



DBMS - Part I

Complete Course on CS - Unit IV: Database Management Systems

Database Management System

Most Important Topics of DBMS through Previous Years Analysis

Previous Year	Total No of Questions	Topics
Dec 2019	10	Functional Dependency and Normalization – 6 Question ER Diagram – 1 Question SQL – 1 Question B Tree – 1 Question Relational Algebra – 1 Question
June 2019	9	Functional Dependency and Normalization – 1 Question Basic Concept of RDBMS- 1 Question Concept of Key – 2 Question Referential Integrity – 1 Question Relational Algebra and Relational Calculus – 2 Question Big Data Hadoop – 1 Question SQL- 1 Question

Previous Years Analysis of DBMS

Previous Year	Total No of Questions	Topics
Dec 2018	8	Functional Dependency and Normalization – 1 Question SQL – 3 Question Data Warehouse – 2 Question Indexing - 1 Question Transaction and Concurrency – 1 Question

Important Topics of DBMS to Crack NET JRF

Important Topics of DBMS	Total No of Question in NTA pattern
Functional Dependency and Normalization	8
SQL	5
Relational Algebra and Relational Calculus	3
Concept of Key	2
Referential Integrity	1
ER Diagram	1
Basic Concept of RDBMS	1
Transaction and Concurrency	1
Indexing	1
B Tree	1
Big Data Hadoop	1
Data Warehouse	2

Functional dependency

Database Management System

Relational Data Model in DBMS:

What is Relational Model?

RELATIONAL MODEL (RM) represents the database as a collection of relations. A relation is nothing but a table of values. Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.

The table name and column names are helpful to interpret the meaning of values in each row. The data are represented as a set of relations. In the relational model, data are stored as tables. However, the physical storage of the data is independent of the way the data are logically organized.

Some popular Relational Database management systems are:

- DB2 and Informix Dynamic Server - IBM
- Oracle and RDB – Oracle
- SQL Server and Access - Microsoft

Relational Model Concepts

1.Attribute: Each column in a Table. Attributes are the properties which define a relation. e.g., Student_Rollno, NAME,etc.

2.Tables – In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.

3.Tuple – It is nothing but a single row of a table, which contains a single record.

4.Relation Schema: A relation schema represents the name of the relation with its attributes.

5.Degree: The total number of attributes which in the relation is called the degree of the relation.

6.Cardinality: Total number of rows present in the Table.

7.Column: The column represents the set of values for a specific attribute.

8.Relation instance – Relation instance is a finite set of tuples (also known as rows or records)in the RDBMS system. Relation instances never have duplicate tuples.

9.Relation key - Every row has one, two or multiple attributes, which is called relation key.

10.Attribute domain – Every attribute has some pre-defined value and scope which is known as attribute domain

Table also called Relation

Primary Key

Domain
Ex: NOT NULL

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Column OR Attributes
Total # of column is **Degree**

Tuple OR Row
Total # of rows is **Cardinality**

The diagram shows a table with three columns: CustomerID, CustomerName, and Status. The CustomerID column is highlighted in yellow and labeled as the Primary Key. The CustomerName column is labeled as having a Domain of NOT NULL. The table contains three rows of data. The first row is highlighted in yellow and labeled as a Tuple OR Row. The total number of rows is labeled as Cardinality, and the total number of columns is labeled as Degree.

Functional dependency in DBMS

What is a Functional Dependency?

Functional Dependency (FD) determines the relation of one attribute to another attribute in a database management system (DBMS) system. Functional dependency helps you to maintain the quality of data in the database.

The attributes of a table is said to be dependent on each other when an attribute of a table uniquely identifies another attribute of the same table.

A functional dependency $A \rightarrow B$ in a relation holds if two tuples of attribute **A have same value** then for attribute **B also have same value** in that two tuples.

If column A of a table uniquely identifies the column B of same table then it can be represented as $A \rightarrow B$

Functional Dependency

$A \rightarrow B$

B - functionally dependent on **A**

A - determinant set

B - dependent attribute

How to Determine Functional Dependency ??

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d2
a3	b3	c2	d2
a4	b4	c4	d4
a5	b3	c3	d3

A \rightarrow B holds or not?

A \rightarrow C holds or not?

A \rightarrow D holds or not?

B \rightarrow C holds or not?

C \rightarrow A holds or not?

D \rightarrow A holds or not?

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d1
a2	b2	c1	d2
a3	b3	c2	d2
a4	b4	c4	d4
a5	b3	c3	d3

A -> B ❌

(a1 there is b1 or b2 so Confusing) Its does not holds

A->C

(a1-> c1 or c2) Its does not holds

A->D

(a1->d1 ✓ily, So unique) Its Holds

B->C

(b2 ->c2 ❌c1) Its does not holds

C->A

(c1->a1 ❌ a2) Its does not holds

D->A

(D2->a2 or a3) Its does not holds



Consider a Relation R(ABC) having the tuples.

A	B	C
1	2	3
4	2	3
5	3	3

Options:

(a) $A \rightarrow B$

(b) $BC \rightarrow A$

(c) $B \rightarrow C$

(d) $AC \rightarrow B$

Which of the functional dependency does not hold over R?

Consider a Relation R(ABC) having the tuples.

A	B	C
1	2	3
4	2	3
5	3	3

$A \rightarrow B$



$BC \rightarrow A$



$B \rightarrow C$



$AC \rightarrow B$



Which of the functional dependency does not hold over R?

The functional dependency is a relationship that exists between two attributes. It typically exists between the primary key and non-key attribute within a table.

$$X \rightarrow Y$$

The left side of FD is known as a determinant, the right side of the production is known as a dependent.

For example:

Assume we have an employee table with attributes: Emp_Id, Emp_Name, Emp_Address.

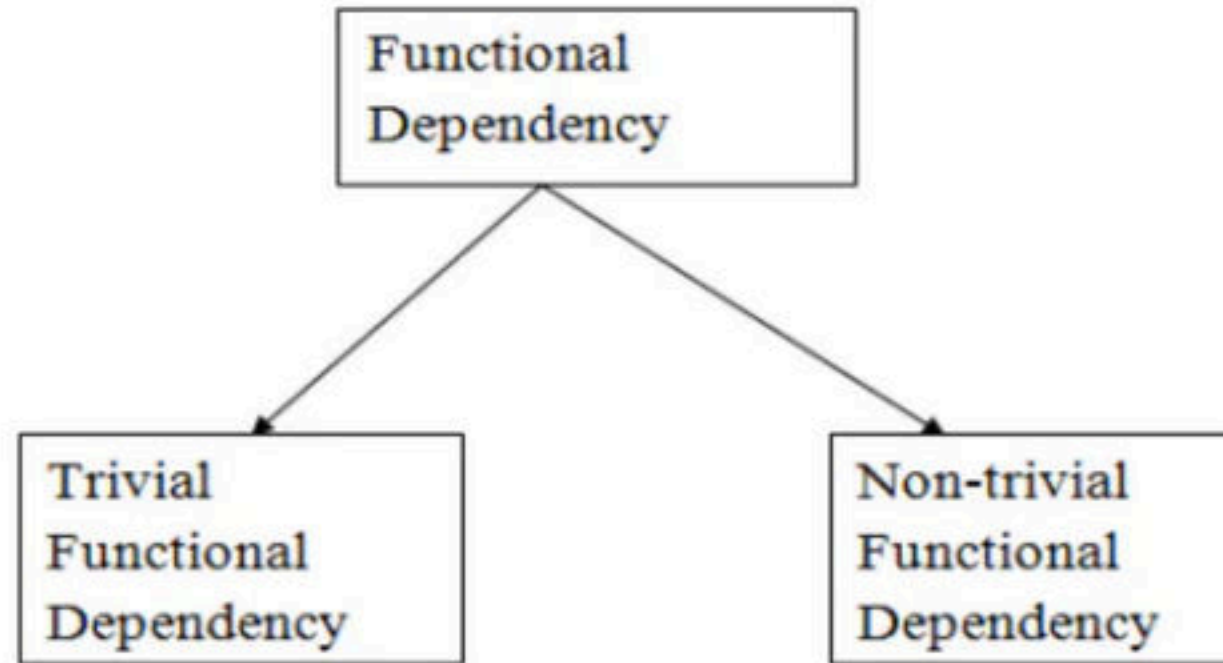
Here Emp_Id attribute can uniquely identify the Emp_Name attribute of employee table because if we know the Emp_Id, we can tell that employee name associated with it.

Functional dependency can be written as:

$$\text{Emp_Id} \rightarrow \text{Emp_Name}$$

We can say that Emp_Name is functionally dependent on Emp_Id.

Types of Functional dependency



1. Trivial functional dependency
2. Non-trivial functional dependency

1. Trivial functional dependency

$A \rightarrow B$ has trivial functional dependency **if B is a subset of A.**

The following dependencies are also trivial like: $A \rightarrow A$, $B \rightarrow B$

Consider a table with two columns **Employee_Id** and **Employee_Name**.

$\{\text{Employee_id}, \text{Employee_Name}\} \rightarrow \text{Employee_Id}$

is a trivial functional dependency as

Employee_Id is a subset of $\{\text{Employee_Id}, \text{Employee_Name}\}$.

$\text{Employee_Id} \rightarrow \text{Employee_Id}$

$\text{Employee_Name} \rightarrow \text{Employee_Name}$

are trivial dependencies too.

2. Non-trivial functional dependency

$A \rightarrow B$ has a non-trivial functional dependency if B is not a subset of A .

When $A \cap B$ is NULL, then $A \rightarrow B$ is called as complete non-trivial.

Example:

$ID \rightarrow Name,$

$Name \rightarrow DOB$

Transitive dependency:

A transitive is a type of functional dependency which happens when t is indirectly formed by two functional dependencies.

$X \rightarrow Z$ is a transitive dependency if the following three functional dependencies hold true:

$X \rightarrow Y$

Y does not $\rightarrow X$

$Y \rightarrow Z$

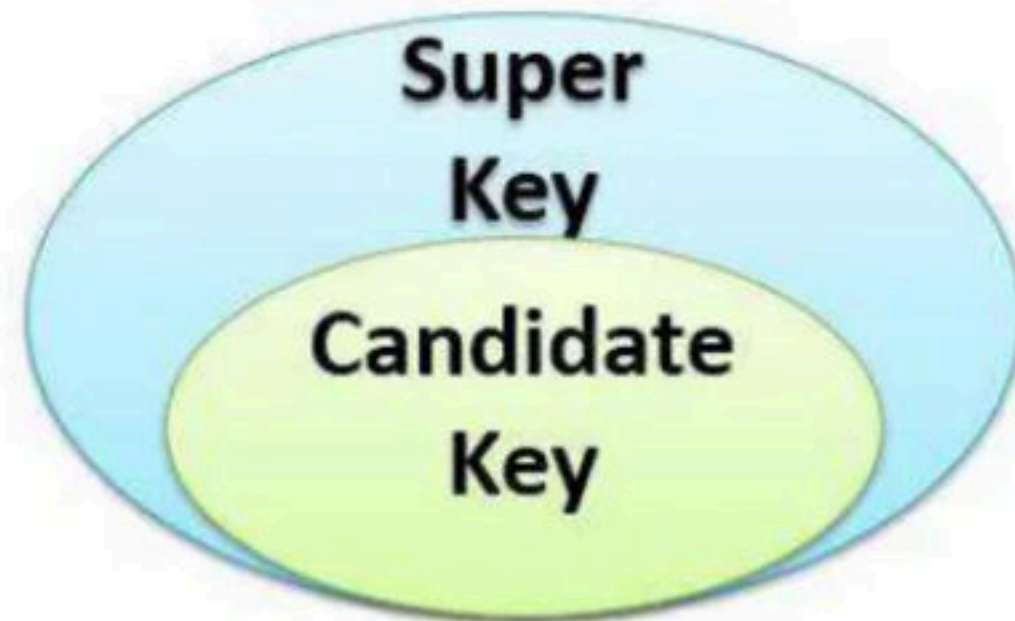
Note: A transitive dependency can only occur in a relation of three or more attributes.

Super Key, Candidate Key and Primary Key

Super Key: The set of attributes which can uniquely identify a tuple is known as Super Key.

Candidate Key: The minimal set of attribute which can uniquely identify a tuple is known as candidate key

Primary Key: There can be more than one candidate key in relation out of which one can be chosen as the primary key.



Q. Consider the following database table having A, B, C and D as its four attributes and four possible candidate keys (I, II, III and IV) for this table :

A	B	C	D
a ₁	b ₁	c ₁	d ₁
a ₂	b ₃	c ₃	d ₁
a ₁	b ₂	c ₁	d ₂

This is a correction of Previous Class.
Please Note down the correction

I : {B} II : {B, C} III : {A, D} IV : {C, D}

If different symbols stand for different values in the table (e.g., d₁ is definitely not equal to d₂), then which of the above could not be the candidate key for the database table ?

(1) I and III only
(3) II only

(2) III and IV only
(4) I only

(UGC NET
Jul 2016/p2)

Ans:

I : {B} II : {B, C} III : {A, D} IV : {C, D}

- ✓ (1) I and III only (2) III and IV only
(3) II only (4) I only

BC cannot be candidate key

because it is Superset of candidate key(B) that's why it is superkey

**This is a correction of Previous Class.
Please Note down the correction**

Comprehension:

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 98

Q.98

Identify primary key of table R with functional dependency set F

(1) BC

(2) AD

(3) A

(4) AB

Comprehension:

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 98

Q.98

Identify primary key of table R with functional dependency set F

(1) BC

(2) AD

(3) A ✓

(4) AB

Functional Dependency and Attribute Closure

STUDENT

STUD_NO	STUD_NAME	STUD_PHONE	STUD_STATE	STUD_COUNTRY	STUD_AGE
1	RAM	9716271721	Haryana	India	20
2	RAM	9898291281	Punjab	India	19
3	SUJIT	7898291981	Rajsthan	India	18
4	SURESH		Punjab	India	21

Table 1

{ STUD_NO->STUD_NAME,
STUD_NO->STUD_PHONE,
STUD_NO->STUD_STATE,
STUD_NO->STUD_COUNTRY,
STUD_NO -> STUD_AGE,
STUD_STATE->STUD_COUNTRY }

Functional Dependency Set: Functional Dependency set or FD set of a relation is the set of all FDs present in the relation. For Example, FD set for relation STUDENT shown in table 1 is:

Attribute Closure:

Attribute closure of an attribute set can be defined as set of attributes which can be functionally determined from it.

