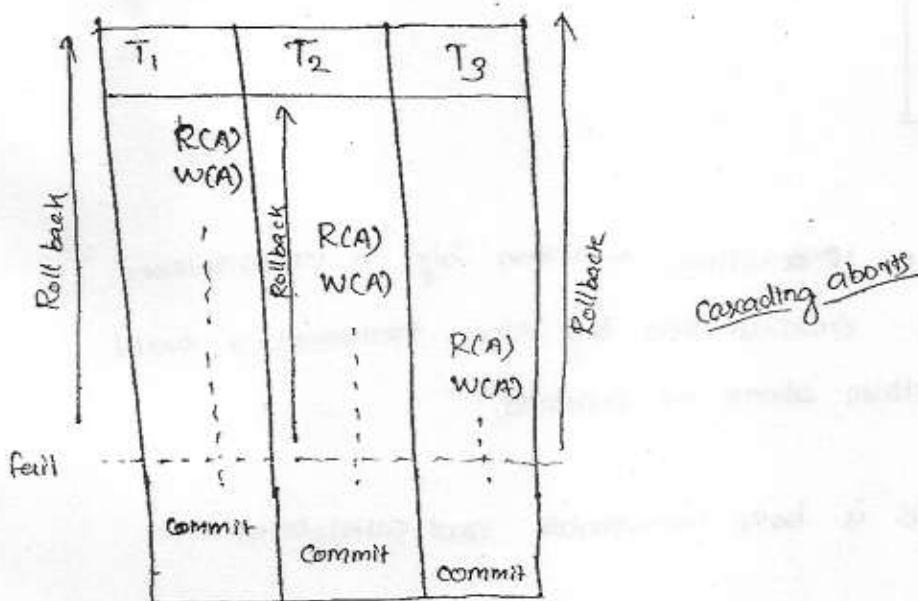
A recoverable schedule is one where for each pair of transactions T_i and T_j such that T_j reads a data item that was previously written by T_i , then the commit operation of T_i should appear before commit operation of T_j .

4) Cascadeless schedule

145

Cascading aborts :-

If one transaction failure causes multiple transactions to roll back, it is called cascading rollback or cascading aborts.



Cascadeless schedule :-

A cascadeless schedule is one, where for each pair of transactions T_i and T_j such that T_j reads a data item that was written by T_i then the commit operation of T_i should appear before the read operation of T_j.

T ₁	T ₂	T ₃
R(A) W(A) Commit	R(A) W(A) Commit	R(A) W(A) Commit

cascadeless schedule

3) Strict Schedule.

T ₁	T ₂
R(A)	
W(A)	
Commit	W(A)

Schedule that is Strict if a value written by a transaction can not be read or over-written by other transactions until the transaction is either aborts or commits.

Every Strict schedule is both recoverable and cascadeless.



14-12-19

4) Equivalent Schedule

The two schedules S_1 and S_2 are said to be equivalent schedules if they produce the same final database state.

(1) Result equivalent Schedules

Two schedules are said to be result equivalent if they produce the same final database state for some initial values of data.

	S_1	S_2	A
	R(A)	R(A)	100
110	$A = A + 10$	$A = A + 11$	110
110	W(A)	W(A)	110

① S_1 and S_2 are result equivalent for $A = 100$

Qn. Is the following schedules are result equivalent for the initial value of X and Y is (2,5) respectively.

147

S_1

T_1	T_2
$R(X)$	
$X = X + 5$	
$W(X)$	

T_1	T_2
	$R(X)$
	$X = X + 3$
	$W(X)$

T_1	T_2
$R(Y)$	
$y = y + 5$	
$W(Y)$	

S_2

T_1	T_2
	$R(X)$
	$X = X + 3$
	$W(X)$
	$R(Y)$
	$y = y + 5$
	$W(Y)$

$\frac{X}{2}$
 $\frac{Y}{10}$
21

$\frac{X}{2}$
 $\frac{Y}{10}$

$\left\{ \begin{array}{l} S_1 \text{ and } S_2 \text{ are not} \\ \text{result equivalent} \end{array} \right\}$

$\frac{X}{2}$
 $\frac{Y}{10}$

$\frac{X}{2}$
 $\frac{Y}{10}$

② Conflict equivalent

Conflict Operations

T_i	T_j
$R(A) \dots W(A)$	
$W(A) \dots R(A)$	
$W(A) \dots W(A)$	

Conflict operations

$R(A) \quad R(A)$

Operating on diff. data

non-conflicting operations.

Two schedules are said to be conflict equivalent if all conflicting operations in both the schedules must be executed in the same order.

S_1

T_1	T_2
$R(A) \dots W(A)$	$R(A) \dots W(A)$
$R(B) \dots W(B)$	

S_2

T_1	T_2
$R(A) \dots W(A)$	
$R(B) \dots W(B)$	$R(A) \dots W(A)$

③ S_1 is conflict equivalent to S_2

$S_1 \underset{C}{\equiv} S_2$

Ques Test which of the following schedules are conflict equivalent? 148

S₁: R₁(A) W₂(B) W₁(A) W₂(B)

S₂: R₂(B) R₁(A) W₂(B) W₁(A)

} no conflict is there in both schedules, $S_1 \sqsubseteq S_2$

S₁: R₁(A) W₁(A) R₂(B) W₂(B) R₁(B)

S₂: R₁(A) W₁(A) R₁(B) R₂(B) W₂(B)

$S_1 \neq S_2$

S₁ not conflict equivalent to S₂

S ₁	
T ₁	T ₂
R(A)	
	W(B)
	R(B)
	W(B)

W₂(B) \rightarrow R₁(B)

S ₂	
T ₁	T ₂
R(A)	
	W(A)
	R(B)
	W(B)

R₁(B) \rightarrow W₂(B)

S ₁	
T ₁	T ₂
R(A)	R(B)
	W(B)

S ₂	
T ₁	T ₂
R(A)	R(A)
	W(B)

$\Rightarrow S_1 \sqsubseteq S_2$

Conflict Serializable

A schedule is said to be conflict serializable if it is conflict equivalent to a serial schedule.

S ₁	
T ₁	T ₂
R(A), W(A)	
	R(A), W(A)
R(B), W(B)	
	R(B), W(B)

S ₂	
T ₁	T ₂
R(A), W(A)	
	R(A), W(A)
R(B), W(B)	
	R(B), W(B)

S₁ is conflict serializable to serial schedule T₁ \rightarrow T₂

T ₁	T ₂
R(A) W(A) R(B)	
	R(A) W(A). R(B) W(B)

T ₁	T ₂
R(A) W(A) R(B) W(B)	
	R(A) W(A) R(B) W(B)

T ₁	T ₂
	R(A) W(A) R(B) W(B)
	R(A) W(A) R(B) W(B)

 $R_2(B) \rightarrow W_1(B)$ $W_1(B) \rightarrow R_2(B)$ $S_1 \neq S_3$ $S_1 \not\sim S_2$ $\Rightarrow S_1$ is not conflict serializable.Note:-

A schedule that is conflict serializable must ensure consistency of the database.

Test for Conflict Serializability using precedence graph

① construct a directed graph where each vertex corresponds to a transaction and each directed edge represents a conflicting operation.

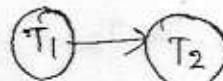
② If the directed graph contains cycles then the concurrent schedule is not conflict serializable.

③ If the graph contains no cycle then the schedule is called conflict serializable.

The serializability order is determined based on the directed edges.

Ex:-

T ₁	T ₂
R(A) W(A)	R(A) W(A)
R(B) W(B)	R(B) W(B)

 S_1 is CS $T_1 \rightarrow T_2$

T ₁	T ₂
R(A) W(A) R(B)	
	R(A) W(A) R(B) W(B)

 S_1 is not CS

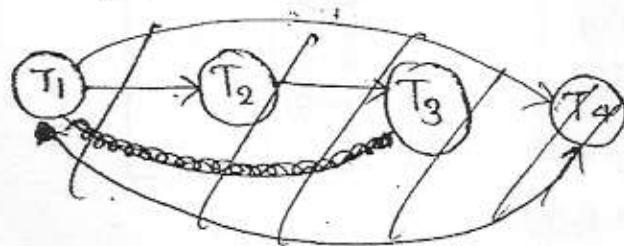
B

150

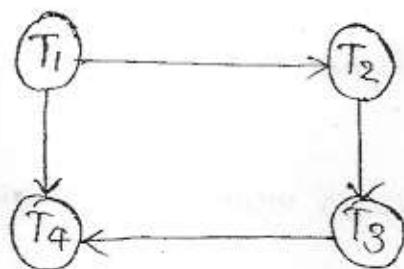
Ques:-

D) Find the serializability order of the concurrent schedule given below.

$R_1(x)$ $w_3(x)$ $w_4(y)$, $R_2(y)$ $w_2(z)$ $R_4(x)$ $R_4(y)$



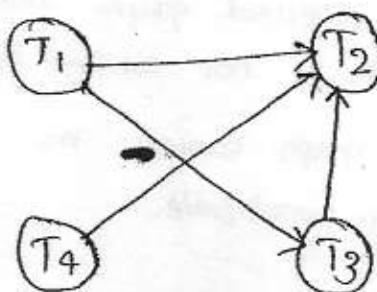
C.S $T_1 - T_2 - T_3 - T_4$



Q2

$R_1(A)$ $R_2(A)$ $R_3(A)$ $w_1(B)$ $w_2(A)$ $R_3(B)$ $w_2(B)$

- a) not C.S
- b) $T_3 T_4 T_1 T_2$
- c) $T_1 T_4 T_3 T_2$
- d) ~~$T_1 T_3 T_1 T_4$~~



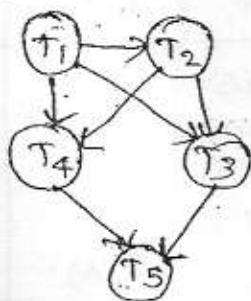
C.S = $T_1 T_4 T_3 T_2$

" $T_4 T_1 T_3 T_2$

$T_1 T_3 T_4 T_2$

Q) Consider the precedence graph given below.

T₁ T₂ T₃



Find the no. of possible serial schedules?

$$\begin{array}{c}
 T_1 - T_2 \\
 \swarrow \quad \searrow \\
 \text{----} \quad \text{----} \\
 T_4 - T_3 - T_5
 \end{array}
 \left. \right\} \text{G.S. = } \underline{\underline{2}}$$

⑧ View equivalent schedule

Two schedules S and S' are said to be view equivalent if the following 3-conditions are met. for each data item (say A).

- (i) For each data item A if transaction T_i reads the initial value A in schedule S then transaction T_i must in schedule S' also read the initial value of A.
- (ii) If transaction T_i executes read - RCA in schedule S . and that was produced by transaction T_j (if any) then transaction T_i must in schedule S' also read the value of A that was produced by T_j .
- (iii) For each data item the transaction (if any) that performs final write of A operation in S must also perform the final write of A operation in S' .

Note:- All write-Read Sequences to be maintained in the same order in both S and S' .

S_1		S_2	
T_1	T_2	T_1	T_2
R(A) W(A)		R(A)	
	R(A) W(A)	R(B) W(B)	
R(B) W(B)			R(A) W(A)
	R(B) W(B)	R(B) W(B)	

$S_1 \stackrel{v}{=} S_2$
 $\Rightarrow S_1$ is view
 serializable.

Ques

 S_1

T_1	T_2
R(A)	
W(A)	

S_2	
T_1	T_2
R(A) W(A)	
	W(A)

 $S_1 \neq S_2$

S_3	
T_1	T_2
R(A)	
W(A)	

 $S_2 \neq S_3$ $\Rightarrow S_1$ is not view serializable

Ques

 S_1 :

T_1	T_2	T_3
R(A)		
W(A)		W(A)

 S_2 :

T_1	T_2	T_3
R(A)		
W(A)	W(A)	W(A)

 $S_1 \stackrel{v}{=} S_2$ S_1 is view serializable.