How to find attribute closure of an attribute set?

To find attribute closure of an attribute set:

Add elements of attribute set to the result set.

Recursively add elements to the result set which can be functionally determined from the elements of the result set.

Using FD set of table 1, attribute closure can be determined as:

$(\text{STUD_NO})^+ = \{\text{STUD_NO}, \text{STUD_NAME}, \text{STUD_PHONE}, \text{STUD_STATE}, \text{STUD_COUNTRY}, \text{STUD_AGE}\}$

$(\text{STUD_STATE})^+ = \{\text{STUD_STATE}, \text{STUD_COUNTRY}\}$

How to find Candidate Keys and Super Keys using Attribute Closure?

If attribute closure of an attribute set contains all attributes of relation, the attribute set will be super key of the relation.

If no subset of this attribute set can functionally determine all attributes of the relation, the set will be candidate key as well.

For Example, using FD set of table 1,

$(\text{STUD_NO}, \text{STUD_NAME})^+ = \{\text{STUD_NO}, \text{STUD_NAME}, \text{STUD_PHONE}, \text{STUD_STATE}, \text{STUD_COUNTRY}, \text{STUD_AGE}\}$

$(\text{STUD_NO})^+ = \{\text{STUD_NO}, \text{STUD_NAME}, \text{STUD_PHONE}, \text{STUD_STATE}, \text{STUD_COUNTRY}, \text{STUD_AGE}\}$

$(\text{STUD_NO}, \text{STUD_NAME})$ will be super key but not candidate key because its subset $(\text{STUD_NO})^+$ is equal to all attributes of the relation. So, STUD_NO will be a candidate key.

Q. Consider the following database table having A, B, C and D as its four attributes and four possible candidate keys (I, II, III and IV) for this table :

A	B	C	D
a ₁	b ₁	c ₁	d ₁
a ₂	b ₃	c ₃	d ₁
a ₁	b ₂	c ₁	d ₂

I : {B} II : {B, C} III : {A, D} IV : {C, D}

If different symbols stand for different values in the table (e.g., d₁ is definitely not equal to d₂), then which of the above could not be the candidate key for the database table ?

(1) I and III only
(3) II only

(2) III and IV only
(4) I only

(UGC NET
Jul 2016/p2)

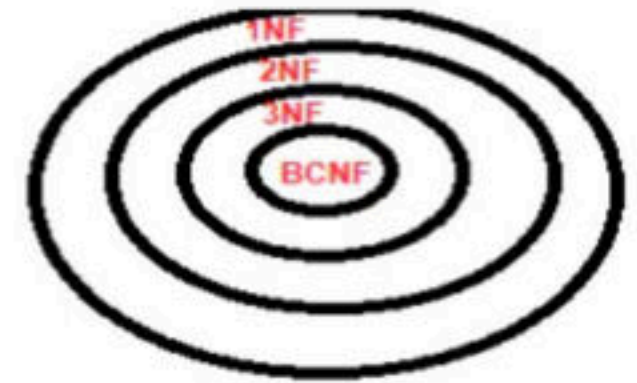
Ans:

I : {B} II : {B, C} III : {A, D} IV : {C, D}

-  (1) I and III only (2) III and IV only
(3) II only (4) I only

As B is the Candidate key so BC is the Super Key.
BC can not be a Candidate Key.

Normalization



Unnormalized Form

Remove Multivalued and composite attribute

1NF

>Single value in single cell , Value of each coloumn belong to same domain,
>PK can determine all the attribute, Every Column Should have unqiue name.

+

2NF

1.Full part of the LHS Should be candidate key, there should not be any partial dependency
2.If LHS is not CK then RHS should be prime.
3.If LHS and RHS both are non prime then no problem in 2NF

+

3NF

For each FD LHS must be candidate key or super key
OR
RHS is a prime attribute
**** LHS and RHS both non prime is not allowed in 3NF**

**** Must determine all the CK. then check prime attribute otherwise ans may be wrong**

+

BCNF

LHS should be super key or candidate key must ne satisfy
****RHS prime condition is not allowed.**

* Fully Functinaly dependent on candidate key
*(Non Prime->Non Prime) is allowed in 2NF

*No transitive dependency should be allowed
*(Non prime->Non Prime) X **not allowed**
(prime->non prime) X **not allowed**
(Non prime/prime ->prime) allowed

prime/nonprime->prime) **not allowed**
prime/non prime->non prime) **not allowed**

Q. Part 1

Relation R has eight attributes ABCDEFGH. Fields of R contain only atomic values.

$F = \{CH \rightarrow G, A \rightarrow BC, B \rightarrow CFH, E \rightarrow A, F \rightarrow EG\}$

is a set of functional dependencies (FDs) so that F^+ is exactly the set of FDs that hold for R

How many candidate keys does the relation R have?

- A. 3
- B. 4
- C. 5
- D. 6

Q. Part 2

The relation R is

- A. in 1NF, but not in 2NF.
- B. in 2NF, but not in 3NF.
- C. in 3NF, but not in BCNF
- D. in BCNF.

late
2013

R(ABCDEFGHIH)

$CH \rightarrow G$
 $A \rightarrow BC$
 $B \rightarrow CFH$
 $E \rightarrow A$
 $F \rightarrow EG$

D
Not
present
in RHS

In Right hand side D is not present
 So in candidate key D must be

Determining CK

$\checkmark (DA)^+ \rightarrow DABCFHEG$
 $\checkmark (DE)^+ \rightarrow DEABCFHG$
 $\checkmark (DF)^+ \rightarrow DFEGABCH$
 $\checkmark (DB)^+ \rightarrow DBCFHEGA$

Check Normal Normal

Non Prime	$CH \rightarrow G$	Non Prime	\checkmark 2NF	} in INF but Not in 2NF
Prime	$A \rightarrow B$	Prime	\checkmark 2NF	
Prime	$A \rightarrow C$	Non Prime	\times 2NF	
Prime	$B \rightarrow C$	Non Prime	\times 2NF	
Prime	$B \rightarrow F$	Prime	\checkmark 2NF	
Prime	$B \rightarrow H$	Prime	\checkmark 2NF	
Prime	$E \rightarrow A$	Prime	\checkmark 2NF	
Prime	$F \rightarrow E$	Prime	\checkmark 2NF	
Prime	$F \rightarrow G$	Non Prime	\times 2NF	

Prime Attributes are D, A, B, E, F
 Non Prime Attributes are C, G, H

Q. Part 1

Relation R has eight attributes ABCDEFGH. Fields of R contain only atomic values.

$F = \{CH \rightarrow G, A \rightarrow BC, B \rightarrow CFH, E \rightarrow A, F \rightarrow EG\}$

is a set of functional dependencies (FDs) so that F^+ is exactly the set of FDs that hold for R
How many candidate keys does the relation R have?

A. 3

B. 4 ✓

C. 5

D. 6

Q. Part 2

The relation R is

A. in 1NF, but not in 2NF. ✓

B. in 2NF, but not in 3NF.

C. in 3NF, but not in BCNF

D. in BCNF.

Comprehension:

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 98

Q.98

Identify primary key of table R with functional dependency set F

(1) BC

(2) AD

(3) A

(4) AB

Comprehension:

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 98

Q.98

Identify primary key of table R with functional dependency set F

(1) BC

(2) AD

(3) A ✓

(4) AB

Comprehension:

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F : \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 99

Q.99

Identify the normal form in which relation R belong to

(1) 1NF

(2) 2NF

(3) 3NF

(4) BCNF

Comprehension:

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F : \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 99

Q.99

Identify the normal form in which relation R belong to

(1) 1NF

(2) 2NF ✓

(3) 3NF

(4) BCNF

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functional dependency set F is given as : $F : \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 97

Q.97

Assume that given table R is decomposed in two tables

$R_1(A, B, C)$ with functional dependency set $f_1 = \{A \rightarrow B, A \rightarrow C\}$ and

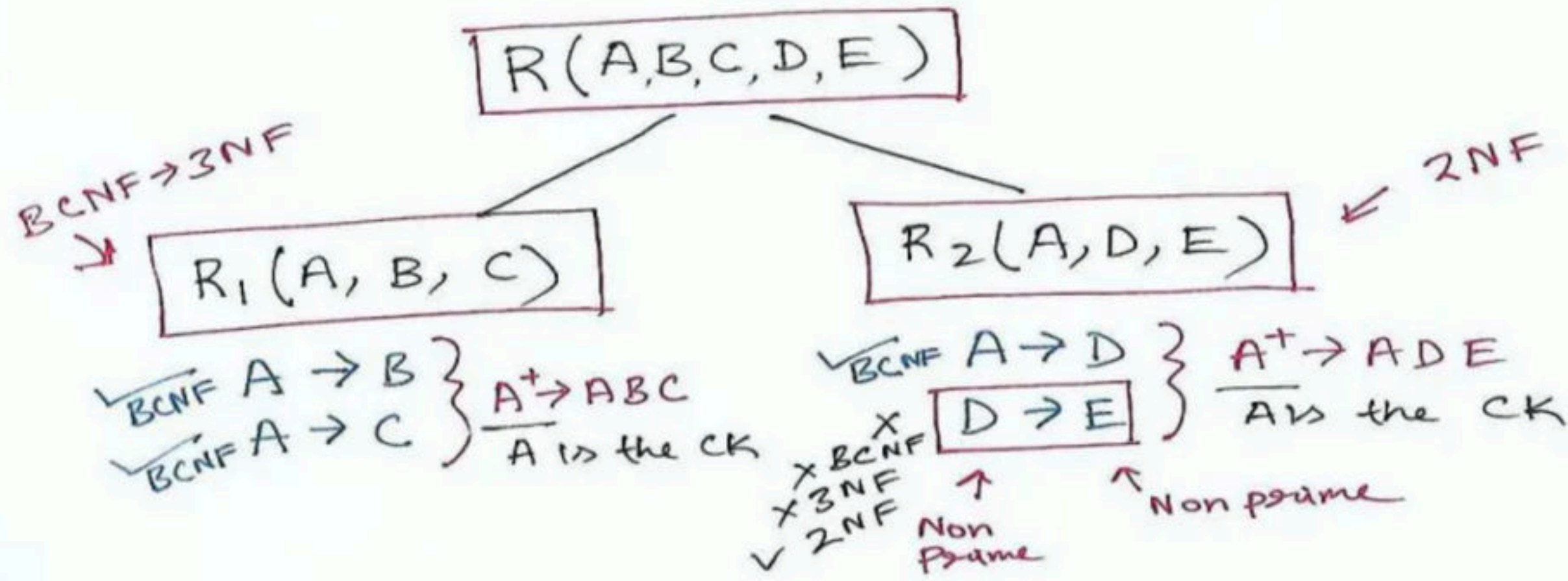
$R_2(A, D, E)$ with FD set $F_2 = \{A \rightarrow D, D \rightarrow E\}$.

Which of the following option is true w.r.t. given decomposition?

- (1) Dependency preservation property is followed
- (2) R_1 and R_2 are both in 2 NF
- (3) R_2 is in 2 NF and R_3 is in 3 NF
- (4) R_1 is in 3 NF and R_2 is in 2 NF

Net Dec 2019

Q) 97



Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 97

Q.97 Assume that given table R is decomposed in two tables

$R_1(A, B, C)$ with functional dependency set $f_1 = \{A \rightarrow B, A \rightarrow C\}$ and $R_2(A, D, E)$ with FD set $F_2 = \{A \rightarrow D, D \rightarrow E\}$.

Which of the following option is true w.r.t. given decomposition?

- (1) Dependency preservation property is followed
- (2) R_1 and R_2 are both in 2 NF
- (3) R_2 is in 2 NF and R_3 is in 3 NF
- (4) R_1 is in 3 NF and R_2 is in 2 NF ✓

Q. Consider the following relational schemes for a library database:

Book (Title, Author, Catalog_no, Publisher, Year, Price)

Collection (Title, Author, Catalog_no)

with in the following functional dependencies:

I. Title Author --> Catalog_no

II. Catalog_no --> Title, Author, Publisher, Year

III. Publisher Title Year --> Price

Assume {Author, Title} is the key for both schemes. Which of the following statements is true?

- (A) Both Book and Collection are in BCNF
- (B) Both Book and Collection are in 3NF only
- (C) Book is in 2NF and Collection is in 3NF
- (D) Both Book and Collection are in 2NF only

Q. Consider the following relational schemes for a library database:

Book (Title, Author, Catalog_no, Publisher, Year, Price)

Collection (Title, Author, Catalog_no)

with in the following functional dependencies:

I. Title Author \rightarrow Catalog_no

II. Catalog_no \rightarrow Title, Author, Publisher, Year

III. Publisher Title Year \rightarrow Price

Assume {Author, Title} is the key for both schemes. Which of the following statements is true?

(A) Both Book and Collection are in BCNF

(B) Both Book and Collection are in 3NF only

(C) Book is in 2NF and Collection is in 3NF



(D) Both Book and Collection are in 2NF only

Q . Consider the following four relational schemas. For each schema, all non-trivial functional dependencies are listed, The underlined attributes are the respective primary keys.

Schema I: Registration(rollno, courses)

Field 'courses' is a set-valued attribute containing the set of courses a student has registered for.

Non-trivial functional dependency

$\text{rollno} \rightarrow \text{courses}$

Schema II: Registration (rollno, courseid, email)

Non-trivial functional dependencies:

$\text{rollno}, \text{courseid} \rightarrow \text{email}$

$\text{email} \rightarrow \text{rollno}$

Schema III: Registration (rollno, courseid, marks, grade)

Non-trivial functional dependencies:

rollno, courseid, \rightarrow marks, grade

marks \rightarrow grade

Schema IV: Registration (rollno, courseid, credit)

Non-trivial functional dependencies:

rollno, courseid \rightarrow credit

courseid \rightarrow credit

Which one of the relational schemas above is in 3NF but not in BCNF?