View Serializable

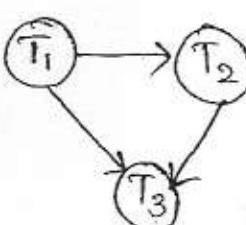
153

A schedule is view serializable if it is view equivalent to a serial sched

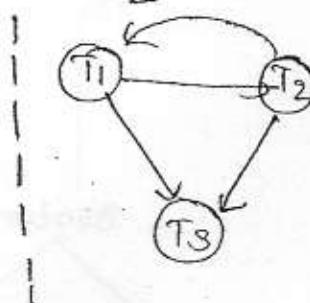
Polygraph

S_i	T_1	T_2	T_3
	$R(A)$		
		$W(A)$	
			$W(A)$

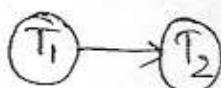
It is not C.S



V.S: $T_1 - T_2 - T_3$



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



It is ~~C.S~~ C.S and V.S also

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

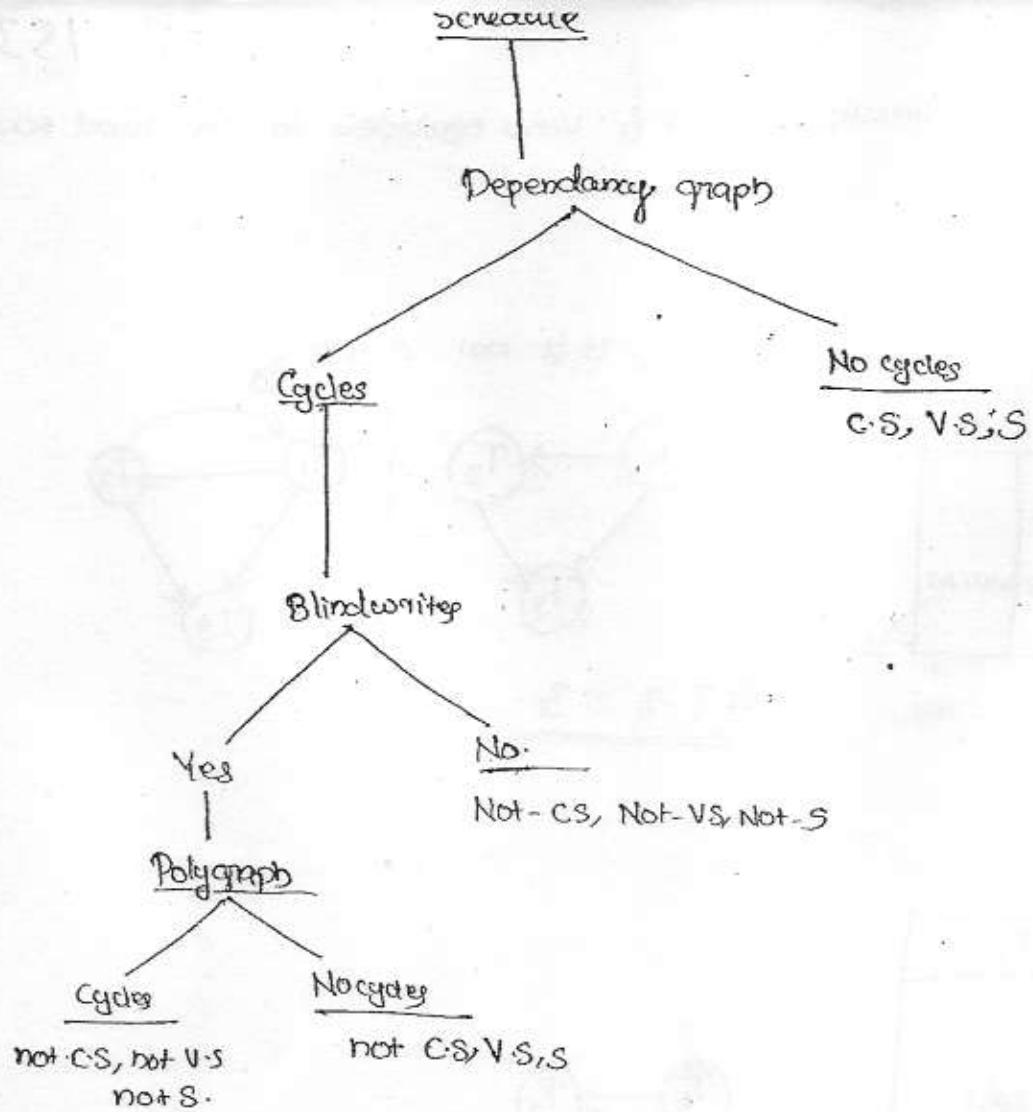


It is not C.S and
not V.S

Note: A schedule that is conflict serializable is also view serializable, but a view serializable schedule need not be conflict serializable.

Blind write: writes without read is called Blind write.

Note: A schedule is V.S but not C.S. There must be atleast one blind write operation.



Serializable schedules :- A schedule is serializable. If it is either conflict serializable or view serializable or both.

Q How many concurrent schedules can be formed over two transactions having two-two operations each?

<u>T₁</u>	<u>T₂</u>
R ₁ (A)	R ₂ (A)
W ₁ (A)	W ₂ (A)

- 1) R₁(A) W₁(A) R₂(A) W₂(A)
 - 2) R₁(A) R₂(A) W₁(A) W₂(A)
 - 3) R₁(A) R₂(A) W₂(A) W₁(A)
 - 4) R₂(A) W₂(A) R₁(A) W₁(A)
 - 5) R₂(A) R₁(A) W₁(A) W₂(A)
 - 6) R₂(A) R₁(A) W₂(A) W₁(A)
- non serial schedule

$$\# \text{ Concurrent schedules} = \underline{6} \longrightarrow \frac{(2+2)!}{2! \cdot 2!} = \frac{4!}{2! \cdot 2!} = \frac{24}{4} = \underline{6}$$

$$\# \text{ Non-serial schedules} = \underline{6-2} = \underline{4}$$

155

Note:-

The no. of concurrent schedules that can be formed over two transactions having n_1 and n_2 operations respectively are.

$$\frac{(n_1+n_2)!}{n_1! \cdot n_2!}$$

The no. of non-serial schedules are

$$= \left[\frac{(n_1+n_2)!}{n_1! \cdot n_2!} \right] - 2$$

Note:-

The no. of concurrent schedules that can be formed over m -transactions having n_1, n_2, \dots, n_m operations respectively are

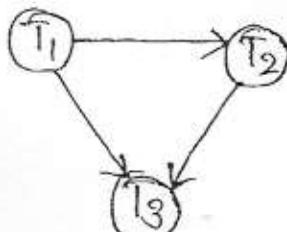
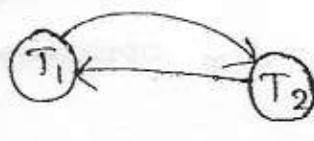
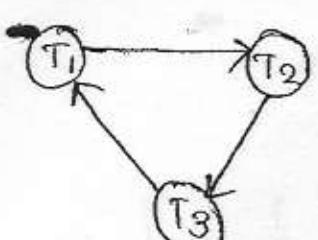
$$\left[\frac{(n_1+n_2+\dots+n_m)!}{n_1! \cdot n_2! \cdot n_3! \cdot \dots \cdot n_m!} \right]$$

The no. of non-serial schedules are

$$\left[\frac{(n_1+n_2+n_3+\dots+n_m)!}{n_1! \cdot n_2! \cdot n_3! \cdot \dots \cdot n_m!} \right] - m!$$

Q4 Determine whether the following schedules are conflict serializable or not?

(P.T.O)

S1: $R_1(A) W_1(A) R_2(B) W_2(B) R_1(B) W_1(B)$ S2: $R_1(A) R_1(B) W_2(A) R_3(A) W_1(B) W_3(A) R_2(B) W_2(B)$ S3: $R_1(A) R_2(A) R_1(B) R_2(B) R_3(B) W_1(A) W_2(B)$ S4: $W_3(A), R_1(A) W_1(B) R_2(B) W_2(C) R_3(C)$.AnsS1:  It is CS $(T_2 - T_1)$ S2:  If v.C.S $\in (T_1 - T_2 - T_3)$ S3:  It is not C.S
 (T_3) S4:  It is not C.SQ Two transactions T_1 and T_2 are given as follows Ref $T_1: R_1(A) W_1(A) R_1(B) W_1(B)$ $T_2: R_2(A) W_2(A) R_2(B) W_2(B)$ Find the total no. of conflict serializable schedules that can be formed by T_1 and T_2 ?

(P.T.O)

$T_1 \rightarrow T_2 : R_1(A) W_1(A) | R_1(B) W_1(B) | R_2(A) W_2(A) | R_2(B) W_2(B)$ - 6

157

$$\frac{(2+2)!}{2! * 2!} = 6$$

6 possibilities.

$T_2 \rightarrow T_1 : R_2(A) W_2(A) | R_2(B) W_2(B) | R_1(A) W_1(A) | R_1(B) W_1(B)$ - 6

$$\frac{(2+2)!}{2! * 2!} = 6$$

6 possibilities.

12

Q4 Two transactions T_1 and T_2 are given as follows

$T_1 : R_1(A) W_1(A) R_1(B) W_1(B)$

$T_2 : R_2(B) W_2(B) R_2(A) W_2(A)$

$T_1 \rightarrow T_2 : R_1(A) W_1(A) R_1(B) W_1(B) | R_2(B) W_2(B) R_2(A) W_2(A)$

$T_2 \rightarrow T_1 : R_2(B) W_2(B) R_2(A) W_2(A) | R_1(A) W_1(A) R_1(B) W_1(B)$

Q5 Consider the following schedules $S_1 : R_2(X) R_2(Y) R_1(X) R_1(Y) W_1(X) R_2(X)$

$S_2 : R_2(X) R_2(Y) W_2(X) R_1(X) R_1(Y)$

$S_3 : R_2(X) R_2(X) R_1(Y) R_1(Y) W_1(X) W_2(X)$

a) which of the above schedules having un-repeatable read problem.

b) which of the above schedules having lost update problem.

a) $S_1 : R_2(X) \dots W_1(X) R_2(X)$

b) $S_3 : \dots W_1(X) W_2(X)$

c) which of the above schedules having committed dirty read problem?

$S_2 : \dots W_2(X), R_1(X) \dots$

$S_1 : \dots W_1(X) R_2(X) \dots$

for the following schedules find whether the schedule is recoverable, cascadeless and strict?

158

1: $R_1(x) R_2(x) W_1(x) W_2(x) Commit_2(C_2) C_1(Commit 1)$

- Recoverable
- Cascadeless
- Not Strict

T_1	T_2
$R(x)$	
$w(x)$	
Commit	$R(x)$ $w(x)$ Commit

2: $R_1(x) W_2(x) W_1(x) R_1(x) Commit_2 Commit_1$

- Recoverable
- Cascadeless
- Not Strict

T_1	T_2
$R(x)$	
$w(x)$	$w(x)$
$R(x)$	Commit
Commit	

3: $R_3(x) R_1(x) R_2(x) w_1(y) R_2(y) w_2(x) C_3 C_1 C_2$

- Recoverable
- cascading
- Not Strict

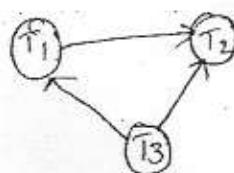
T_1	T_2	T_3
$R(x)$		$R(x)$
$w(y)$	$R(x)$	
Com	$R(y)$ $w(x)$	Com
		C_3

54:

T_1	T_2	T_3
$R(x)$		
$R(z)$	$R(x)$	$R(x)$
$w(x)$ Commit	$R(y)$	
	$w(z)$	
	$w(x)$	$w(y)$
	Commit	Commit

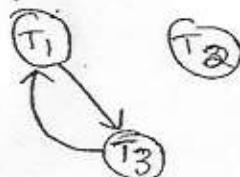
- Recoverable
- cascadeless
- Strict

(1)



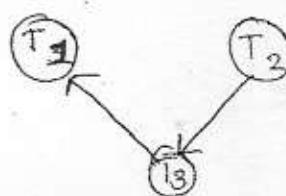
$T_3 - T_1 - T_2$ (C.S)

(2)



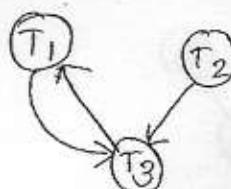
not C.S

(3)



$T_2 - T_3 - T_1$ (C.S)

(4)



not C.S

Ques (2)

T₁

R(X)
RC(Y)
W(X)

T₂

R(X)
RC(Y)
WC(X)
WC(Y)

a) WR=?

T ₁	T ₂
R(X) RC(Y) W(X)	
	R(X) RC(Y) WC(X) WC(Y)

c) WW=?

T ₁	T ₂
R(X) RC(Y)	R(X) RC(Y) WC(X) WC(Y)
WC(X)	

b) RW=?

T ₁	T ₂
R(X)	R(X) RC(Y) WC(X) WC(Y)
R(X) RC(Y) WC(X)	

T_1	T_2
$R(x)$	$R(x)$
$w(x)$	$w(x)$

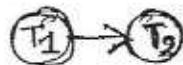
not - serializable
not - conflict serializable.
not - view serializable.