- (A) Schema I
- (B) Schema II
- (C) Schema III
- (D) Schema IV

Schema I	Schema II	Schema III	Schema IV
$\text{Reg}(\underline{R}, C)$ $\underline{R} \rightarrow C$ \uparrow PK (HS)	$\text{Reg}(\underline{R, C}, E)$ $\underline{R, C} \rightarrow E$ (LHS)	$\text{Reg}(\underline{R, C}, M, G)$ $\underline{R, C} \rightarrow M, G$ (LHS)	$\text{Reg}(\underline{R, Courseid}, Credit)$ $\underline{R, Courseid} \rightarrow Credit$ (LHS)
\checkmark BCNF \downarrow \checkmark 3NF \downarrow \checkmark 2NF \downarrow 1NF	\checkmark BCNF \downarrow \checkmark 3NF \downarrow \times BCNF \downarrow \checkmark 3NF \downarrow \times BCNF \downarrow \checkmark 2NF	\checkmark BCNF \downarrow \checkmark 3NF \downarrow \checkmark 2NF \downarrow \times BCNF \downarrow \times 3NF \downarrow \checkmark 2NF	\checkmark BCNF \downarrow \checkmark 3NF \downarrow \checkmark 2NF \downarrow \checkmark 1NF
Relation is in BCNF	$\underline{E} \rightarrow R$ \downarrow Non prime \downarrow Not CK	$\underline{M} \rightarrow G$ \downarrow Non prime	$\underline{Courseid} \rightarrow Credit$ \downarrow prime (LHS)
	$\underline{R} \rightarrow E$ \downarrow Prime (RHS)	$\underline{G} \rightarrow M$ \downarrow Non prime	$\underline{Credit} \rightarrow Courseid$ \downarrow Non prime (RHS)
	Relation is in 3NF but not in BCNF	Relation is in 2NF	Partial dependency \rightarrow \times 2NF \checkmark 1NF
	<u>Ans</u> \checkmark Schema II		Relation is in 1NF only.

GATE 2018 Ques on Normalization

Which one of the relational schemas above is in 3NF but not in BCNF?

(A) Schema I

(B) Schema II ✓

(C) Schema III

(D) Schema IV

Q.

A database of research articles in a journal uses the following schema.

(VOLUME, NUMBER, STARTPAGE, ENDPAGE, TITLE, YEAR, PRICE)

The primary key is (VOLUME, NUMBER, STARTPAGE, ENDPAGE) and the following functional dependencies exist in the schema.

(VOLUME, NUMBER, STARTPAGE, ENDPAGE) \rightarrow TITLE

(VOLUME, NUMBER) \rightarrow YEAR

(VOLUME, NUMBER, STARTPAGE, ENDPAGE) \rightarrow PRICE

The database is redesigned to use the following schemas.

(VOLUME, NUMBER, STARTPAGE, ENDPAGE, TITLE, PRICE)

(VOLUME, NUMBER, YEAR)

Which is the weakest normal form that the new database satisfies, but the old one does not?

- (A) 1NF
- (B) 2NF
- (C) 3NF
- (D) BCNF

GATE 2016

GATE 2016

V, N, S, E, T, Y, P

PK is (V, N, S, E)

This is Not in 2NF
It is only in 1NF

Not in 2NF
Partial Dependency
 $V, N, S, E \rightarrow T$
 $V, N \rightarrow Y$ (Non Prime)
 $V, N, S, E \rightarrow P$

V, N, S, E, T, P

CK
CK
 $V, N, S, E \rightarrow T$
 $V, N, S, E \rightarrow P$
 $(V, N, S, E)^+ \rightarrow VNSETP$
CK
This is in BCNF

V, N, Y

$V, N \rightarrow Y$
 $(V, N)^+ \rightarrow VNY$
CK
This is in BCNF

BCNF
3NF

Ans $V \rightarrow$ 2NF
1NF

← is the weakest Normal Form that the new database satisfies. but the old one does not.

Which is the weakest normal form that the new database satisfies, but the old one does not?

(A) 1NF

(B) 2NF 

(C) 3NF

(D) BCNF

Explanation: Old relation has functional dependency : Volume, Number \rightarrow Year as partial dependency. So it does not follow 2NF.

But, there is no partial dependency in the New relation and so it satisfies 2NF as well as 3NF and BCNF .

Therefore, 2NF is the weakest normal form that the new database satisfies, but the old one does not. Option (B) is true.

GATE 2016

GATE Question: Consider the relation scheme $R = \{E, F, G, H, I, J, K, L, M, M\}$ and the set of functional dependencies $\{\{E, F\} \rightarrow \{G\}, \{F\} \rightarrow \{I, J\}, \{E, H\} \rightarrow \{K, L\}, K \rightarrow \{M\}, L \rightarrow \{N\}$ on R . What is the key for R ? (GATE-CS-2014)

- A. $\{E, F\}$
- B. $\{E, F, H\}$
- C. $\{E, F, H, K, L\}$
- D. $\{E\}$

Answer: Finding attribute closure of all given options, we get:

$$\{E,F\}^+ = \{EFGIJ\}$$

$$\{E,F,H\}^+ = \{EFHGIJKLMN\}$$

$$\{E,F,H,K,L\}^+ = \{EFHGIJKLMN\}$$

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

$\{EFH\}^+$ and $\{EFHKL\}^+$ results in set of all attributes, but EFH is minimal. So it will be candidate key. So correct option is (B).

GATE Question: Consider a relation scheme $R = (A, B, C, D, E, H)$ on which the following functional dependencies hold: $\{A \rightarrow B, BC \rightarrow D, E \rightarrow C, D \rightarrow A\}$. What are the candidate keys of R ?
[GATE 2005]

- (a) AE, BE
- (b) AE, BE, DE
- (c) AEH, BEH, BCH
- (d) AEH, BEH, DEH

Answer: $(AE)^+ = \{ABECD\}$ which is not set of all attributes. So AE is not a candidate key. Hence option A and B are wrong.

$(AEH)^+ = \{ABCDEH\}$

$(BEH)^+ = \{BEHCDA\}$

$(BCH)^+ = \{BCHDA\}$ which is not set of all attributes. So BCH is not a candidate key. Hence option C is wrong.

So correct answer is D.

Equivalence of Functional Dependencies

Given a Relation with different FD sets for that relation, we have to find out whether one FD set is subset of other or both are equal.

How to find relationship between two FD sets?

Let FD1 and FD2 are two FD sets for a relation R.

- 1.If all FDs of FD1 can be derived from FDs present in FD2, we can say that $FD2 \supseteq FD1$.
- 2.If all FDs of FD2 can be derived from FDs present in FD1, we can say that $FD1 \supseteq FD2$.
- 3.If 1 and 2 both are true, $FD1=FD2$.

How to find equivalence in FD?

Two set of FD's 'F' and 'G' are said to be equivalent iff

$$F^+ = G^+$$

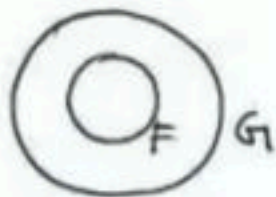
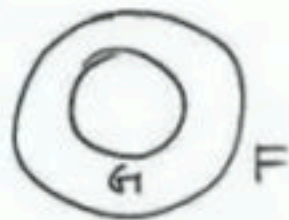
Hence Equivalence means that every FD's in F can be inferred using G and every FD in G can be inferred using F.

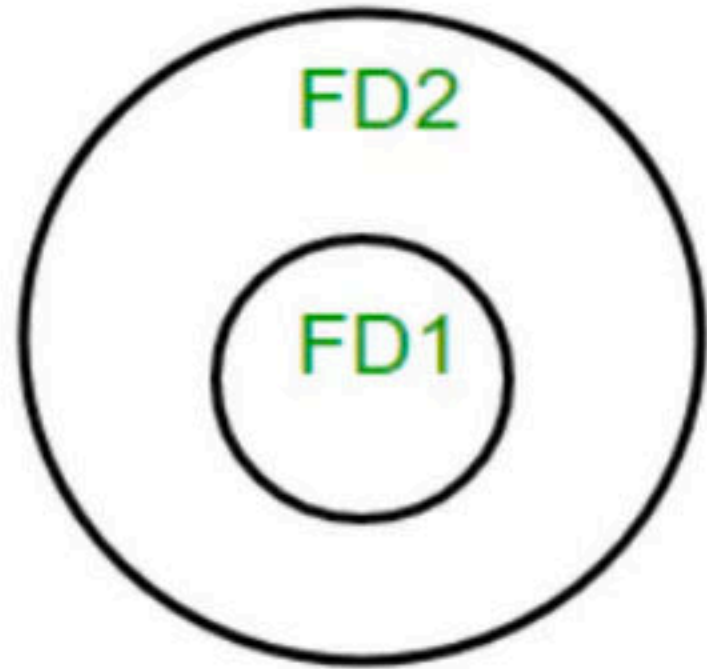
i.e. $F = G$ if both

F covers G and G covers F holds

$F \supseteq G$ ■ F covers G

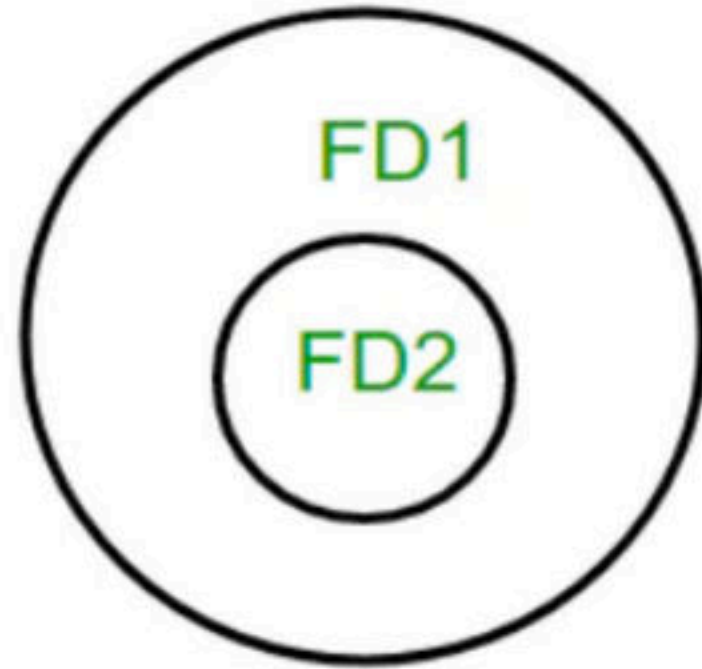
$G \supseteq F$ G covers F





FD2 \supset FD1

FD2 Covers FD1



FD1 \supset FD2

FD1 Covers FD2



FD1 $=$ FD2

So FD1 and FD2 are equivalent

Q. A relation $R(A,B,C,D)$ having two FD sets

$FD1 = \{A \rightarrow B, B \rightarrow C, AB \rightarrow D\}$ and

$FD2 = \{A \rightarrow B, B \rightarrow C, A \rightarrow C, A \rightarrow D\}$

Find the relationship between two FD sets.

two FD sets are semantically equivalent or not?

Step 1. Checking whether all FDs of FD1 are present in FD2

- $A \rightarrow B$ is present in set FD2.
- $B \rightarrow C$ is also present in set FD2.

But

- $AB \rightarrow D$ is not present in FD2 directly. We have to check whether we can derive it or not.

For set FD2, $(AB)^+ = \{A, B, C, D\}$.

It means that AB can functionally determine A, B, C and D.

So $AB \rightarrow D$ will also hold in set FD2.

As all FDs in set FD1 also hold in set FD2,
 $FD2 \supset FD1$ is true.

Step 2. Checking whether all FDs of FD2 are present in FD1

- A->B is present in set FD1.

- B->C is also present in set FD1.

but

- A->C is not present directly in FD1. We have to check whether we can derive it or not.

For set FD1,

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

It means that A can functionally determine A, B, C and D.

SO A->C will also hold in set FD1.

- A->D is not present in FD1. We will check whether we can derive it or not.

For set FD1, $(A)^+ = \{A, B, C, D\}$.

It means that A can functionally determine A, B, C and D.

SO A->D will also hold in set FD1.

As all FDs in set FD2 also hold in set FD1, $FD1 \supset FD2$ is true.

Step 3. As $FD2 \supset FD1$ and $FD1 \supset FD2$ both are true $FD2 = FD1$ is true. These two FD sets are semantically equivalent.

Minimal cover:

A minimal cover of a set of FDs F is a minimal set of functional dependencies F_{\min} that is equivalent to F . There can be many such minimal covers for a set of functional dependencies F .

Important definitions:

Extraneous attributes: An attribute of a functional dependency is said to be extraneous if we can remove it without changing the closure of the set of functional dependencies.

Canonical cover: A canonical cover F_c of a set of functional dependencies F such that ALL the following properties are satisfied:

- F logically implies all dependencies in F_c .
- F_c logically implies all dependencies in F .
- No functional dependency in F_c contains an extraneous attribute.
- Each left side of a functional dependency in F_c is unique. That is, there are no two dependencies $\alpha_1 \rightarrow \beta_1$ and $\alpha_2 \rightarrow \beta_2$ in such that $\alpha_1 \rightarrow \alpha_2$.

1. Use the union rule to replace any dependencies in $\alpha_1 \rightarrow \beta_1$ and $\alpha_1 \rightarrow \beta_2$ with $\alpha_1 \rightarrow \beta_1\beta_2$.
 2. Find a functional dependency $\alpha \rightarrow \beta$ with an extraneous attribute either in α or in β .
 3. If an extraneous attribute is found, delete it from $\alpha \rightarrow \beta$.
- until F does not change**

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F : \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 100

Q.100

Identify the redundant functional dependency in F

(1) $BC \rightarrow D$

(2) $D \rightarrow E$

(3) $A \rightarrow D$

(4) $A \rightarrow BC$

Q) 100

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

$A \rightarrow BC$	}	$A^+ \rightarrow ABCDE$
$D \rightarrow E$		$D^+ \rightarrow DE$
$BC \rightarrow D$		$BC^+ \rightarrow BCDE$
$A \rightarrow D$		

Check after Removing $BC \rightarrow D$ \Rightarrow

$A \rightarrow BC$	}	$A^+ \rightarrow ABCDE$
$D \rightarrow E$		$D^+ \rightarrow DE$
$A \rightarrow D$		$BC^+ \rightarrow \boxed{BC} \text{ --- ?}$

Check after Removing $A \rightarrow D$

$A \rightarrow BC$	}	$A^+ \rightarrow ABCDE \checkmark$
$D \rightarrow E$		$D^+ \rightarrow DE \checkmark$
$BC \rightarrow D$		$BC^+ \rightarrow BCDE \checkmark$

\Leftarrow This is Redundant

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functional dependency set F is given as : $F : \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

SubQuestion No : 100

Q.100

Identify the redundant functional dependency in F

(1) $BC \rightarrow D$

(2) $D \rightarrow E$

(3) $A \rightarrow D$ ✓

(4) $A \rightarrow BC$

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

Q.96

Minimal cover F' of functional dependency set F is

- (1) $F' = \{A \rightarrow B, A \rightarrow C, BC \rightarrow D, D \rightarrow E\}$
- (2) $F' = \{A \rightarrow BC, B \rightarrow D, D \rightarrow E\}$
- (3) $F' = \{A \rightarrow B, A \rightarrow C, A \rightarrow D, D \rightarrow E\}$
- (4) $F' = \{A \rightarrow B, A \rightarrow C, B \rightarrow D, C \rightarrow D, D \rightarrow E\}$

Answer question (96-100) based on the problem statement given below :

An organization needs to maintain database having five attributes A, B, C, D, E . These attributes are functionally dependent on each other for which functionally dependency set F is given as : $F: \{A \rightarrow BC, D \rightarrow E, BC \rightarrow D, A \rightarrow D\}$. Consider a universal relation $R(A, B, C, D, E)$ with functional dependency set F . Also all attributes are simple and take atomic values only.

Q.96

Minimal cover F' of functional dependency set F is

- (1) $F' = \{A \rightarrow B, A \rightarrow C, BC \rightarrow D, D \rightarrow E\}$ ✓
- (2) $F' = \{A \rightarrow BC, B \rightarrow D, D \rightarrow E\}$
- (3) $F' = \{A \rightarrow B, A \rightarrow C, A \rightarrow D, D \rightarrow E\}$
- (4) $F' = \{A \rightarrow B, A \rightarrow C, B \rightarrow D, C \rightarrow D, D \rightarrow E\}$

The following functional dependencies hold true for the relational schema $R\{V, W, X, Y, Z\}$:

$V \rightarrow W$

$VW \rightarrow X$

$Y \rightarrow VX$

$Y \rightarrow Z$

Which of the following is irreducible equivalent for this set of functional dependencies?

A. $V \rightarrow W$

$V \rightarrow X$

$Y \rightarrow V$

$Y \rightarrow Z$

B. $V \rightarrow W$

$W \rightarrow X$

$Y \rightarrow V$

$Y \rightarrow Z$

C. $V \rightarrow W$

$V \rightarrow X$

$Y \rightarrow V$

$Y \rightarrow X$

$Y \rightarrow Z$

D. $V \rightarrow W$

$W \rightarrow X$

$Y \rightarrow V$

$Y \rightarrow X$

$Y \rightarrow Z$

GATE 2017

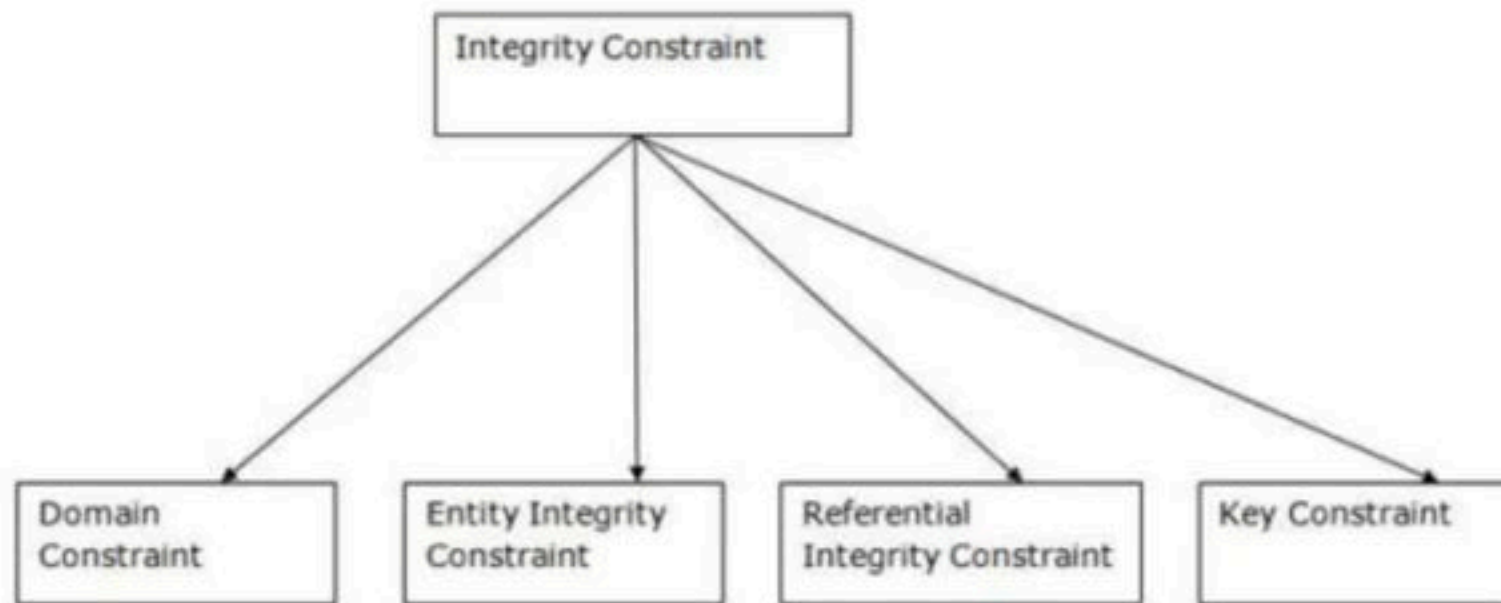
Ans: Option A is the Correct Ans

Try to Compare all the Attributes Closer with the options.

Integrity Constraints

- Integrity constraints are a set of rules. It is used to maintain the quality of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to guard against accidental damage to the database.

Types of Integrity Constraint



1. Domain constraints

- Domain constraints can be defined as the definition of a valid set of values for an attribute.
- The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

ID	NAME	SEMENSTER	AGE
1000	Tom	1 st	17
1001	Johnson	2 nd	24
1002	Leonardo	5 th	21
1003	Kate	3 rd	19
1004	Morgan	8 th	A

Not allowed. Because AGE is an integer attribute

2. Entity integrity constraints

- The entity integrity constraint states that primary key value can't be null.
- This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
- A table can contain a null value other than the primary key field.

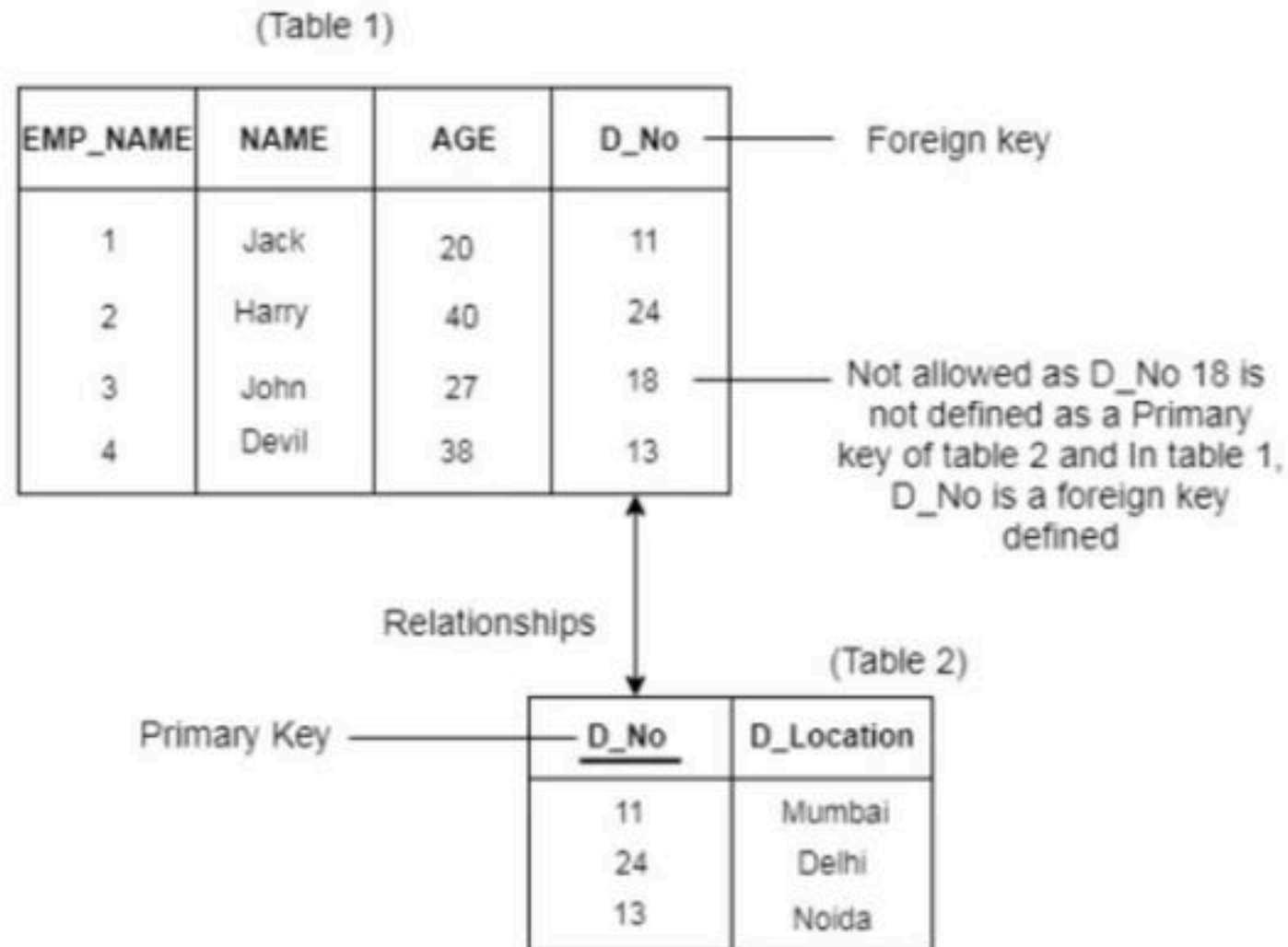
EMPLOYEE

EMP_ID	EMP_NAME	SALARY
123	Jack	30000
142	Harry	60000
164	John	20000
	Jackson	27000

Not allowed as primary key can't contain a NULL value

3. Referential Integrity Constraints

- A referential integrity constraint is specified between two tables.
- In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.



4. Key constraints

- Keys are the entity set that is used to identify an entity within its entity set uniquely.
- An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.

ID	NAME	SEMENSTER	AGE
1000	Tom	1 st	17
1001	Johnson	2 nd	24
1002	Leonardo	5 th	21
1003	Kate	3 rd	19
1002	Morgan	8 th	22

Not allowed. Because all row must be unique

Referential Integrity (Foreign Key maintain this)

Referenced table

No violation

Referencing table

May cause violation

Insert

Delete

May cause violation
if the entry exists in
referencing table also
→ On Delete cascade
→ On Delete set null

will not cause any violation

Update

May cause violation
→ On update cascade
→ On update set null

May cause violation if
you update the FK.

on delete cascade and on delete set null.

On delete cascade: if a row of the referenced table is deleted, then all matching rows in the referencing table are deleted.

On delete set null: referencing table to be set null in same condition above.

	empid	managerid
x	20	40
	25	40
x	30	35
x	35	20
	40	45
	45	25

← ~~on delete~~ deleting the tuple (20, 40) the set of other tuple must be deleted to maintain the referential integrity is (35, 20) then (30, 35)

One of the purposes of applying data integrity constraints in databases is

- A. Controlling that user to access data
- B. Improving the quality of data entered for specific property
- C. Data cannot be updated
- D. Avoiding to enter duplicate records

One of the purposes of applying data integrity constraints in databases is

- A. Controlling that user to access data
- B. Improving the quality of data entered for specific property
- C. Data cannot be updated
- D. Avoiding to enter duplicate records

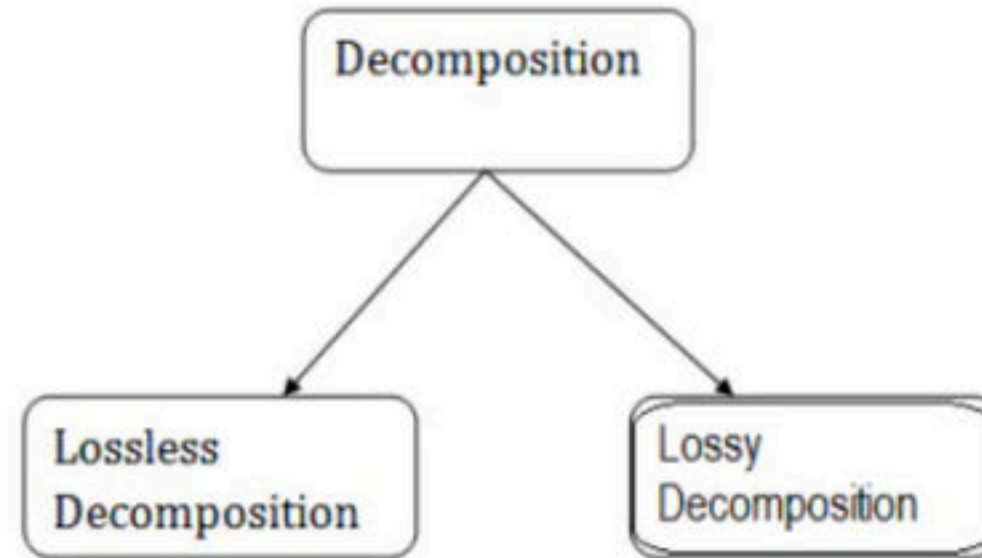
Relational Decomposition and Join

Concept and PYQs

Relational Decomposition

- When a relation in the relational model is not in appropriate normal form then the decomposition of a relation is required.
- In a database, it breaks the table into multiple tables.
- If the relation has no proper decomposition, then it may lead to problems like loss of information.
- Decomposition is used to eliminate some of the problems of bad design like anomalies, inconsistencies, and redundancy.

Types of Decomposition



Lossless Decomposition

- If the information is not lost from the relation that is decomposed, then the decomposition will be lossless.
- The lossless decomposition guarantees that the join of relations will result in the same relation as it was decomposed.
- The relation is said to be lossless decomposition if natural joins of all the decomposition give the original relation.