- Recoverable
- Cascadable
- Not Strict

b)

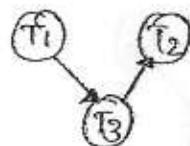
T_1	T_2
$w(x)$	$R(y)$
$R(y)$	$R(x)$



- C.S
- V.S
- S
- Recoverable
- Cascading aborts
- Not Strict

c)

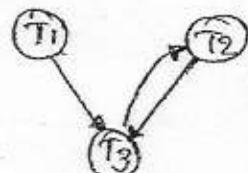
T_1	T_2	T_3
$R(x)$		
	$R(y)$	$w(x)$
	$R(x)$	



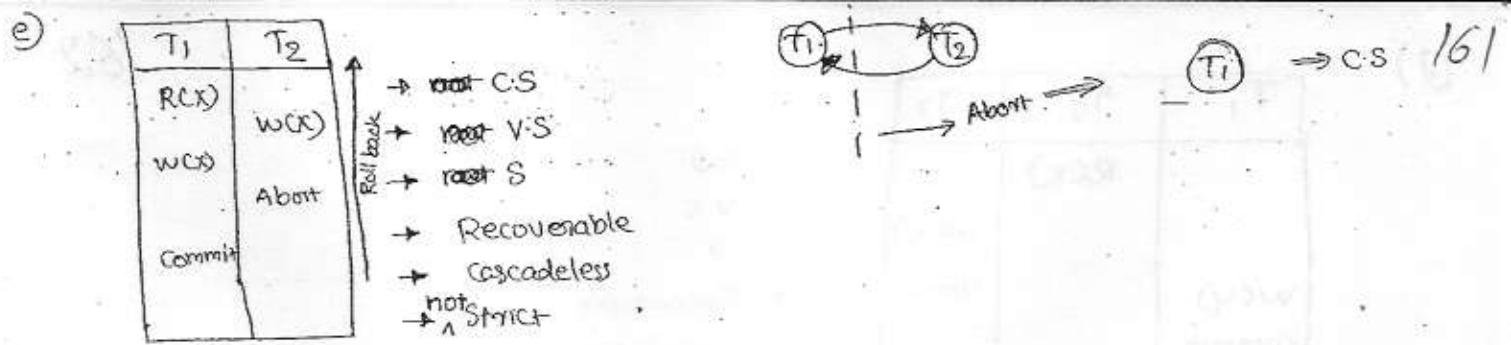
- C.S
- V.S
- S
- Recoverable
- Cascading aborts
- Not Strict

d)

T_1	T_2	T_3
$R(x)$		
$R(y)$		
$w(x)$		
	$R(y)$	$w(y)$
$w(x)$		
	$R(y)$	



- not C.S
- not V.S
- not S
- Recoverable.
- Cascading aborts.
- Not Strict



f)

T ₁	T ₂
R(X)	W(X)
W(X)	C
C	

- not CS
- not VS
- not S
- Recoverable
- cascadeless
- Not Strict

g)

T ₁	T ₂
W(X)	R(X)
W(X)	Aabort
Commit	

- C.S [aborted]
- not VS
- S
- Recoverable
- Cascading Abort
- Not Strict

h)

T ₁	T ₂
W(X)	R(X)
W(X)	
Commit	Commit

- not C.S
- not V.S
- not S
- not Recoverable
- Cascadeless
- not strict

i)

T ₁	T ₂
W(X)	R(X)
W(X)	
Aabort	Commit

- ✓ C.S ✓
- ✓ V.S ✓
- ✓ S ✓
- Not Recoverable
- cascadeless
- not strict

(j)

T ₁	T ₂	T ₃
W(x) Commit	R(x) R(y) W(z) Commit	W(x) Commit

- C.S
- V.S
- S
- Recoverable
- Cascadable
- Strict

T ₁	T ₂	T ₃
R(x)	W(x)	

- not CS
- not VS
- not S
- Recoverable
- ~~Cascading Aborts~~ Cascadable.
- Strict

(k)

T ₁	T ₂	T ₃
R(x)	W(x)	

- not CS
- not VS
- not S
- Recoverable
- Cascading Aborts
- Not Strict

(2-6-20)

Concurrency Control

D) Lock based protocols

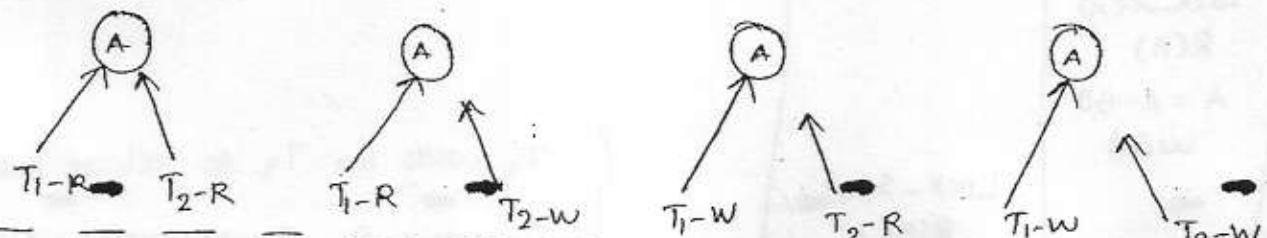
Which requires that all data items must be accessed in a mutually exclusive manner. i.e., when one transaction is accessing the data item no other transaction simultaneously update the data item.

To implement lock based protocols we need two locks.

1) Shared lock :- only Read (Lock-S)

2) Exclusive lock :- both Read & write (Lock-X)

$\begin{array}{c cc} S & X \\ \hline S & \checkmark & X \\ X & X & X \end{array}$	}	compatibility between lock models



T₁: A $\xrightarrow{\text{50}} B$

Lock-X(A)

Read(A)

A = A - 50

Write(A)

Unlock(A)

Lock-X(B)

Read(B)

B = B + 50

Write(B)

Unlock(B)

T₂: Display (B+A)

Lock-S(B)

Read(B)

Unlock(B)

Lock-S(A)

Read(A)

Unlock(A)

Display(B+A)

T_1	T_2	A	B	
Lock-X(A)		before T_1 : 1000	2000	$(A+B) = 3000$
Read(A)		After T_1 , 950	2050	is the only consistent data item.
$A = A - 50$				
write(A)				
Unlock(A)				
	Lock-S(B)			
	Read(B)			
	Unlock(B)			
	Lock-S(A)			
	Read(A)			
	Unlock(A)			
	Display(B+A)			
Lock-X(B)				
Read(B)				
$B = B + 50$				
write(B)				
Unlock(B)				

modified S1

T_1	T_2
Lock-X(A)	
R(A)	
$A = A - 50$	
w(A)	
	Lock-S(B)
	R(B)
	Lock-S(A)
	R(A)
	Display(B+A)
	Commit
	Unlock(B)
	Unlock(A)
Lock-X(B)	
R(B)	
$B = B + 50$	
w(B)	
Commit	
Unlock(B)	
Unlock(A)	

$\left\{ \begin{array}{l} T_1 \text{ waits for } T_2 \text{ to release lock on } B \\ T_2 \text{ waits for } T_1 \text{ to release lock on } A \end{array} \right\}$

\Rightarrow Causes Deadlock

Note:-

~~if we do not unlock a data item before requesting a lock on another data item. deadlocks may occur.~~

If we do not unlock a data item before requesting a lock on another data item. deadlocks may occur.

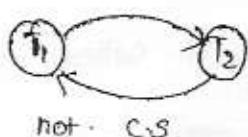
Q. Consider the following sequence and determine the serializability order.

165

T_1	T_2
L(A)	
U(A)	
	L(B)
	U(CB)
L(CB)	
U(B)	

$T_2 \rightarrow T_1$

T_1	T_2
L(A)	
U(A)	
	L(A)
	U(A)
	L(CB)
	U(CB)
L(CB)	
U(B)	
L(B)	
U(B)	



(3)

T_1	T_2	T_3
	L-SC(A)	
(L_P) L-X(B)	U(A)	L-SC(A) ----- (Lock point)
	V(B)	L-X(C)
L-S(B)		
(L_P) L-X(A)	U(A)	U(C)
U(B)		

It is conflict serializable

$$- \left[\begin{matrix} T_2 - T_3 - T_1 \\ T_3 - T_2 - T_1 \end{matrix} \right] -$$

2. Phase locking protocol

which require both locks and un-locks being done in 2-phases.

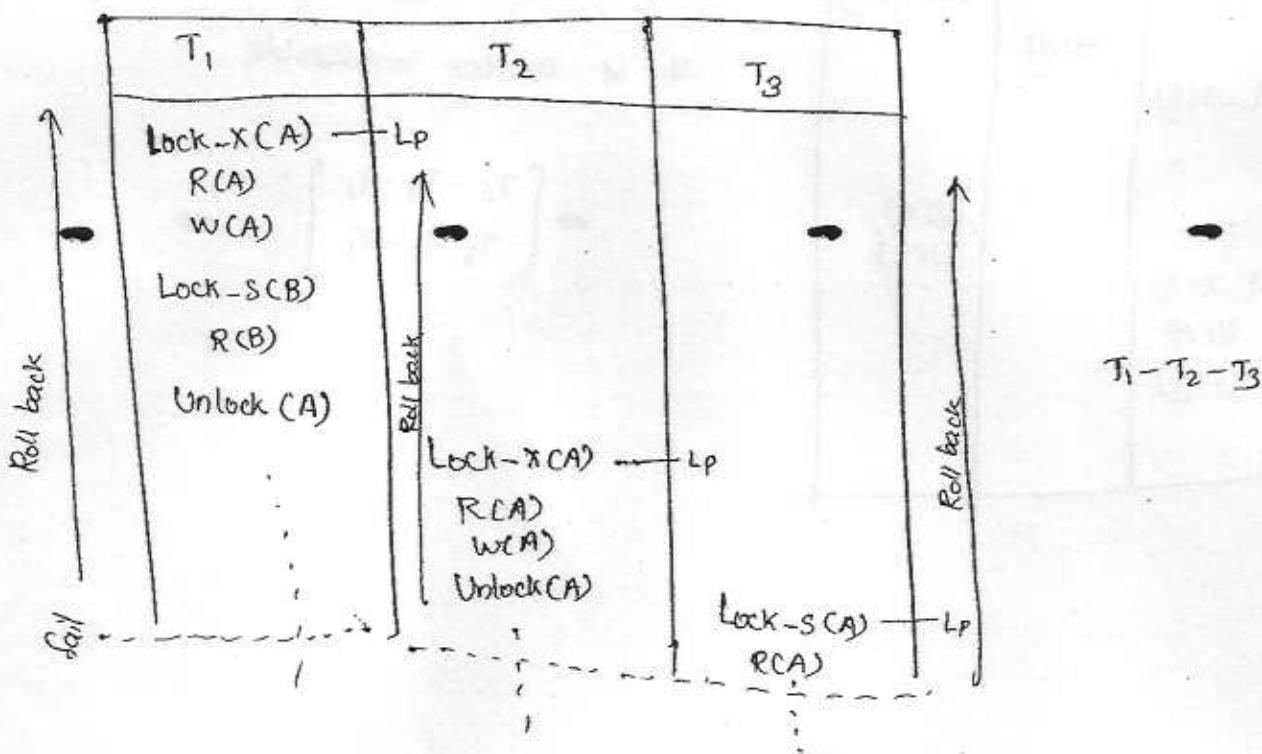
- 1) Growing Phase - "Obtain locks"
 - 2) shrinking phase. - "Release locks"

Lock point :-

The point at which the transaction has obtained its final lock is called, a lock point

- The lock point is used to determine the serializability order of a concurrent schedule

Two phase locking protocol ensures conflict serializability and serializability order is determined based on their lock points.



- Cascading rollbacks may occur under 2-phase locking protocols.

Deadlocks may occur under two phase schedules locking protocols.

167

Cascading rollbacks can be avoided by a modification of 2-phase locking protocols

(1) Strict 2-phase locking protocol (Strict 2PL) :-

which requires that in addition to locking being 2-phase all exclusive mode locks taken by a transaction must be held until the transaction commits.

(2) Rigorous 2-phase locking protocol (Rigorous 2PL) :-

which requires that in addition to locking being 2-phase all locks must be held until the transaction commits.

Avoids Cascading Rollbacks
but deadlocks can occur

(3) Conservative 2PL :-

which requires the transaction to update all the locks before it starts and release all the locks after it commits.

Avoids cascading Rollbacks
and deadlocks

Q. Which of the following schedules (Transactions) are in Strict 2PL

a) Lock-SCA
R(A)
Lock-X(B)
RCB)
Unlock(A)
W(B)
Unlock(B)

b) Lock-SCA,
R(A)
Lock-X(B)
Unlock(A)
RCB)
Unlock(B)
Commit

c) Lock-SCA
RCA
Lock-X(B)
RCB
Unlock(B)
Commit
Unlock(CD)

d) Lock-S(A)

Lock-X(B)

R(B)

W(B)

R(A)

Commit

Unlock(A)

Unlock(B)

e - nor 2PL

a, b, c, d → basic - 2PL

b, c, d → Strict 2PL

c, d → Rigorous 2PL

d → Conservative

e) Lock-S(A)

R(A)

Unlock(A)

Lock-X(B)

R(B)

W(B)

→ simple transaction with

Unlock(B)

Unlock(A)

Commit

T_1	T_2
Lock-S(Concurrent)	
Read(Concurrent)	
Lock-S(di)	
Read(di)	
asset = asset + di;	
	Lock-S(asset)
	Read(asset)
Lock-S(w ₁)	
Read(w ₁)	
asset = asset - w ₁	
Upgrade(asset)	
Write(asset)	
Commit	
Unlock(asset)	
:	

2- Lock Conversions

Upgrade → Shared to Exclusive

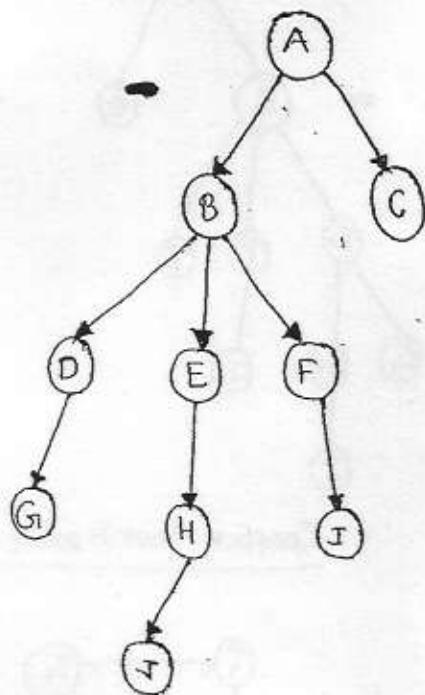
Downgrade → Exclusive to Shared

2) Graph based protocol

To implement graph based protocol we need additional information on how each transaction will access the database. The simplest model requiring that we have prior knowledge about the order in which the database items will be accessed. To acquire acquire such prior knowledge we impose partial ordering on set of all data items. ie, if $d_i \rightarrow d_j$ is an ordering any transaction which requires d_j must require to access d_i before d_j .