Example:

EMPLOYEE_DEPARTMENT table:

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY	DEPT_ID	DEPT_NAME
22	Denim	28	Mumbai	827	Sales
33	Alina	25	Delhi	438	Marketing
46	Stephan	30	Bangalore	869	Finance
52	Katherine	36	Mumbai	575	Production
60	Jack	40	Noida	678	Testing

The above relation is decomposed into two relations

EMPLOYEE and **DEPARTMENT**

EMPLOYEE table:

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY
22	Denim	28	Mumbai
33	Alina	25	Delhi
46	Stephan	30	Bangalore
52	Katherine	36	Mumbai
60	Jack	40	Noida

DEPARTMENT table

DEPT_ID	EMP_ID	DEPT_NAME
827	22	Sales
438	33	Marketing
869	46	Finance
575	52	Production
678	60	Testing

Now, when these two relations are joined on the common column "EMP_ID", then the resultant relation will look like:

Employee ⋈ Department

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY	DEPT_ID	DEPT_NAME
22	Denim	28	Mumbai	827	Sales
33	Alina	25	Delhi	438	Marketing
46	Stephan	30	Bangalore	869	Finance
52	Katherine	36	Mumbai	575	Production
60	Jack	40	Noida	678	Testing

Hence, the decomposition is Lossless join decomposition.

Lossless Decomposition in DBMS

Decomposition of a relation R into R1 and R2 is a lossless-join decomposition if at least one of the following functional dependencies are in F+ (Closure of functional dependencies)

* There must be a

$$R1 \cap R2 \rightarrow R1-R2$$

OR

$$R1 \cap R2 \rightarrow R2-R1$$

*any one of them

Lossy Decomposition :

"The decomposition of relation R into R1 and R2 is **Lossy** when the join of R1 and R2 does not yield the same relation as in R."

Lossless and Lossy Decomposition.

- * There must be a common attribute in the decomposed table R_1 and R_2 for lossless decomposition.
- * Common attribute should be CK or SK of either R_1 or R_2 or Both. for loss-less decomposition.
- * $R_1 \cup R_2 \equiv R$
- * $R_1 \cap R_2 \neq \emptyset$
- * R_1 and R_2 is lossless if and only if F^+ contains either FD

$$\text{OR } \left. \begin{array}{l} R_1 \cap R_2 \rightarrow R_2 - R_1 \\ R_1 \cap R_2 \rightarrow R_1 - R_2 \end{array} \right\} \text{ any one of them}$$

Q) $R(A, B, C)$ with FD $(A \rightarrow B)$ is decomposed into $R_1(A, B)$ and $R_2(B, C)$
 Check whether the decomposition is lossy or lossless

$$* \boxed{R_1 \cap R_2 \rightarrow R_1 - R_2}$$

$$\underbrace{(A, B) \cap (B, C)}_B \rightarrow \underbrace{(A, B) - (B, C)}_A$$

[Deduct the common element from 1st one]

we have to check the FD $\boxed{B \rightarrow A}$ is present in the F^+

OR not.

given FD is $\boxed{A \rightarrow B}$, from this we can't determine $\boxed{B \rightarrow A}$

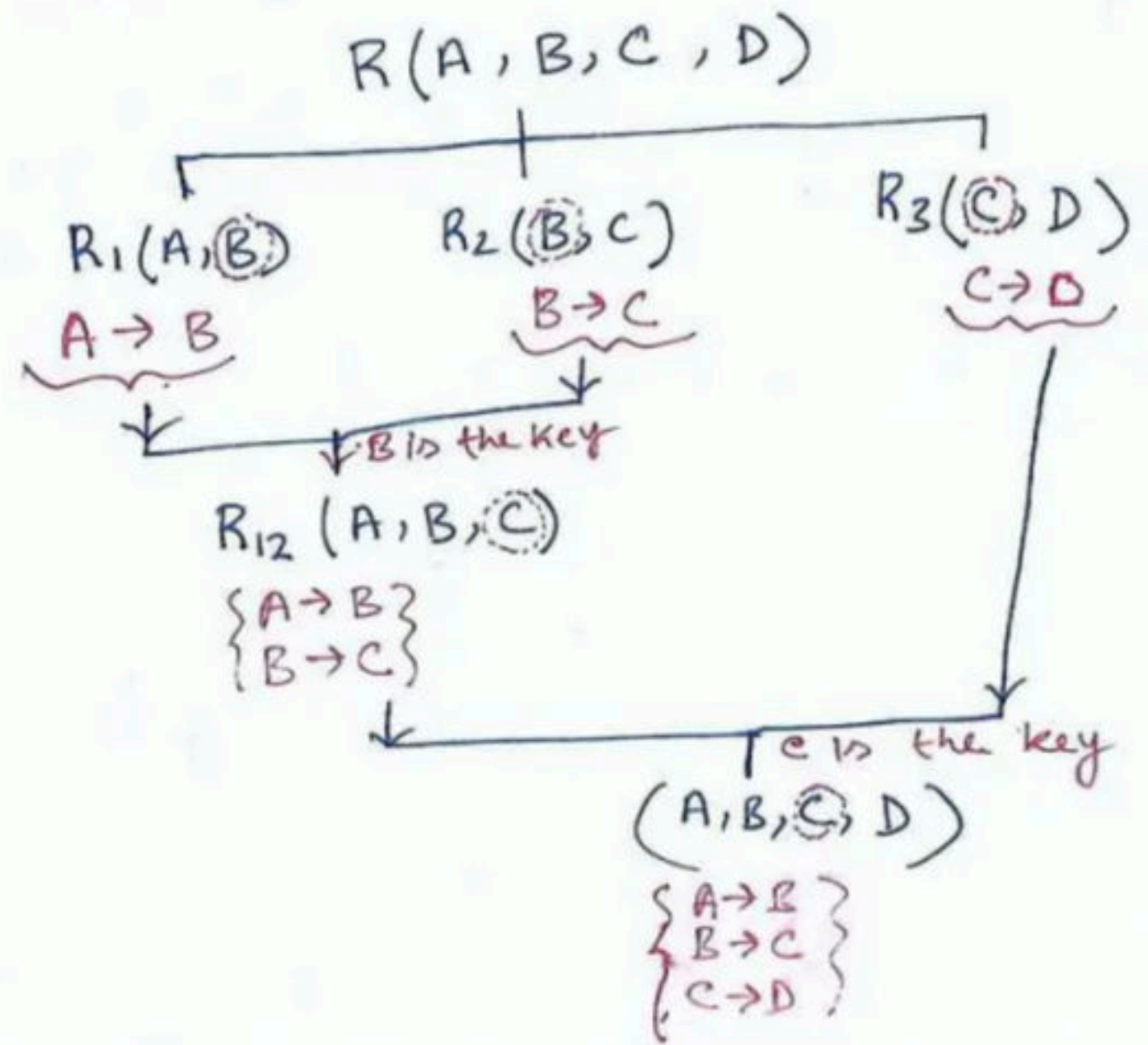
so this is not present in F^+ .

$$* \boxed{R_1 \cap R_2 \rightarrow R_2 - R_1}$$

$$\underbrace{(A, B) \cap (B, C)}_B \rightarrow \underbrace{(B, C) - (A, B)}_C$$

$B \rightarrow C$ can't be determine from the given FD.
~~So~~ So, none of them are contains in the FD.
 Hence Decomposition is lossy.

8) $R(A, B, C, D)$
 $F: \{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$ decomposed into
 $R_1(A, B), R_2(B, C), R_3(C, D)$.
 Is this lossless?



if $(R_1 \bowtie (R_2 \bowtie R_3)) = R$
 OR $((R_1 \bowtie R_2) \bowtie R_3) = R$
 and the
 Common attribute
 in both the relation
 is the key.
 So the decomposition
 is lossless.

Q) $R(ABCDEFGHIJ)$

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

$R_1(ABC) \quad R_2(ADE) \quad R_3(BF) \quad R_4(FGH) \quad R_5(DIJ)$

Check is it lossless or lossy.

$(((((ABC) \bowtie (ADE)) \bowtie (BF)) \bowtie (BF)) \bowtie (FGH)) \bowtie (DIJ))$

\downarrow
 $((((ABCDE) \bowtie BF) \bowtie (FGH)) \bowtie (DIJ))$

\downarrow
 $((((ABCDEF) \bowtie (FGH)) \bowtie (DIJ))$

\downarrow
 $((ABCDEFGHI) \bowtie (DIJ))$

\downarrow
 $(ABCDEFGHIJ)$
R

So the decomposition is lossless.

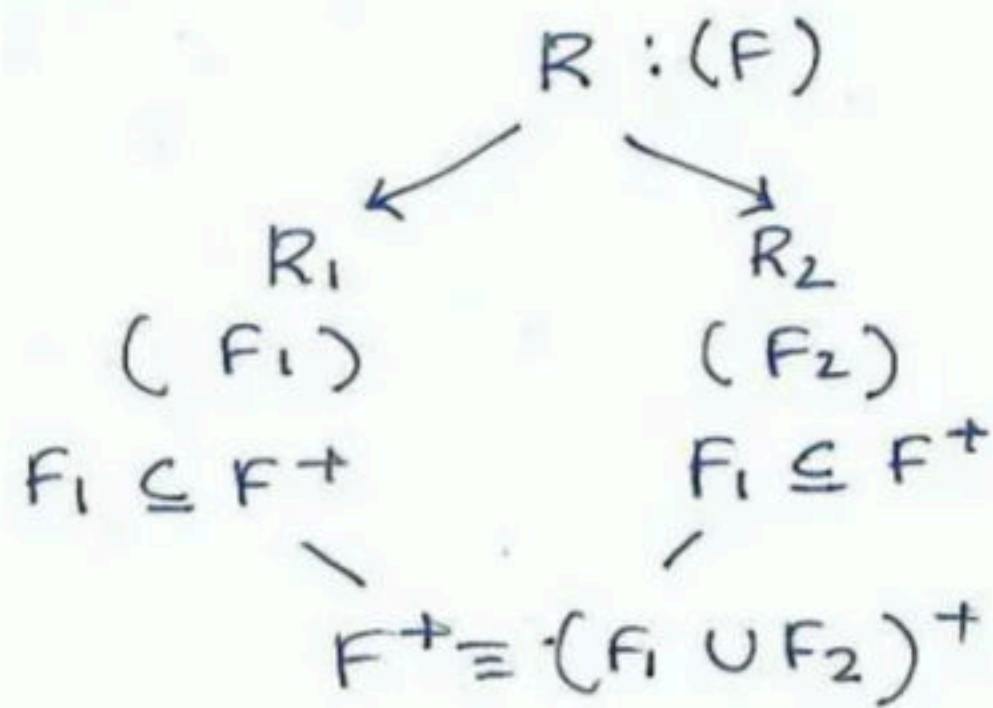
Dependency Preserving

- It is an important constraint of the database.
- In the dependency preservation, at least one decomposed table must satisfy every dependency.
- If a relation R is decomposed into relation R_1 and R_2 , then the dependencies of R either must be a part of R_1 or R_2 or must be derivable from the combination of functional dependencies of R_1 and R_2 .
- For example, suppose there is a relation $R(A, B, C, D)$ with functional dependency set $(A \rightarrow BC)$. The relational R is decomposed into $R_1(ABC)$ and $R_2(AD)$ which is dependency preserving because FD $A \rightarrow BC$ is a part of relation $R_1(ABC)$.

Dependency Preserving decomposition

The decomposition of the relation R with FD's ' F ' into R_1 and R_2 with FD's ' F_1 ' & ' F_2 ' respectively, is said to be dependency preserving iff

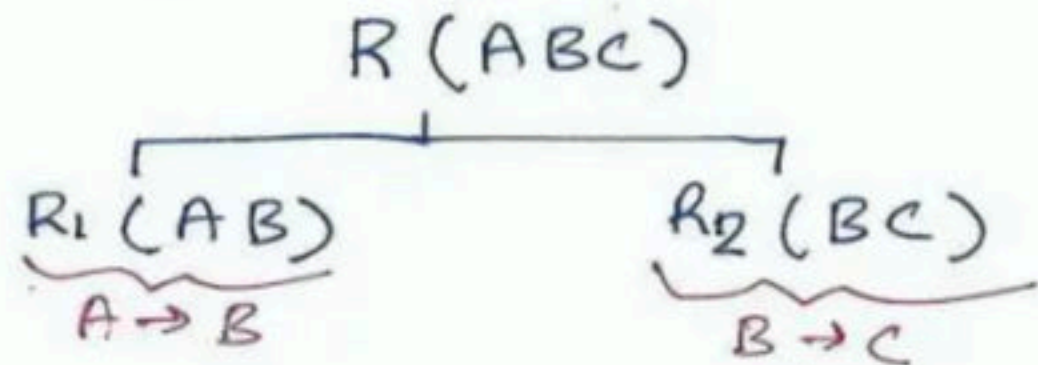
$$F^+ = (F_1 \cup F_2)^+$$



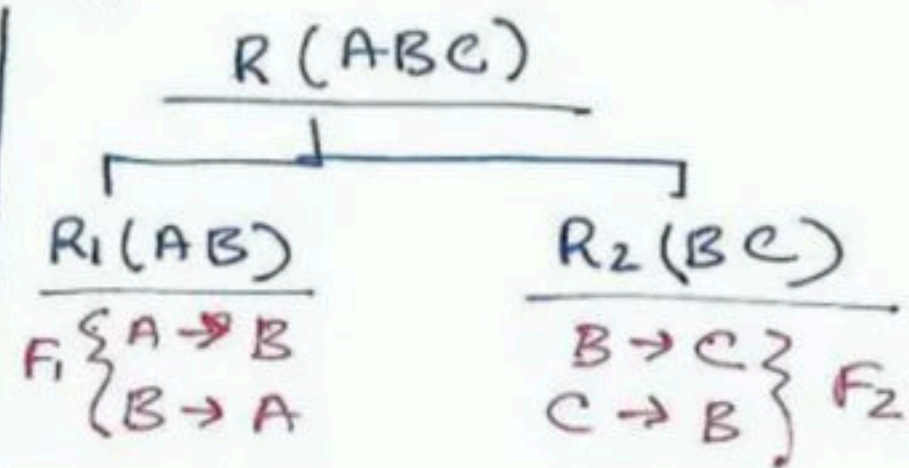
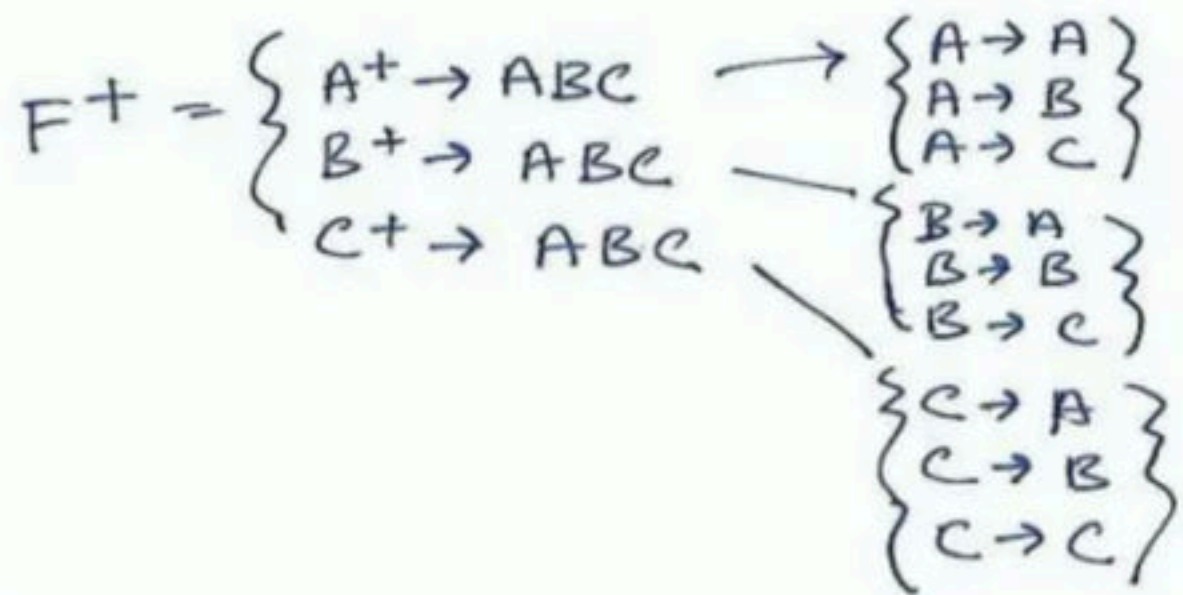
F^+ covers F_1
and F^+ covers F_2
and F^+ equivalence
to $(F_1 \cup F_2)^+$