The partial ordering is represented as a directed acyclic graph called a database graph. The simplest protocol of graph based protocol is called "Tree protocol" which requires to employ only exclusive mode locks.

Ex:-



In tree protocol the only lock instruction allowed is lock-X and each transaction T_i can lock a data item atmost once and must observe the following rules.

170

- (1) The first lock by T_i may be on any data item.
- (2) Subsequently a data item can be locked by T_i only if the parent of the data item is currently locked by T_i .
- (3) Data items may be unlocked at any time.
- (4) A Data item that has been locked and unlocked by T_i can not subsequently relocked by T_i .

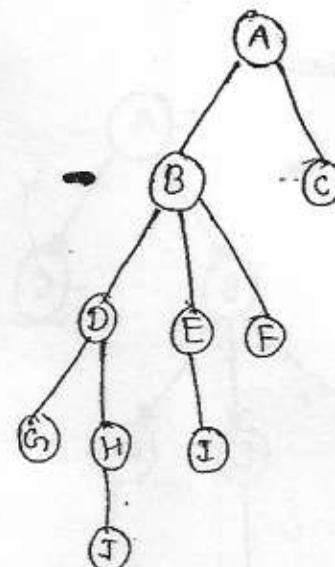
Note :-

All schedules that are legal under tree protocol are conflict serializable.

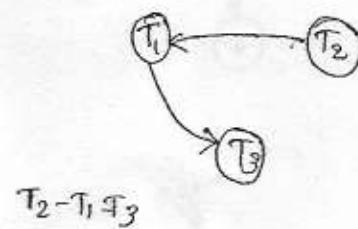
Ex:-

T_1	T_2	T_3
Lock-X(B)		
Lock-X(E) Lock-X(D) Unlock(B) Unlock(E)	Lock-X(C) Lock-X(H) Unlock(C)	
	Unlock(H)	Lock-X(B) Lock-X(E)
Lock-X(G) Unlock(D)		Unlock(E) Unlock(B)
Unlock(G)		

Tree of protocol



Conflict Serializable

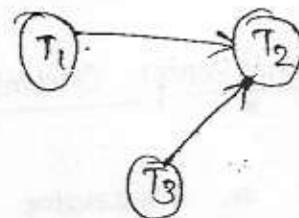
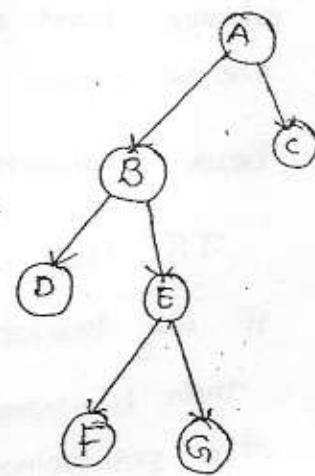


$T_2 - T_1, T_3$

Q4 Test whether the following schedule can occur under tree protocol? If possible give the serializability order.

171

T_1	T_2	T_3
L(A)		
L(B)		
L(D)		
U(B)		
	L(B)	
L(C)		
		L(E)
U(D)		
U(A)		
		L(G)
		(L(G))
		(U(G))
		U(G)
		U(E)
	L(E)	
	U(B)	
	U(E)	
U(C)		



$\left\{ \begin{array}{l} T_1, T_3, T_2 \\ T_3, T_1, T_2 \end{array} \right\}$ G.S

Advantages:-

- (i) Early unlocks causes increase in concurrency
- (ii) Ensures conflict serializability
- (iii) It is a deadlock free protocol

Drawbacks:-

- (i) Requires prior knowledge about the order of the data items
- (ii) Unnecessary locking overheads ie, In some cases it is required lock the data items that it does not access.

Which requires that ordering among the transactions is determined in advance, based on their timestamps. (ie, the time when ever it enters into the system for execution.)

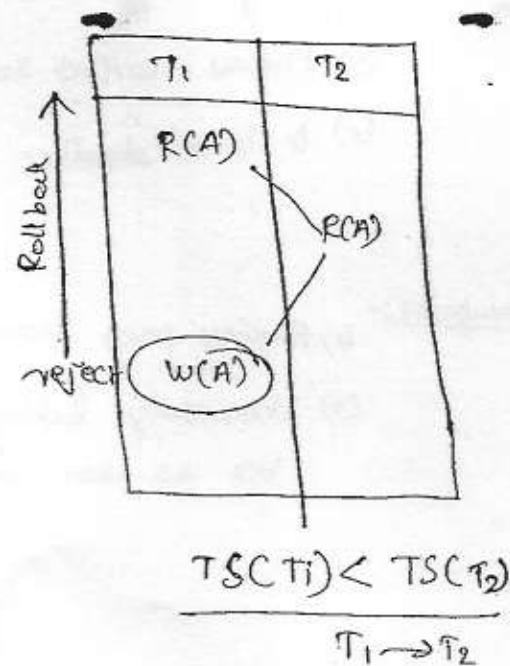
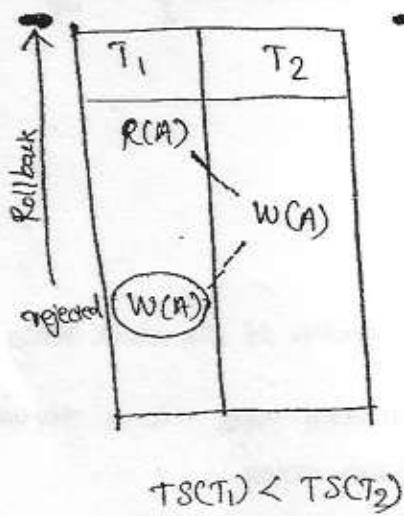
Each transaction is given unique fixed timestamp denoted by $TSC(T_i)$.

If any transaction T_j that enters after T_i , the relation between their timestamps be $TSC(T_i) < TSC(T_j)$ which means that the producing schedule must be equivalent to a serial schedule $T_i \rightarrow T_j$

(1) Timestamp Ordering protocol

This timestamp ordering protocol ensures that any conflicting read and write operations are executed in timestamp order if not such an operation is rejected and the transaction will be rolled back. The rolled back transaction will be restarted with a new timestamp.

Ex:-



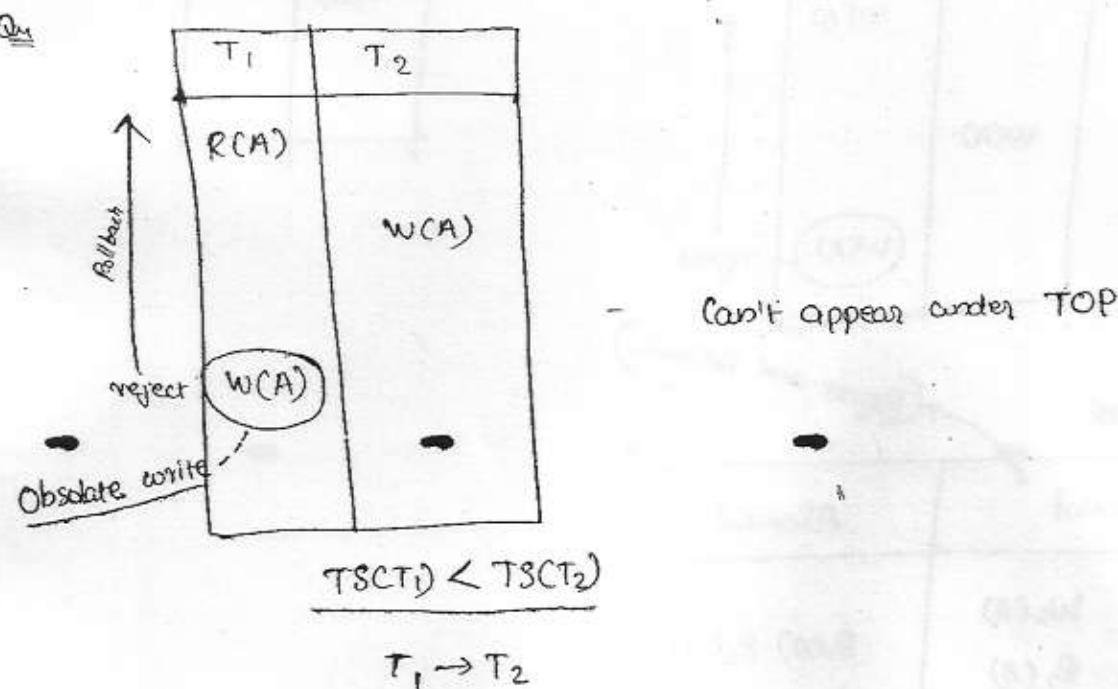
Q. Check if the following schedule can appear under timestamp ordering protocol.

173

T ₁	T ₂
R(B)	
	R(B) B = B - 50 W(B)
R(A)	
Display(A+B)	R(A)
	A = A + 50 W(A)
	Display(A+B)

It can appear under TOP

$$\frac{TSCT_1 < TSCT_2}{T_1 \rightarrow T_2}$$

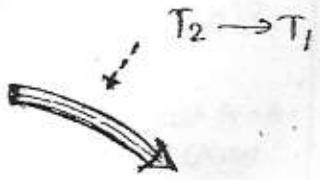


which requires to ignore all obsolete write operations

Ex:-

T_1	T_2
	R(A)
W(A)	W(A) (circled)

$$TSC(T_2) < TSC(T_1)$$



T_1	T_2
	R(A)
	W(A)

T_1	T_2
W(A)	W(A)
W(A)	W(A) (circled)

reject

(Reject and Rollback)

Thomas rule

	Not allowed	Allowed
TCP	R ₁ (A) W ₂ (A) W ₁ (A) R ₂ (A) W ₁ (A) W ₂ (A)	R(A) R ₂ (A)
TWR	R(A) W ₂ (A) W ₁ (A) R ₂ (A)	R ₁ (A) R ₂ (A) W ₁ (A) W ₂ (A)

① if $T_2 \rightarrow T_1$, then $R_2(A)$ must be present

if timestamp order and there is $R_1(A)$ is present

Q4 8: $w_1(x) \cdot w_2(x) \cdot w_3(x) \cdot R_2(x) \cdot w_4(x)$

Consider the schedule and the statement 1 is

Statement 1: "The above schedule is possible under timestamp ordering protocol"

Statement 2: The above Schedule is possible under Thomas write rule.

which of the above ~~schedules~~ statements are true about the Schedule given above?

~~Answer~~

T ₁	T ₂	T ₃	T ₄
w(x)	w(x) R(x)	w(x)	w(x)

1 - 2 - 3 - 4

According to
TOP - reject & Roll back

The above Schedule cannot appear under both timestamp ordering protocol and thomas write rule protocol.

T ₁	T ₂
R(B) R(A)	R(B) w(A)

T₁ → T₂

reject w₁(A) and roll back T₁ according to TOP
Ignore w₁(A) and continue with T₁ according to TWR

Advantage

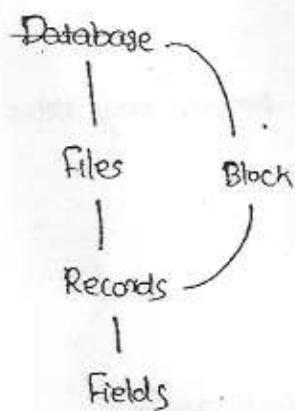
This protocol ensures

Adv

- This protocol ensures serializability (according to T.S)
- ensures freedom from deadlock

Disadv

- Starvation may occur due to continuously getting aborted and restarting the transaction.



Data base is a collection of files,
each file is a collection of records,
each record is a sequence of fields

Blocking factor

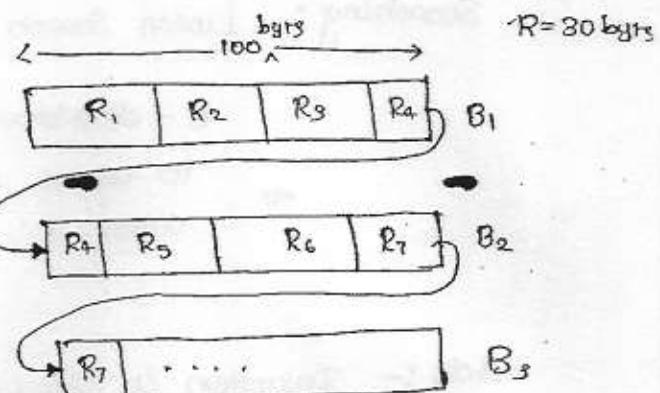
Blocking factor is the average no. of records per block.

Strategies for storing file of records into block

① Spanned strategy

It allows, partial part of record can be stored in a block

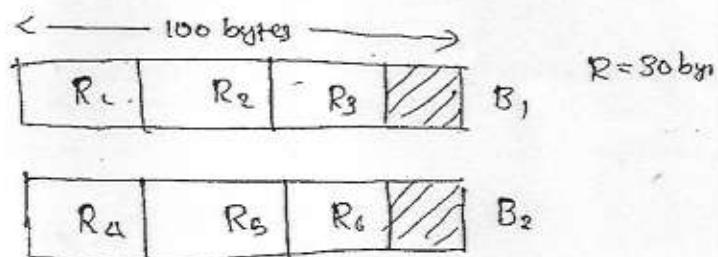
- Adv:- No wastage of memory
- Disadv:- Block accesses increases
- Suitable :- It is suitable for Variable length record.