Q) $R(ABC)$ FD: $\{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$ is decomposed into $R_1(AB)$ and $R_2(BC)$ ~~is~~
is this decomposition is dependency preserving?

Q) $R(ABC)$ FD: $\{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$ is decomposed into $R_1(AB)$ and $R_2(BC)$ ~~is~~
 is this decomposition is dependency preserving;



So where is $C \rightarrow A$?



$(F_1 \cup F_2)^+ = F^+$ Hence dependency preserving

$\left\{ \begin{array}{l} A \rightarrow B \\ B \rightarrow A \\ B \rightarrow C \\ C \rightarrow B \end{array} \right\} + \left\{ \begin{array}{l} C \rightarrow A \\ A \rightarrow C \\ A \rightarrow A \\ B \rightarrow B \\ C \rightarrow C \end{array} \right\}$ by the closure

Q. Consider a schema $R(A, B, C, D)$ and following functional dependencies.

$A \rightarrow B$

$B \rightarrow C$

$C \rightarrow D$

$D \rightarrow B$

Then decomposition of R into $R_1(A, B)$, $R_2(B, C)$ and $R_3(B, D)$ is _____ .

- (A) Dependency preserving and lossless join.
- (B) Lossless join but not dependency preserving.
- (C) Dependency preserving but not lossless join.
- (D) Not dependency preserving and not lossless join.

Answer: (A)

Explanation: Schema $R(A, B, C, D)$ is decomposed into three relation $\rightarrow R_1(A, B)$, $R_2(B, C)$ and $R_3(B, D)$

Now dependencies derived from $R_1(A, B)$ are:

$A \rightarrow B$

$B \rightarrow C$ but C is not attribute here in this relation.

Dependencies derived from $R_2(B, C)$ are:

$B \rightarrow C$

$C \rightarrow D$

D is not the attribute in relation.

Dependencies derived from $R_3(B, D)$ are:

$D \rightarrow B$

$B \rightarrow C$

$C \rightarrow D$

All the dependencies are preserved and it is a lossless decomposition.

So, option (A) is correct.

Q. Consider a schema $R(MNPQ)$ and functional dependencies $M \rightarrow N, P \rightarrow Q$. Then the decomposition of R into $R_1(MN)$ and $R_2(PQ)$ is_____.

- (A) Dependency preserving but not lossless join
- (B) Dependency preserving and lossless join
- (C) Lossless join but not dependency preserving
- (D) Neither dependency preserving nor lossless join.

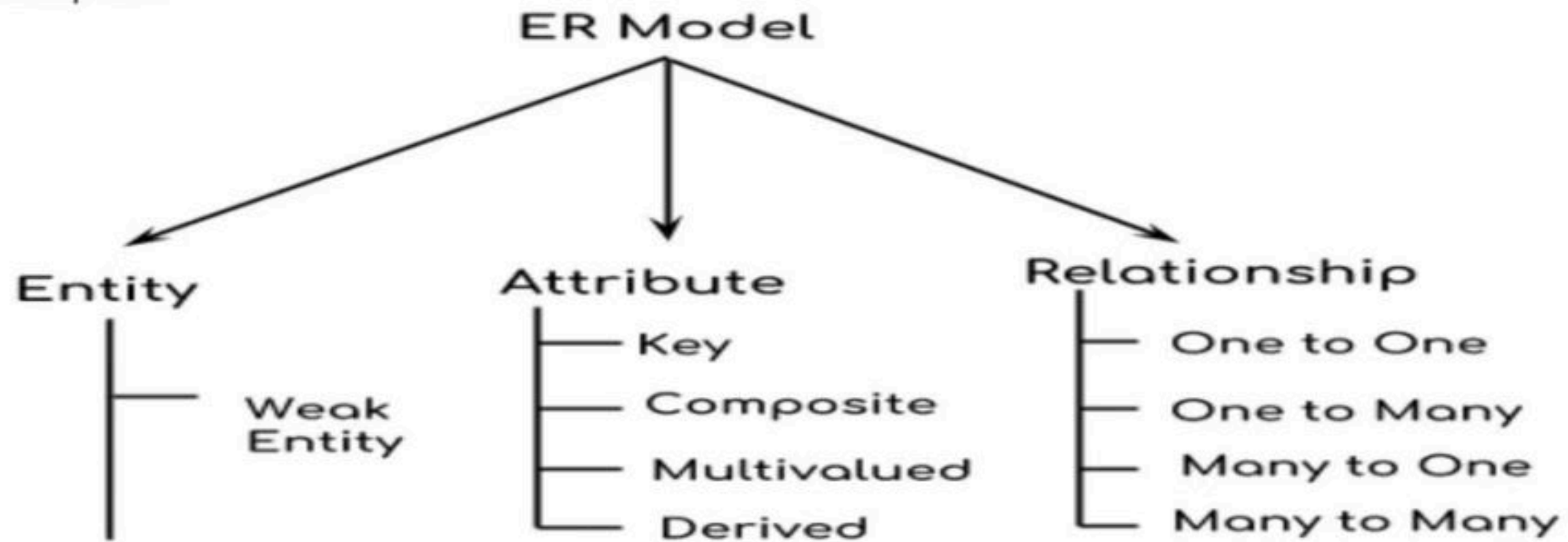
Answer: (A)

Explanation: Schema $R(MNPQ)$ is decomposed into $R_1(MN)$ $M \rightarrow N$ is preserved and $R_2(PQ)$ $P \rightarrow Q$ is also preserved, dependency will be preserved and there will be no loss of any dependency. So, option (A) is correct.

ER Diagram

Entity Relationship Diagram – ER Diagram in DBMS

An **Entity–relationship model (ER model)** describes the structure of a database with the help of a diagram, which is known as **Entity Relationship Diagram (ER Diagram)**. An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.



Components of ER Diagram

 Entity Set

 Relationship Set


 Weak Entity Set

 Weak Relationship Set


 attribute

 Composite Attribute


 multivalued Attribute

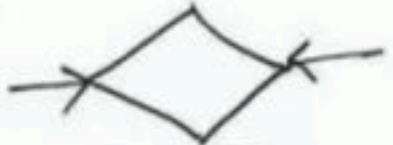
 derived Attribute


 Key Attribute


 Partial Key Attribute

 key constraint

 total participation

 1:1 (one to one)

 1:m (one to many)

 m:1 (many to one)

Match List I with List II.

List I (E-R symbols)

List II (Description)

(A)



(I) Key Attribute Type

(B)



(II) Weak Entity Type

(C)



(III) Total Participation of Entity in a relation

(D)



(IV) Multivalued Attribute type

Choose the correct answer from the options given below:

- (1) A-II, B-IV, C-III, D-I
- (2) A-IV, B-I, C-II, D-III
- (3) A-II, B-I, C-IV, D-III
- (4) A-III, B-IV, C-I, D-II

UGC NET 2020

Match List I with List II.

List I (E-R symbols)

List II (Description)

(A)



(I) Key Attribute Type

(B)



(II) Weak Entity Type

(C)



(III) Total Participation of Entity in a relation

(D)



(IV) Multivalued Attribute type

Choose the correct answer from the options given below:

(1) A-II, B-IV, C-III, D-I

(2) A-IV, B-I, C-II, D-III

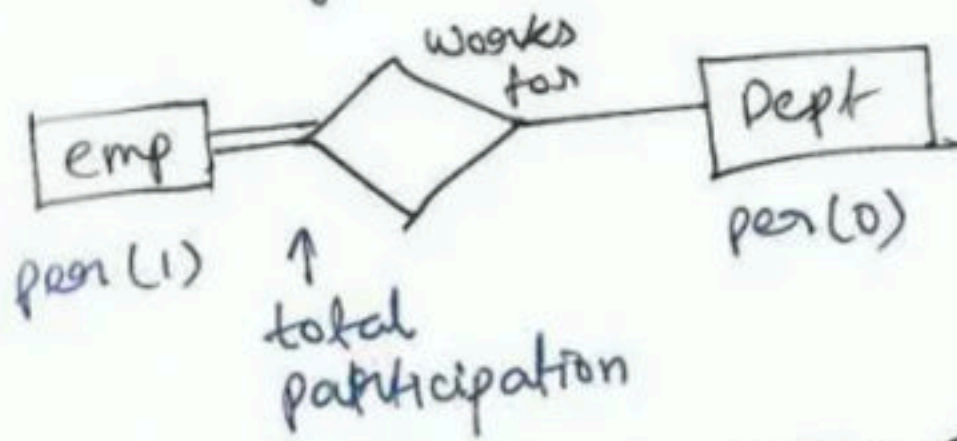
(3) A-II, B-I, C-IV, D-III

(4) A-III, B-IV, C-I, D-II

Correct Ans: Option 3

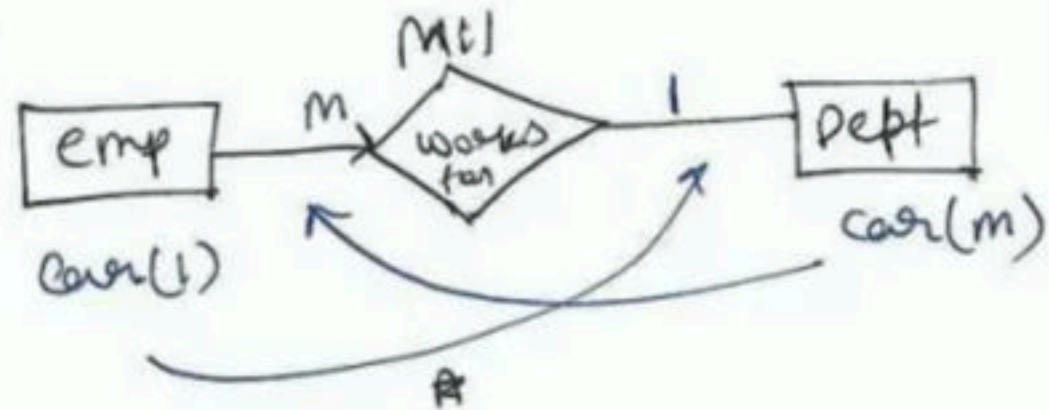
Participation

What is the minimum no of relations an entity can participate



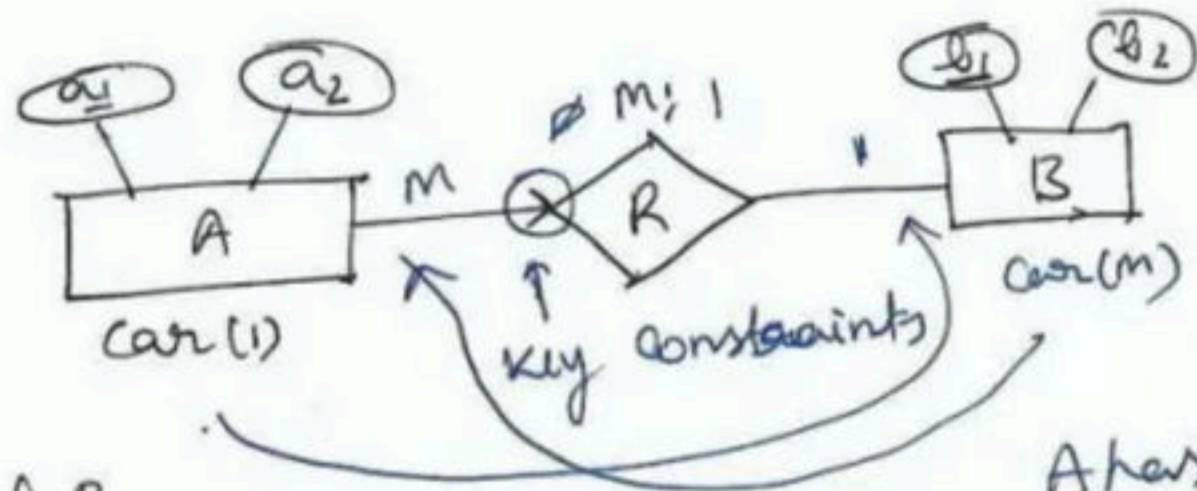
Cardinality

What is the max no of relations in which an entity can participate.



Key Constraints

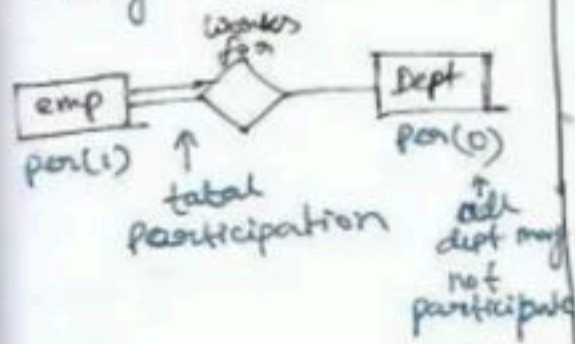
an entity A is associated with at most one entity of B



A has the key constraints so we can combine R with A.

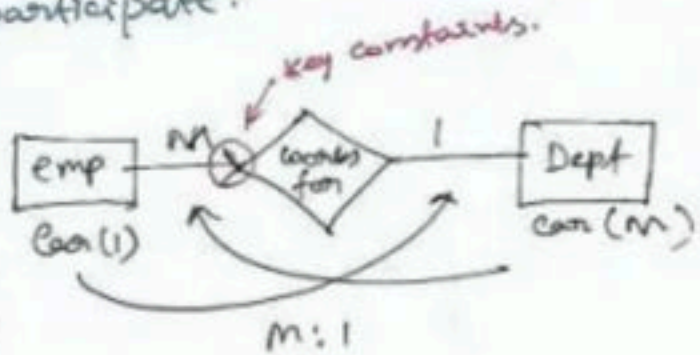
Participation

What is the minimum no of relations an entity can participate

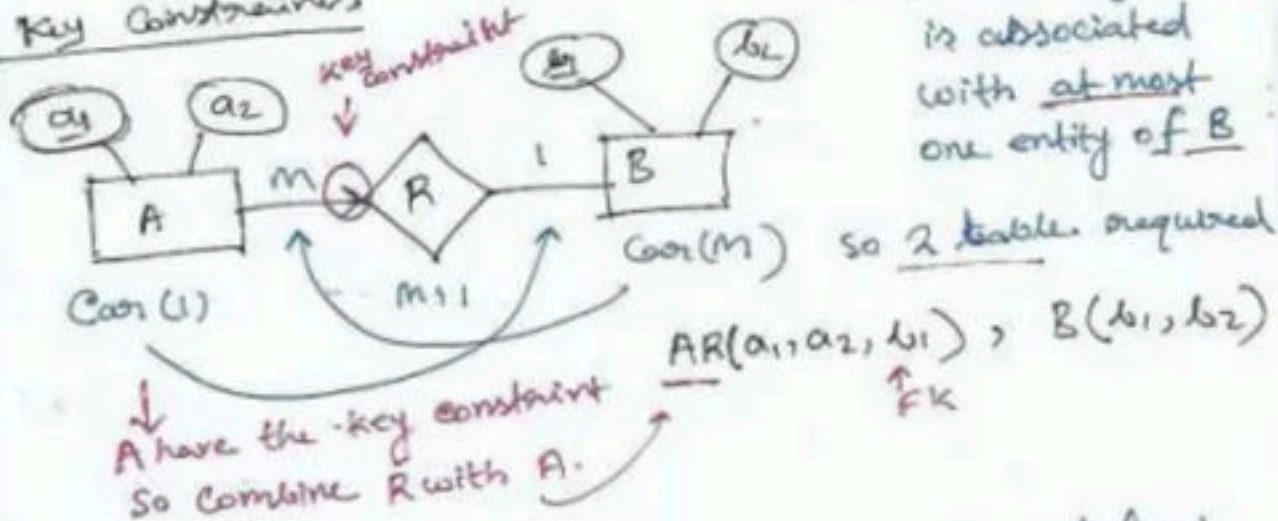


Cardinality

What is the max no of relations in which an entity can participate.



Key Constraints

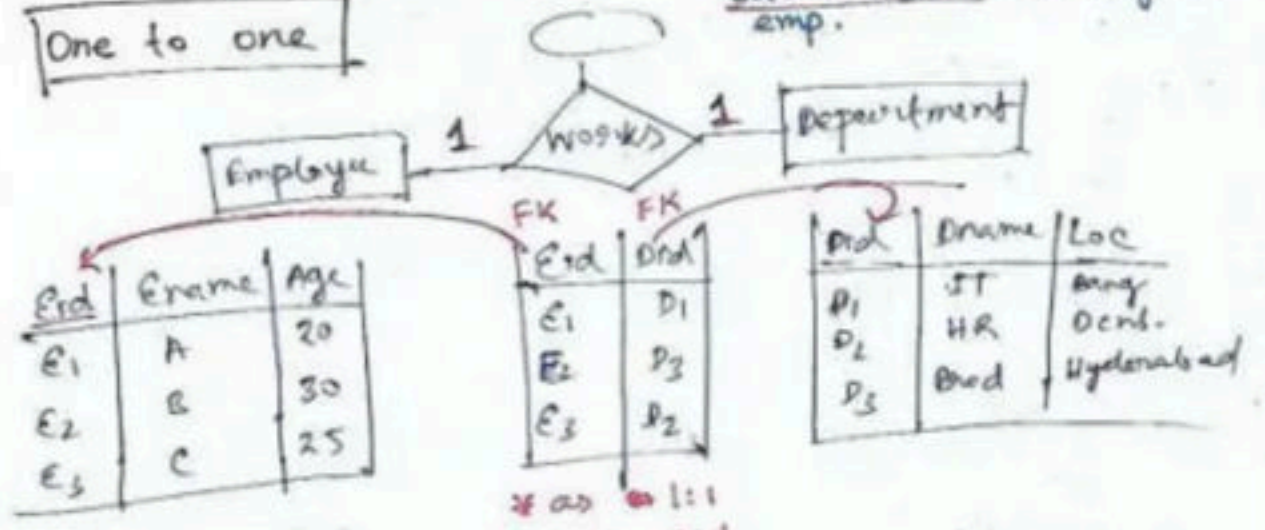


* Constraints about how few (at minimum) and how many (at maximum) individuals (objects) of one class may be connected to a single individual of the other class. This is called the multiplicity and cardinality. (ER term: participation)

Relational DB model supports Multiplicity of the association

One to one

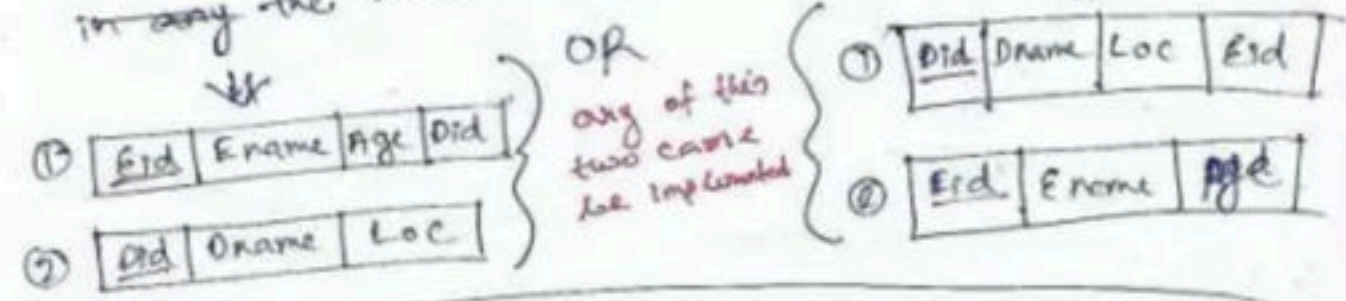
Dept
at most one entity of emp.



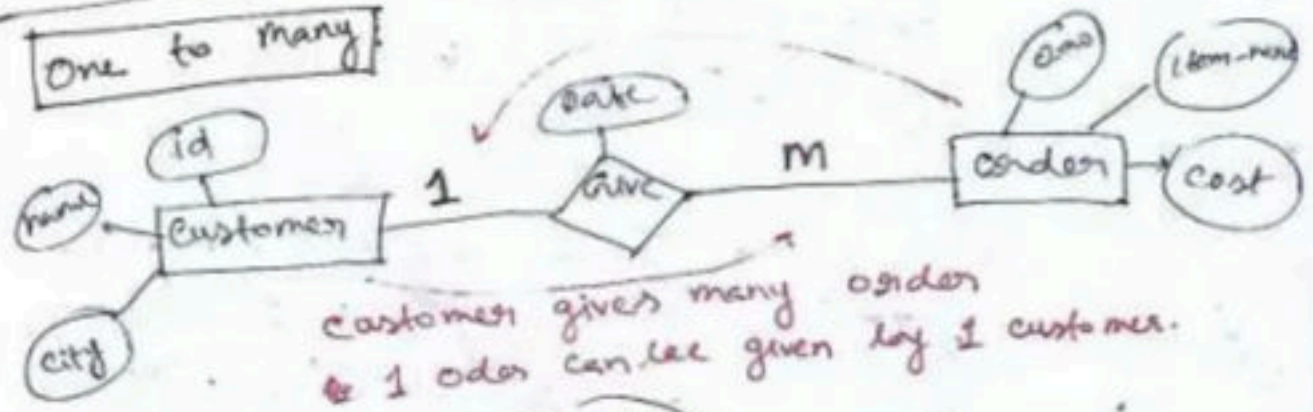
2 table req

as 1:1
So Erid, Drid any one can be primary key

can be reduced to two table, combined in any the relationship in any table.

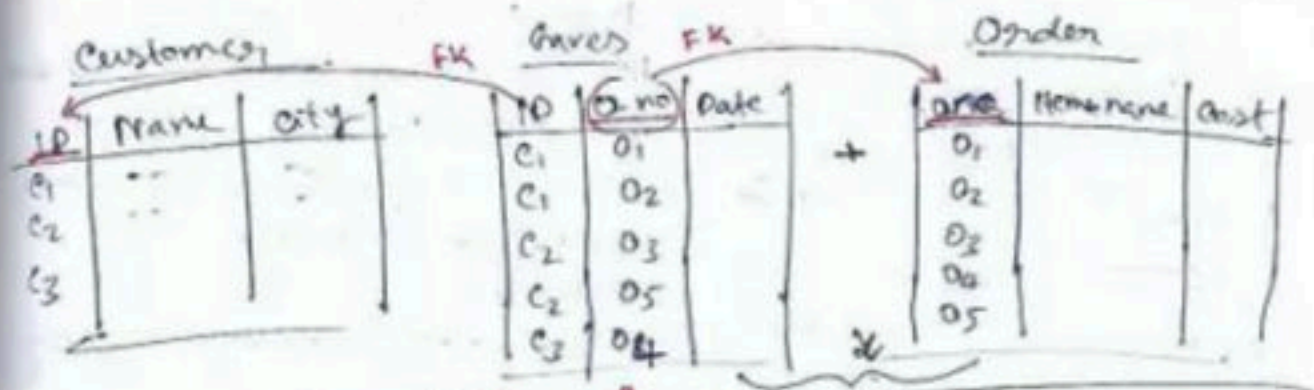


One to many



customer gives many order
1 order can be given by 1 customer.

2 table req



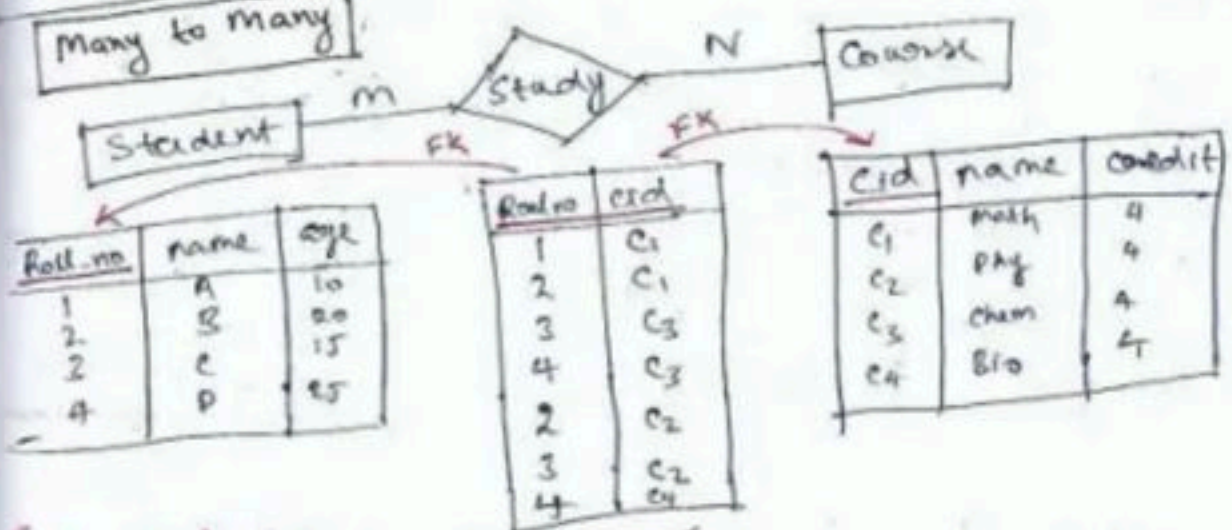
we can reduce table by merging give and order.

O.no can be PK

O.no	Item name	Cost	ID	Date
O1	C1	..
O2	C1	..
O3	C2	..
O4	C2	..
O5	C3	..

* m side can be merge with the relation

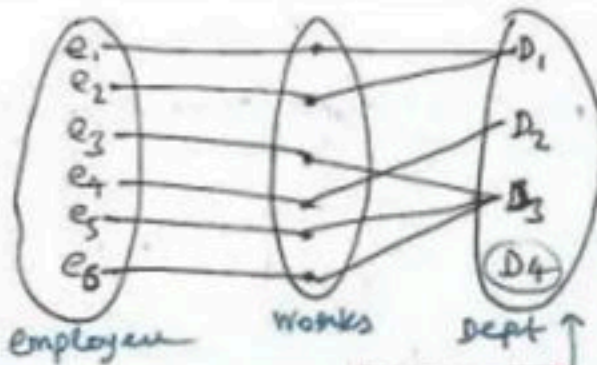
Many to Many



* No Reduction in (M-N) Relation

3 table req.

Can't make PK by any one of the attr have to combine RollNo - cid to make PK (Composite key)



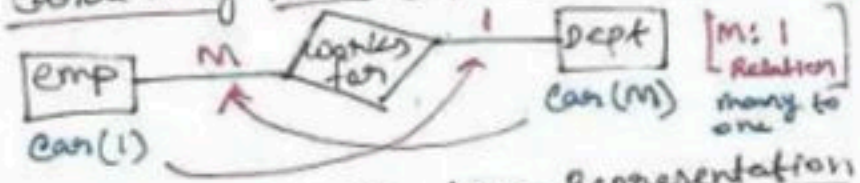
Degree of a Relationship:
 How many entity sets are participating
 Here Degree is 2.