② Un-spanned strategy

No record can be stored in more than 1-block.

- Disadvantage :- Wastage of memory
- Advantage :- Block accesses reduced.
- Suitable :- It is suitable for fixed length record.



1) Ordered file organization

All file of records are ordered based on some search key value

Searching :- Binary Search.

B - data blocks

To access a record, the average no. of block access

$$= \log_2 B \text{ blocks}$$

Adv :- Searching is efficient

Disadvantage :- Insertion is expensive due to re-organization of the entire file.

2) Un-ordered file organization

All file of records are inserted at wherever the place is available, usually at the end of the file.

Searching :- Linear Search

B - data block

To access a record, the average no. of block access

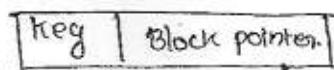
$$= \frac{B}{2} \text{ Blocks}$$

Adv :- Insertion is efficient

Disadv :- Searching is inefficient compared to ordered file organization.

Indexes are used to improve the searching efficiency.

Index is a record consists of two fields.



pointer to a block where 'key' is available.

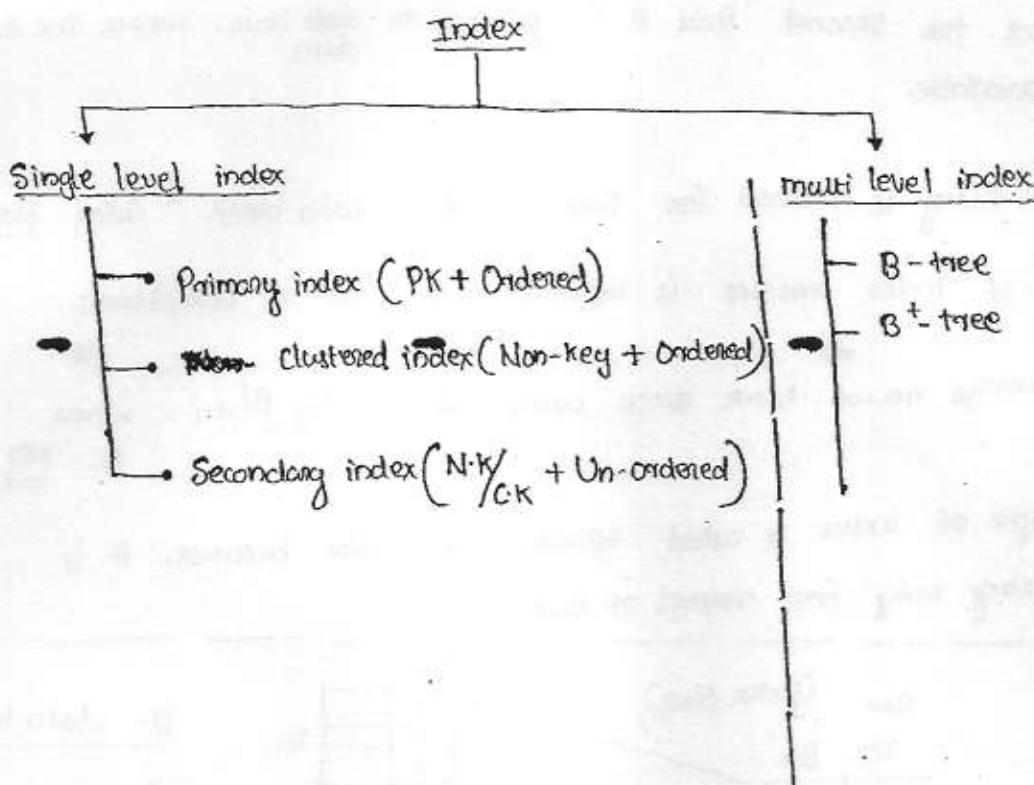
- ↳ Index is an ordered file
- ↳ Searching: Binary Search
- ↳ To access a record using index, the avg no. of block accesses.

$$= \log_2 B_i + 1$$

B_i = index block

Index block access Data block access

- ☒ Index can be created on any field of a relation, (primary key, non-key, candidate)



Classification of Indexes

180

1) Dense index

2) Sparse index

Dense index

If an index entry is created for every search key value that index is called dense index.

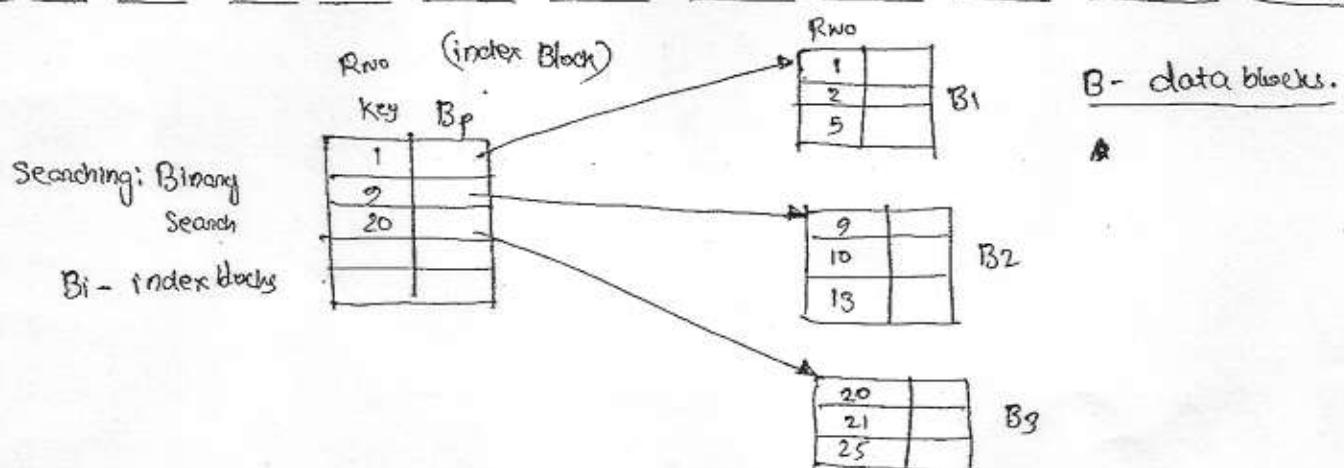
Sparse index

If an index entry is created only for some search key values it is called sparse index.

Primary Index (P.K + ordered)

A primary index is an ordered file whose records are of fixed length with two fields the first field is same as primary key of data file and the second field is a pointer to ~~data~~ block where the key is available.

- Index entry is created for first record of each block * called "block anchor"
- The no. of index entries is equal's to the no. of data blocks.
- The average no. of block access using index = $\log_2 B_i + 1$ where
 B_i - total index blocks.
- The type of index is called sparse block index because it is indexing only first record of each block



(Q) Suppose that we have an ordered file of 30,000 records stored on a disk. /8)

- (1) with block size 1024 Bytes. File records are of fixed length and are un-spanned of size 100 bytes and suppose that we have created a primary index on the key field of the file of size 9 bytes and a block pointer of size 6 bytes. Then find the avg. no. of blocks to search for a record using with and without index?

Ans

$$N = 30,000 \text{ ordered}$$

$$B = 1024 \text{ Bytes}$$

$$R = 100 \text{ bytes, fixed length, Un-spanned}$$

$$\begin{array}{r} 1024 \\ 1024 \\ \hline 6 \\ = 1024 \\ \hline 6 \end{array}$$

$$\text{Blocking factor} = \left\lfloor \frac{1024}{100} \right\rfloor = 10 \text{ records/block.}$$

$$\text{no. of data block} = \left\lceil \frac{30000}{10} \right\rceil = 3000 \text{ blocks}$$

$$\text{Avg. no. of block accesses} \left[\log_2 3000 \right] = 12 \text{ block accesses [without indexing]}$$

$$\begin{array}{r} 30 \\ 30 \\ \hline 6 \\ = 30 \\ \hline 6 \end{array}$$

$$\hookrightarrow \text{size of index blocks} = 9+6=15 \text{ bytes}$$

$$\hookrightarrow \text{no. of index records/blocks} = \left\lfloor \frac{1024}{15} \right\rfloor = 68$$

$$\hookrightarrow \text{no. of index records} = \text{no. of data blocks} = 3000$$

$$\text{no. of index blocks} = \left\lceil \frac{3000}{68} \right\rceil = 45$$

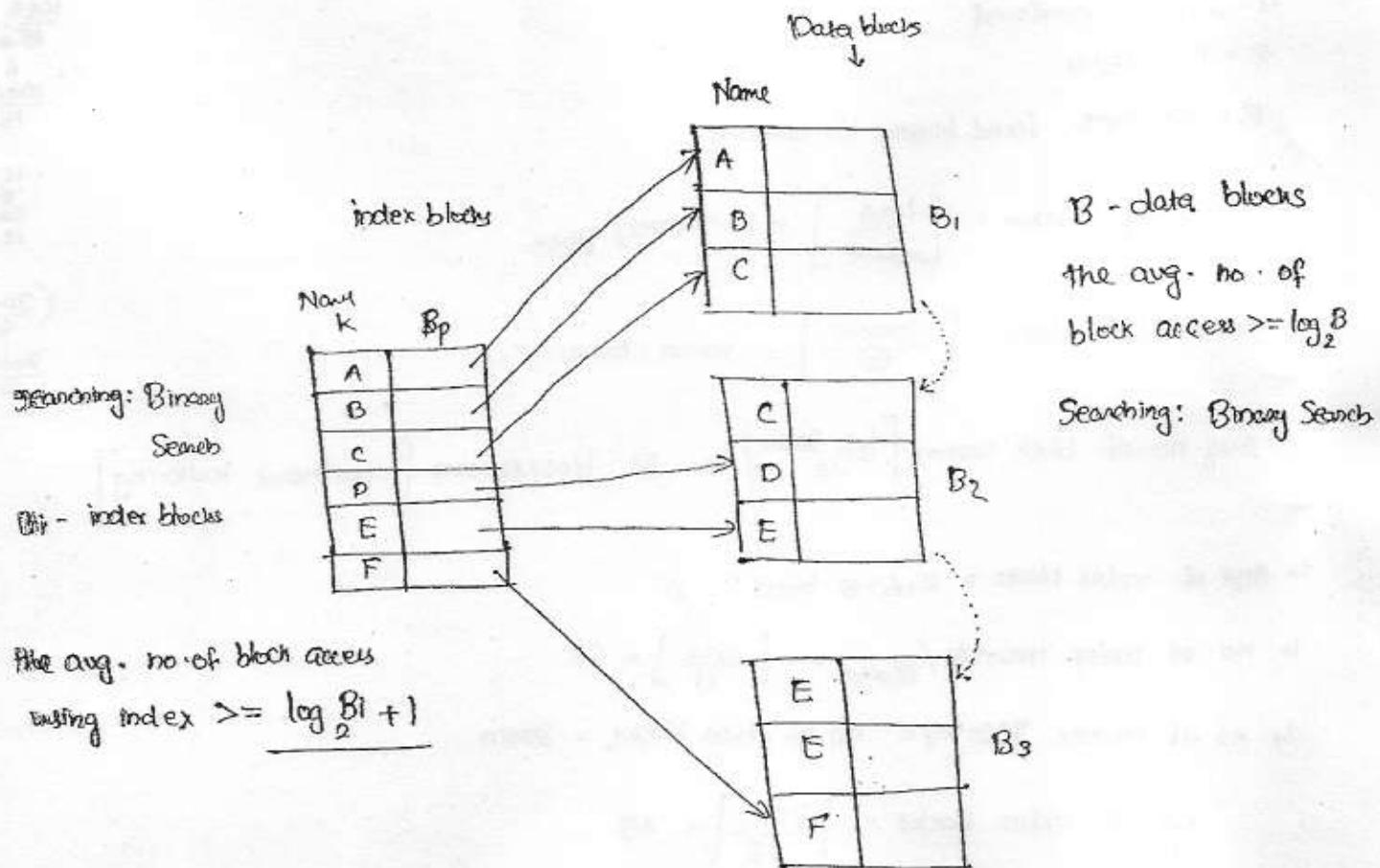
$$\text{Avg. no. of block accesses} = \left[\log_2 45 \right] + 1 = 6+1 = 7 \text{ block accesses [with indexing]}$$

Clustered Index (NK + Ordered)

clustered index is an ordered file with two fields, the first field is same as the clustering field is called non-key and the second field is a block pointer.

clustered index is created on data file whose file records are physically ordered on a non-key field which does not have a distinct value for each record that field is called clustering field.

- Index entry is created for distinct each distinct value of a clustering field.
- The block pointer points to first block in which the key is available.
- Type of index is Dense (Index entry is created for each ^{distinct} key value) / Sparse (index entry is not created for every record).

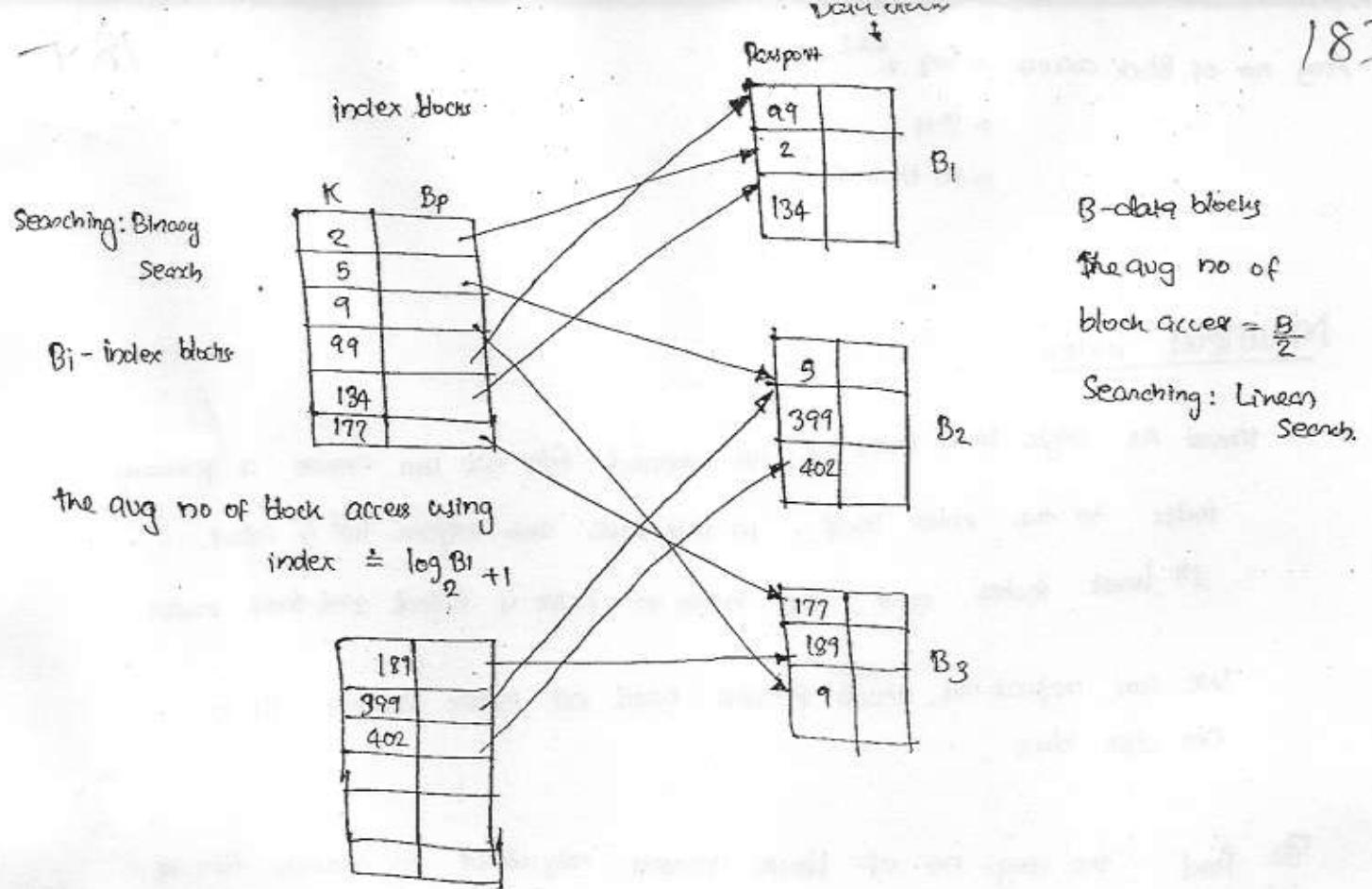


Secondary index (NK/C_K Un-ordered)

Secondary index provides a secondary means of accessing a file for which some primary access already exists.

Secondary index may be on a non-key or a candidate key.

- Index entry is created for each record in a data file.
- No. of index entries = No. of records.
- Type of secondary index is Dense.



Q Consider a secondary index is built on the key field of the file. of Question 1. then find avg no of block accesses to access a record using with and without index.

Ans

$$r = 30000, \text{ Un-ordered}$$

$$B = 1024 \text{ Bytes}$$

$$R = 100 \text{ bytes, fixed length, Unspaced}$$

$$k = 9 \text{ bytes, } B_p = 6 \text{ Bytes}$$

$$\text{Blocking factor } \left\lfloor \frac{1024}{100} \right\rfloor = 10 \text{ records/ block}$$

$$\text{no. of data blocks} = \left\lceil \frac{30000}{10} \right\rceil = 3000 \text{ blocks}$$

$$\text{the avg. no. of block access} = \frac{3000}{2} = 1500 \text{ block access}$$

$$\rightarrow \text{Size of index record} = 15 \text{ bytes}$$

$$\rightarrow \text{no. of index records/block} = \left\lfloor \frac{1024}{15} \right\rfloor = 68$$

$$\rightarrow \text{no. of index records} = 30000$$

$$\rightarrow \text{no. of index blocks} = \left\lceil \frac{30000}{68} \right\rceil = 442 \text{ index blocks}$$

B-data blocks

the avg no of

$$\text{block access} = \frac{B}{2}$$

Searching: Linear Search

$$\begin{aligned} \text{Avg no. of Block access} &= \log_2^{442+1} \\ &= 9+1 \\ &= 10 \text{ block access.} \end{aligned}$$

Multilevel index

Ans As single level index is an ordered file we can create a primary index to the index itself. In this case the original file is called 1st level index and, the index to index is called 2nd level index.

We can repeat the above process until all index entries fit in one disk block.

- Ques. 1 Find the avg. no. of block accesses required to search for a record if multi level index is created on the data file of

Ques. 2

Ans

242 index blocks

10 records

Blocking factor = 10 records / block

No. of data blocks = 3000

1st level

No. of index blocks = 442

2nd level

No. of index records = 442 (No. of 1st level blocks)

$$\text{No. of blocks} = \left\lceil \frac{442}{68} \right\rceil = 7$$

3rd level

No. of index records = 7 (No. of 2nd level blocks)

$$\text{No. of blocks} = \left\lceil \frac{7}{68} \right\rceil = 1$$

$$\text{Avg. no. of block access.} = 1 + 1 + 1 + 1$$



Note:- If there are n -levels in multilevel index the no. of block accesses to search for a record = $n+1$ (at each level one block and one data block) 185

Q Consider a file of 16,384 records each record is of size 32 bytes and key field is of size 6 bytes and the file organization is Un-spreaded and records are of fixed length stored on a disk with Block size 1024 bytes and the size of the block pointer is 10 bytes. If the secondary index is built on the key field of the file and multilevel index scheme is used the no. of 1st level and 2nd level blocks in multilevel index respectively are ?

- a) 810
- b) 12816
- c) 31212
- d) 25614

Ans

$$n = 16384$$

$$R = 32 \text{ Bytes} = 2^5$$

$$k = 6 \quad B_p = 10 \text{ Bytes}$$

$$B = 1024 \text{ bytes} = 2^{10}$$

$$\Rightarrow 2^5 \text{ records/Block}$$

$$\text{index record size} = 6 + 10 = 16 = 2^4$$

$$\Rightarrow \text{no. of index records per block} = 2^6 = \underline{64} \text{ records/Block}$$

1st level
no. of records = 16384

$$\text{no. of blocks} = \frac{16384}{64} = 256$$

2nd level
no. of records = 256

$$\text{no. of blocks} = 4$$

$$2+1+1$$

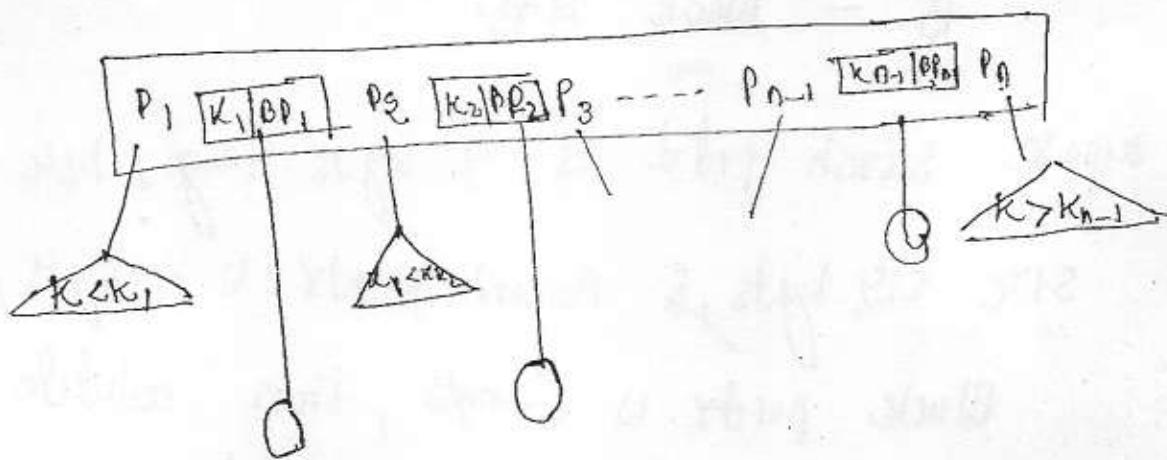
630
6-5
6-9-1

Q If one block access requires 30 ms. - 100 blocks index requiring ?

- a) 3000
- b) 4000
- c) 1800
- d) 210

$$(\log \frac{100}{2})$$

- B-tree is a balanced search tree.
- Node structure of B-tree corresponds to a block.
- Structure of a B-tree node



$K_1 < K_2 < K_3 \dots < K_{n-1} \leftarrow$ keys

$BP_1, BP_2, \dots, BP_{n-1} \leftarrow$ Record pointers / data pointers

$P_1, P_2, \dots, P_n \leftarrow$ tree pointer / block pointers

Order of a B-tree : the no of tree pointers
(let n) in. node.

$n \leftarrow$ tree pointers

$(n-1) \leftarrow$ keys & Record pointers

$\rightarrow [n * p] \leftarrow$ total size tree points

$$\rightarrow [n * p + (n-1)(k+p)] \leq B$$

p is size of tree pointer

$k+pr$ is size of index record

B - block size.

* suppose search field is 9-bytes long; index block

size 5 bytes, & Record pointer is 7-bytes,

Block pointer is 6-bytes, then calculate order
of b-tree node

sol. $k = 9, B = 512, pr = 7, p = 6$

$$n * 6 + (n-1)(9+7) \leq 512$$

$$(n + 1)(n - 1) \leq 512$$

$$2n^2 - 2n \leq 512$$

$$\underline{n \leq 24}$$

Q11 Suppose that we const, B-tree. on the Q1, calculate
 apparent no. of n-trees of L3 b-tree. Assume
 that each node 69% full. 188

$$\text{order} = 24 * 0.69 = 16$$

Root node 16 points 15 keys

$$L1 \quad 16 \text{ node } 16 \times 16 \text{ points} = 256 \quad 16 \times 15 = 240$$

$$L2 \quad 256 \text{ node } 16 \times 256 \quad 256 \times 15 = 3840 \\ = 4096$$

$$L3 \quad 4096 \text{ node } 4096 \times 16 \quad 4096 \times 15 \\ = 65,536 \quad = 61,440$$

$$\text{Total no internal cntrs} = 15 + 240 + 3840 + 61,440$$

$$= \underline{\underline{65,535}}$$

$$\boxed{\text{Total no internal cntrs} = 65,535}$$

Q consider a table T, in relation DB, with key field K. A B-tree of order p is used as an access structure, on K. where p denotes max no. of tree points in B-tree index node. Assume that K is 10 bytes

disk block size 512 bytes, each data pointer
 is 8 bytes. & each block pointer is 5 bytes
 in order for each B-tree node, to fit
 in 1 disk block. the max value of p
 is 10

$$\underline{sd} \quad k = 10, B = 512, p_t = 8$$

$$n = n * 5 + (n-1)(10+8) \leq \underline{512}$$

$$5n + 18n - 18 \leq 512$$

$$23n \leq 536$$

$$n \leq 23$$

* insertion into a B-tree node:

→ if order is n

max: n tree pointers — $(n-1)$ keys

min: $\left\lceil \frac{n}{2} \right\rceil$ tree pointers } exception for
 $\left\lceil \frac{n}{2} \right\rceil - 1$, keys } root & leaf of
 min or max

Eg:order: 5

max: 5 - tree per

4 - keys

MIN: 3 - tree per

2 - keys

→ new element always instead of leaf node.

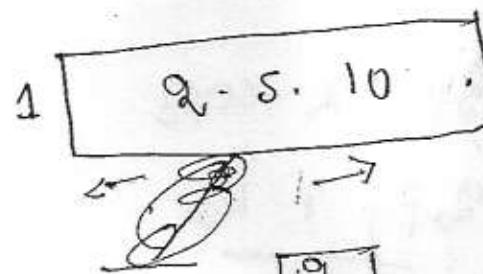
when node is full then split the node

like 2-nodes moving the middle element
to one higher level and then insert

The elest at the proper place

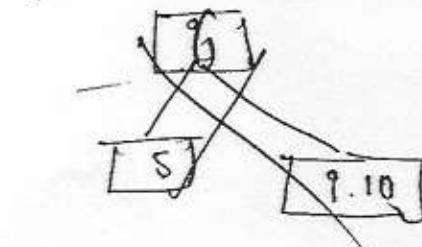
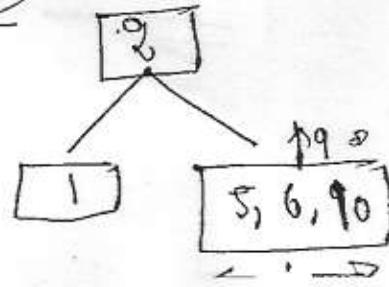
Q11 Insert the following keys in B-tree of order-4

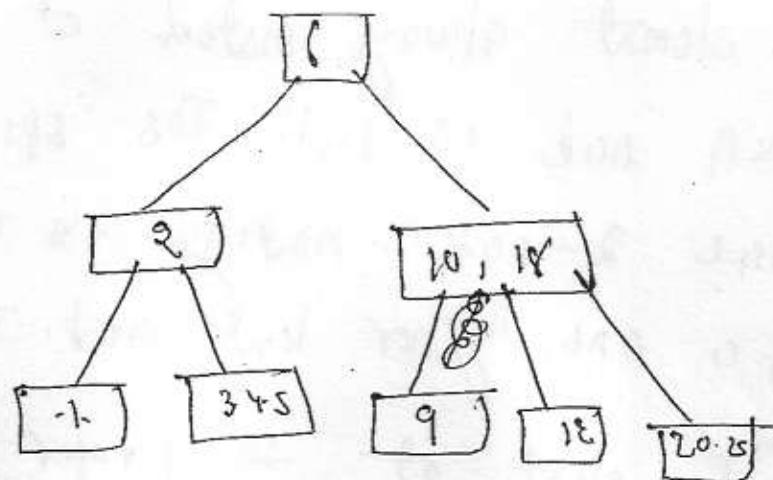
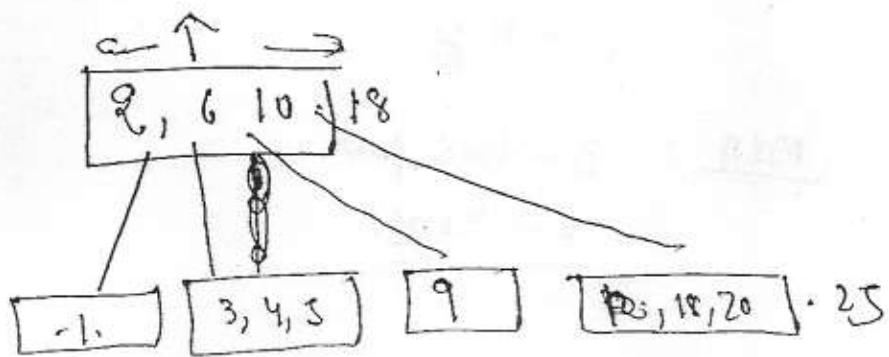
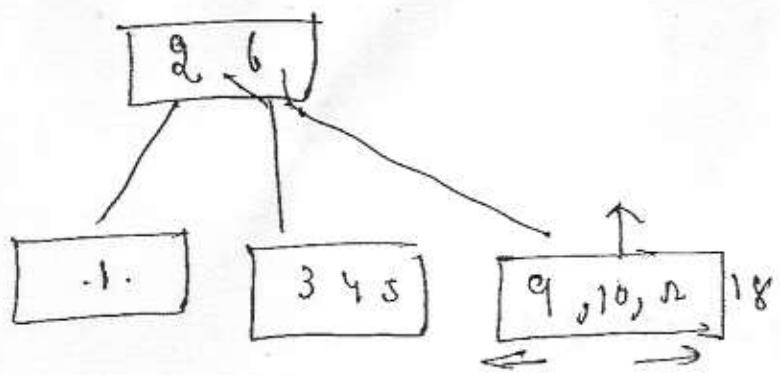
keys: 2, 5, 10, 11, 6, 9, 4, 3, 12, 18, 20, 25



max: 4P, 5K

MIN: 2P, 1K



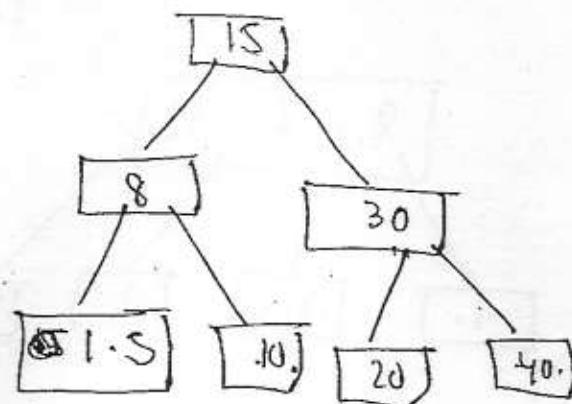
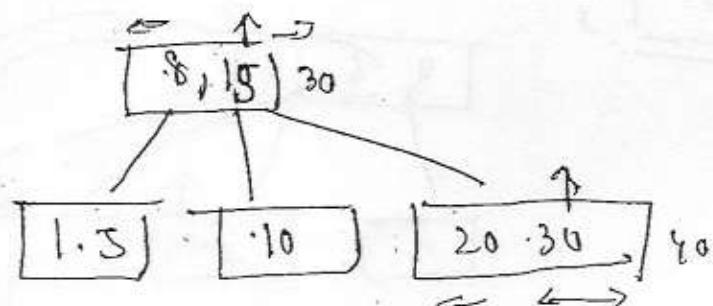
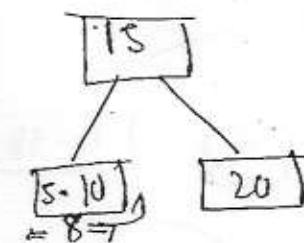
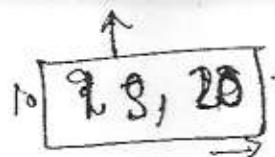


Q. Construct B-tree of order 3

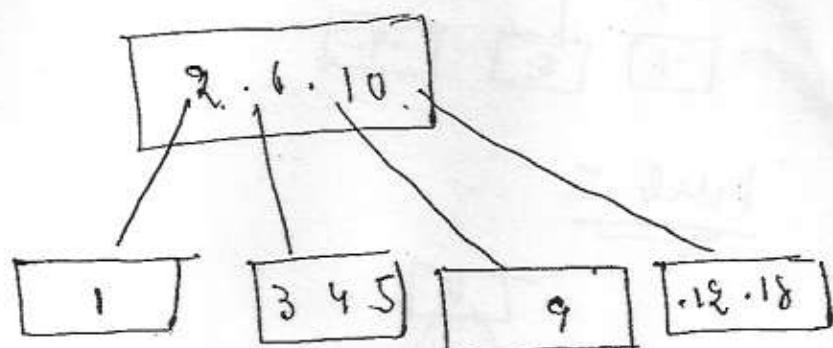
for key 20, 15, 10, 5, 8, 30, 1, 40

MARQ: 3 p, 2 keys

MIN: 2 p, 1 key



* Deletion

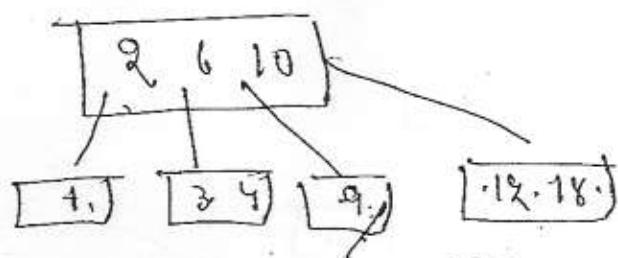


ORDER : 4

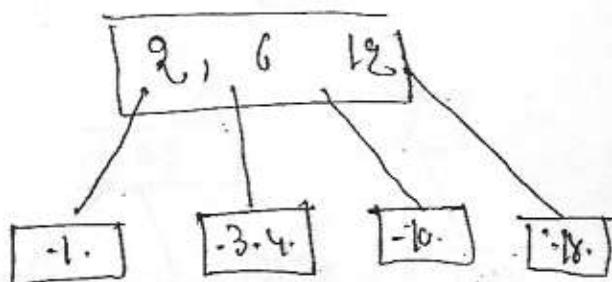
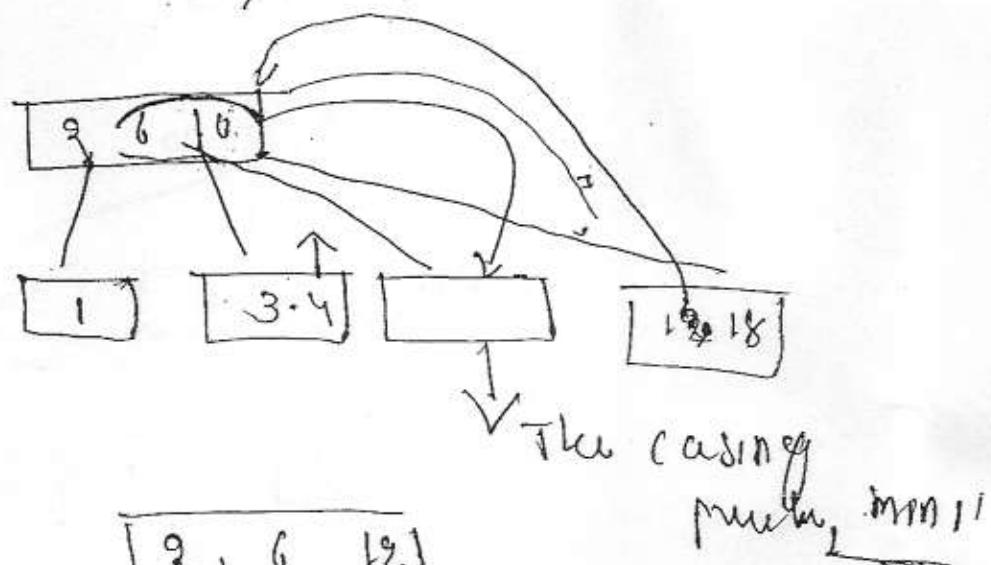
max = 4 position 3 keys

min = 2 n 1 "

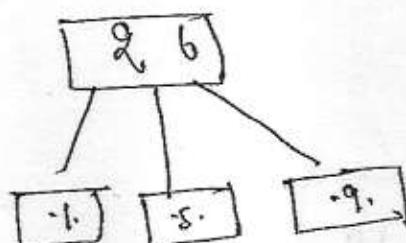
→ Jelolve 5:



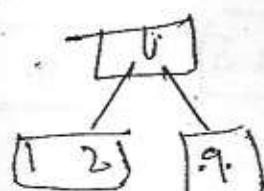
→ Je niet 9.;



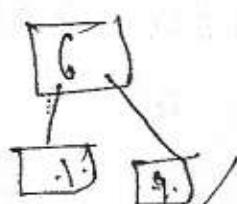
~~8~~



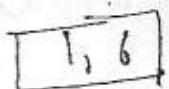
Drie 5:



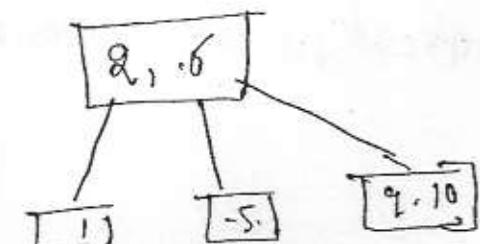
~~Wtq~~ -



• Deletion

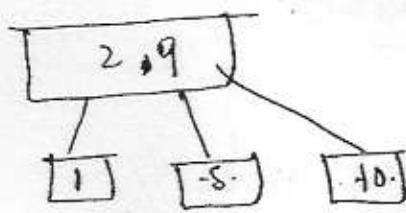
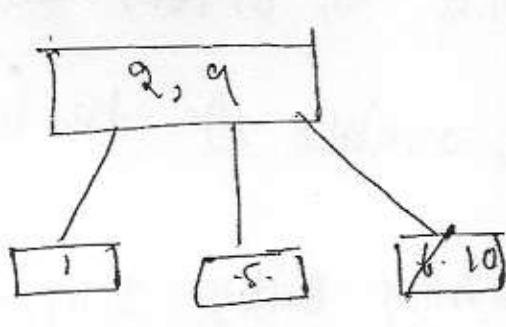


6g :-



D6

D2



B-tree provides direct access, i.e. if key to search is found at sum level, it directly access the data block where the key is available, by passing all the lower levels of index.

* B⁺-trees

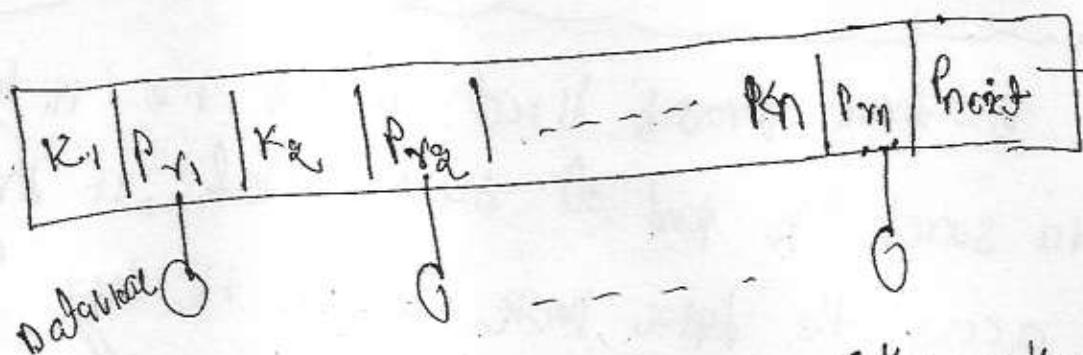
→ from B-tree leaf node, locate record points.

→ from B-tree internal node, remove all record pointers.

→ B⁺-tree provides an ordered access i.e. all keys are available at the leaf level).

i.e. T searching using single level in tree is possible.

* Structure of Leaf Node



$K_1 < K_2 < \dots < K_m \leftarrow \text{Keys}$

$P_{11}, P_{12}, \dots, P_{1m} \leftarrow \text{Record points}$

Point \leftrightarrow pointer to its sibling/tree pointer

order of leaf node:

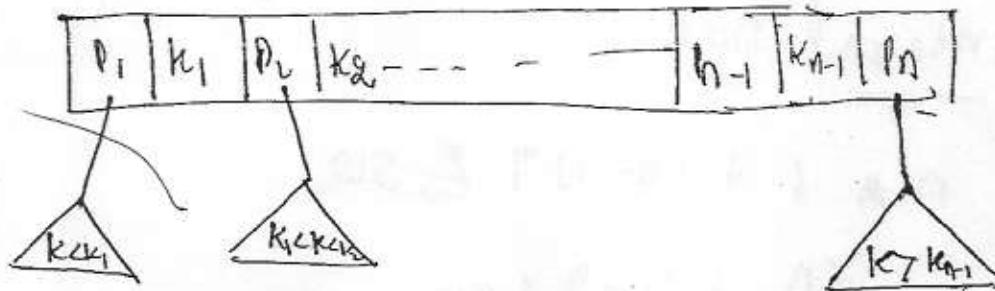
→ no of $(K_0, \text{ record pointer})$ pairs

$n \leftarrow (K_0, \text{ record pointer})$

$l \leftarrow \text{tree pointer}$

$$n(k+p) + pl \leq B$$

* structure of internal node:



$k_1 < k_2 < \dots < k_{n-1}$ keys

$p_1 \ p_2 \ \dots \ p_n$ tree pointers

order of internal node (let n):

no. of tree pointers

$$n * p + (n-1)k \leq B$$

B+ tree allows duplicates B-tree not
contains duplicates

calculate, order of the B-tree node suppose

If the search key field is 9 Bytes long,
 Block size is 512 bytes, record pointer is
 7 Bytes, so block pointer = 6 Bytes.

$$k = 9, B = 512, P_r = 7, P = 6$$

order of internal node

$$n * 6 + (n-1)9 \leq 512$$

$$6n + 9n - 9 \leq 512$$

$$15n \leq 521$$

$$n \leq 34$$

order of leaf node

$$n(9+7) + 6 \leq 512$$

$$16n \leq 506$$

$$n \leq \underline{31}$$

Q2 calculate the approximate no. of nodes in
 a B⁺ tree of level -3 of order 1 suppose
 that if each B⁺ tree node is 69% full.

$$\text{order of internal node} = \frac{34}{\underline{\quad}} + 0.09 = 23$$

$$\text{leaf node} = 31 * 0.69 = 21$$

<u>Root</u>	: <u>1 node</u>	<u>23 parents</u>
<u>level 1</u>	23 nodes	$23 \times 23 = 529$
<u>level 2</u>	529 nodes	$23 \times 529 = 12,167$ parents
<u>leaf level</u>	12,167 nodes	$21 \times 12,167 = 2,55,507$ nodes

Q. A B⁺-tree of size 1024 is to be built, on the
 name attributes of the Relation student
 assume that all student names are of
 length 8 bytes. Disk blocks are
 of size 512 bytes & tree pointers
 are of size 4 bytes, given
 this somewhat number be the no.
 of nodes to agree on the answer

$$K=8, B=512, P=4$$

$$n*4 + (n-1)*8 \leq 512$$

$$4n + 8n - 8 \leq 512$$

$$12n \leq 520$$

$$n \leq \underline{43}$$

Q11 The order of a leaf node in a B^+ -tree, max

No of key, data record pairs

it can hold. given that block size
1 kb, Data record point 7 bytes. The
value field is 9 bytes. & block pairs
is 6 bytes.

'what is order of leaf node'

$$B = 1K, P = 7, K = 9, l = 6$$

$$n(9+7) + 6 \leq 108$$

$$16n \leq 108$$

$$n \leq \underline{63}$$

B⁺-tree insertion.

200

Note: "we assume that the order of root node & internal node is same. But practically it is different"

order: n

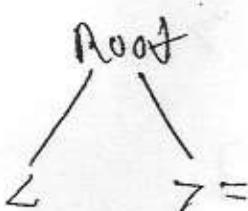
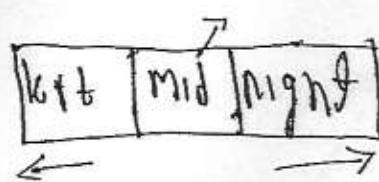
max: n points

n-1 keys

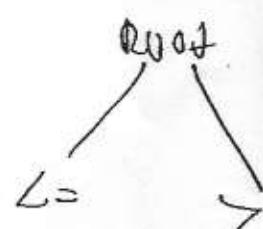
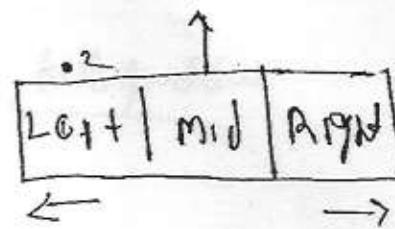
min: $\lceil nh \rceil$ points

$\lceil nh \rceil - 1$ keys

Split of leaf node



cor



Note : splitting a leaf node require to maintain
201

duplicate - a copy of the middle key.

but internal node split does not

you get the ~~key~~ ~~key~~ to maintain

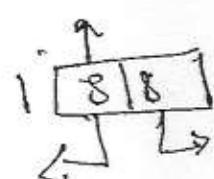
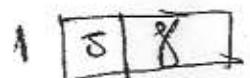
∴ insertion of say 6 keys 8, 5, 1, 7, 3, 12,

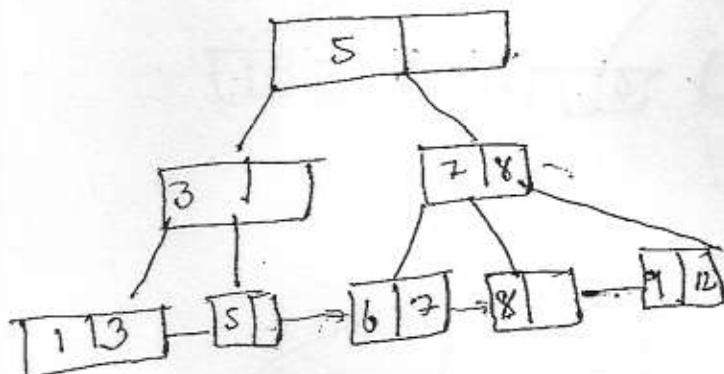
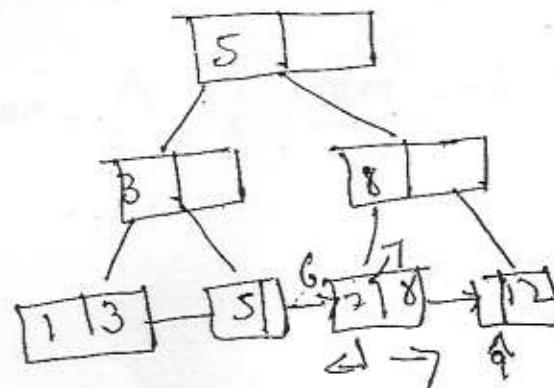
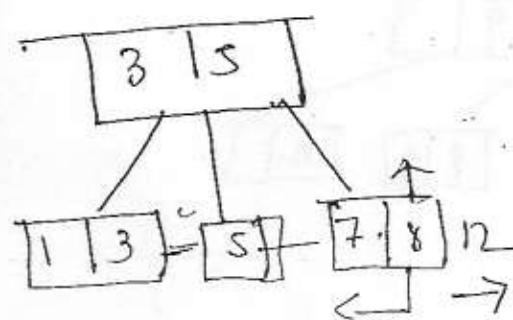
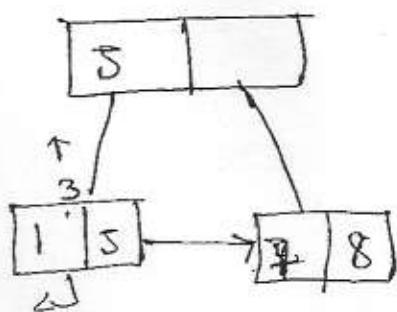
9, 6 in a BT tree of order 3.

SJM

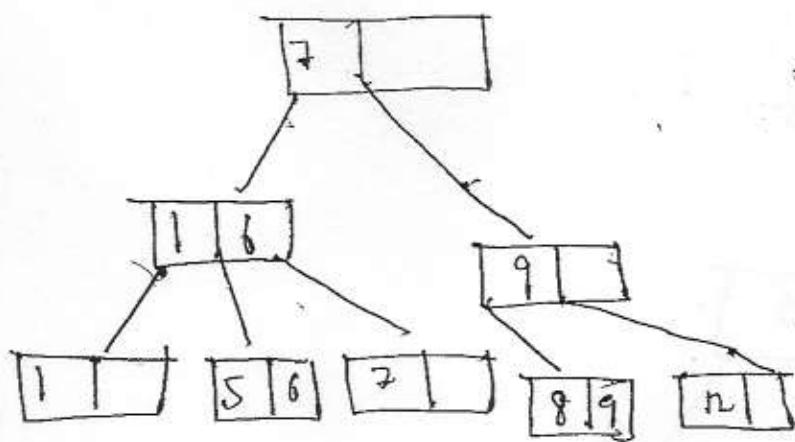
mag: 3 p min = 2 p
2 k 1 k

assumption



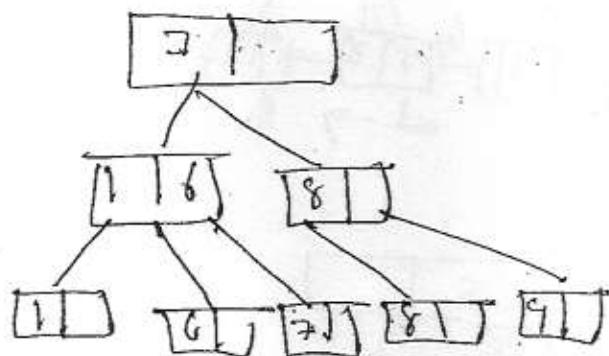


Deletion

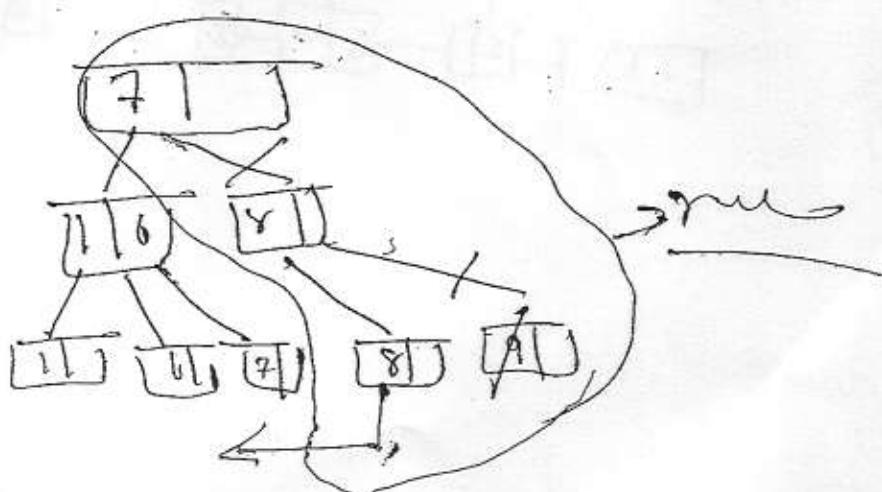


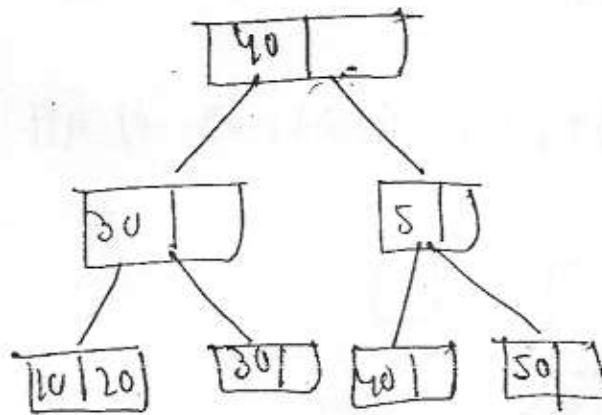
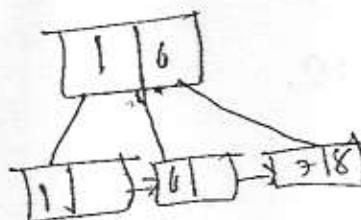
Del 5, 12, 9 m say

\rightarrow 05, 012, 11 m⁰ m m, \wedge \leftarrow 7 - max



D9 \rightarrow +m k





I : insert 15, 25

II : delete 50.

all key is, is data in the order, who may
be the ^{the} next add post in the tree after two
instructions.

(a) 1 11 2 43 11 4

II delete 50 :

, now the key 15 is deleted from
the other compare the tall status

S1: height is same

S2: $\boxed{20}$

S3: root becomes unchanged

which of the given statement is true

- A) S1, S2 B) S2, S3 C) S1, S3 D) None