Participation (existence constraints)
 What is the minimum no of relationship an entity can participate. (minimum cardinality)

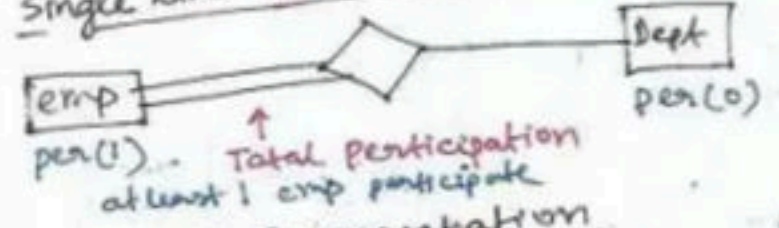
Employee works Dept
 Participation of emp = 1
 $per(emp) = 1$
 $can(emp) = 1$
 $per(dept) = 0$
 $can(dept) = 0$

Cardinality: Ratio
 What is the max no of relations in which an entity can participate

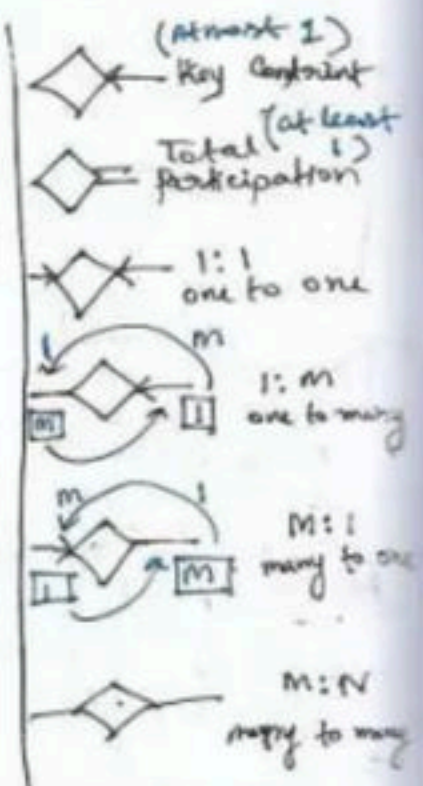
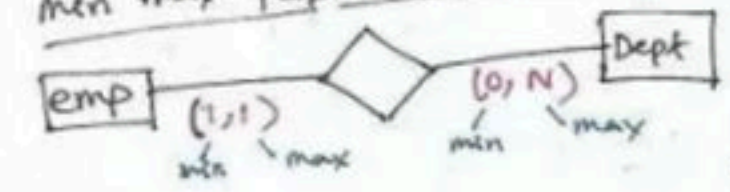
Cardinality Ratio Representation



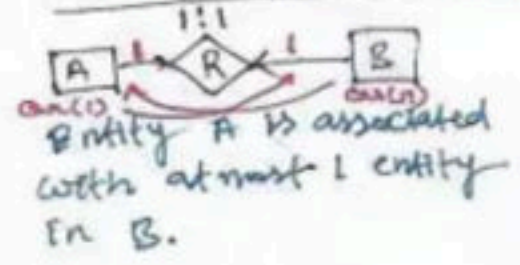
Single line Double line Representation



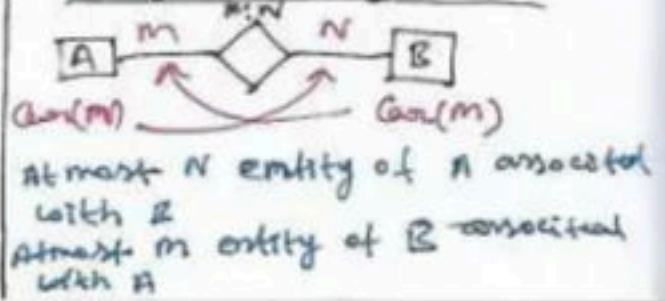
min max Representation

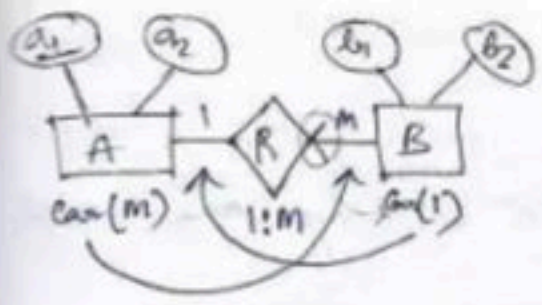


one to one Relationship

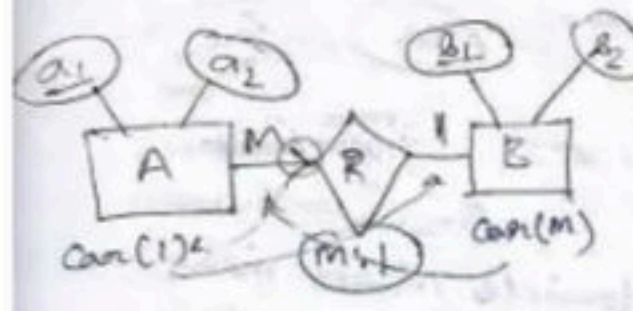


many to Many Relation

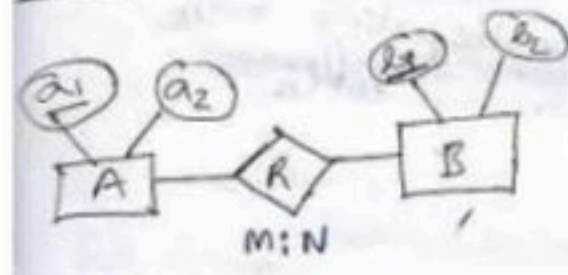




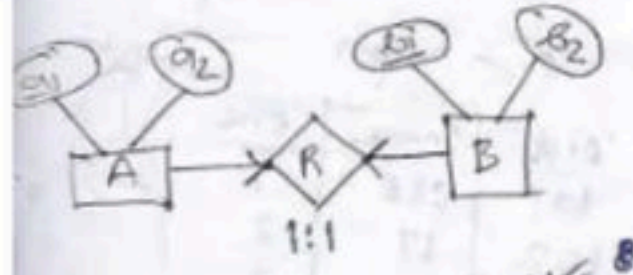
2 tables required
 $\Rightarrow A(a_1, a_2), BR(b_1, b_2, a_1)$
 as B have key constant
 so combine R with B
 FK (Foreign key)



2 tables required
 $\Rightarrow AR(a_1, a_2, b_1), B(b_1, b_2)$
 A have the key constant
 so combine R with A
 FK



3 tables required
 $A(a_1, a_2), B(b_1, b_2), R(a_1, b_1)$
 FK



2 tables required
 both have key constant
 and cardinality 1.
 so we can combine R
 with any of them.
 \swarrow $BAR(a_1, a_2, b_1), B(b_1, b_2)$
 OR
 \swarrow $A(a_1, a_2), BR(b_1, b_2, a_1)$
 FK

There are two types of participation constraints-

- 1.Total participation
- 2.Partial participation

1. Total Participation-

- It specifies that each entity in the entity set must compulsorily participate in at least one relationship instance in that relationship set.
- That is why, it is also called as **mandatory participation**.
- Total participation is represented using a double line between the entity set and relationship set.

2. Partial Participation-

- It specifies that each entity in the entity set may or may not participate in the relationship instance in that relationship set.
- That is why, it is also called as **optional participation**.
- Partial participation is represented using a single line between the entity set and relationship set.

Relationship between Cardinality and Participation Constraints-

Minimum cardinality tells whether the participation is partial or total.

- If minimum cardinality = 0, then it signifies partial participation.
- If minimum cardinality = 1, then it signifies total participation.

Maximum cardinality tells the maximum number of entities that participates in a relationship set.

Given the basic ER and relational models, which of the following is INCORRECT?

- (A) An attribute of an entity can have more than one value
- (B) An attribute of an entity can be composite
- (C) In a row of a relational table, an attribute can have more than one value
- (D) In a row of a relational table, an attribute can have exactly one value or a NULL value

Given the basic ER and relational models, which of the following is INCORRECT?

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

(B) An attribute of an entity can be composite

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

(D) In a row of a relational table, an attribute can have exactly one value or a NULL value

In an Entity-Relationship (ER) model, suppose R is a many-to-one relationship from entity set E_1 to entity set E_2 . Assume that E_1 and E_2 participate totally in R and that the cardinality of E_1 is greater than the cardinality of E_2 . Which one of the following is true about R ?

- (A) Every entity in E_1 is associated with exactly one entity in E_2 .
- (B) Some entity in E_1 is associated with more than one entity in E_2 .
- (C) Every entity in E_2 is associated with exactly one entity in E_1 .
- (D) Every entity in E_2 is associated with at most one entity in E_1 .

In an Entity-Relationship (ER) model, suppose R is a many-to-one relationship from entity set E_1 to entity set E_2 . Assume that E_1 and E_2 participate totally in R and that the cardinality of E_1 is greater than the cardinality of E_2 . Which one of the following is true about R ?

- (A) Every entity in E_1 is associated with exactly one entity in E_2 .
- (B) Some entity in E_1 is associated with more than one entity in E_2 .
- (C) Every entity in E_2 is associated with exactly one entity in E_1 .
- (D) Every entity in E_2 is associated with at most one entity in E_1 .

Given an instance of the STUDENTS relation as shown below:

<i>StudentID</i>	<i>StudentName</i>	<i>StudentEmail</i>	<i>StudentAge</i>	<i>CPI</i>
2345	Shankar	shankar@math	X	9.4
1287	swati	swati@ee	19	9.5
7853	shankar	shankar@cse	19	9.4
9876	swati	swati@mech	18	9.3
8765	ganesh	ganesh@civil	19	8.7

For **(*StudentName*,*StudentAge*)** to be a key for this instance, the value X should NOT be equal to _____.

Given an instance of the STUDENTS relation as shown below:

<i>StudentID</i>	<i>StudentName</i>	<i>StudentEmail</i>	<i>StudentAge</i>	<i>CPI</i>
2345	Shankar	shankar@math	X	9.4
1287	swati	swati@ee	19	9.5
7853	shankar	shankar@cse	19	9.4
9876	swati	swati@mech	18	9.3
8765	ganesh	ganesh@civil	19	8.7

For *(StudentName, StudentAge)* to be a key for this instance, the value X should NOT be equal to _____.

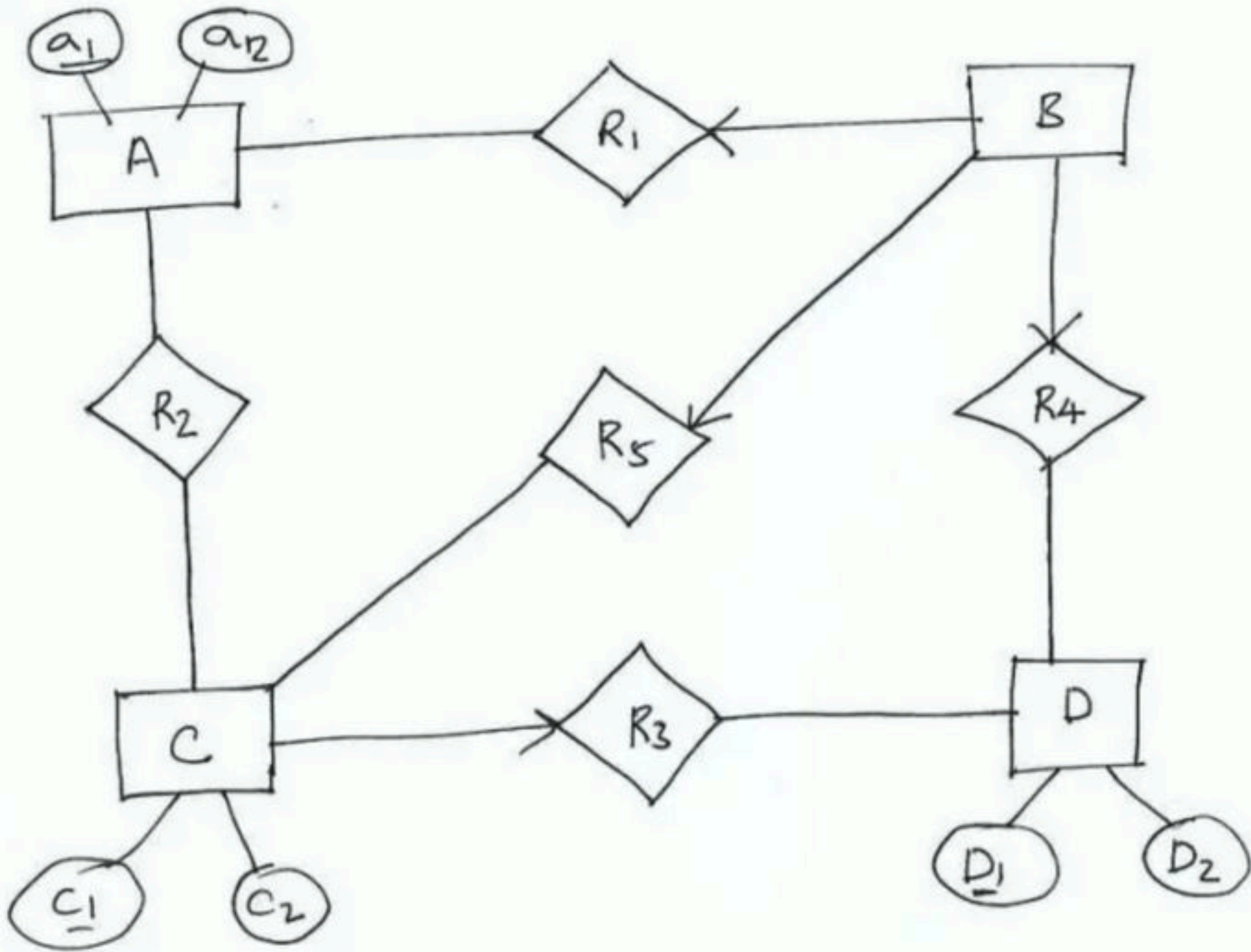
Ans: 19

An ER model of a database consists of entity types A and B. These are connected by a relationship R which does not have its own attribute. Under which one of the following conditions, can the relational table for R be merged with that of A?

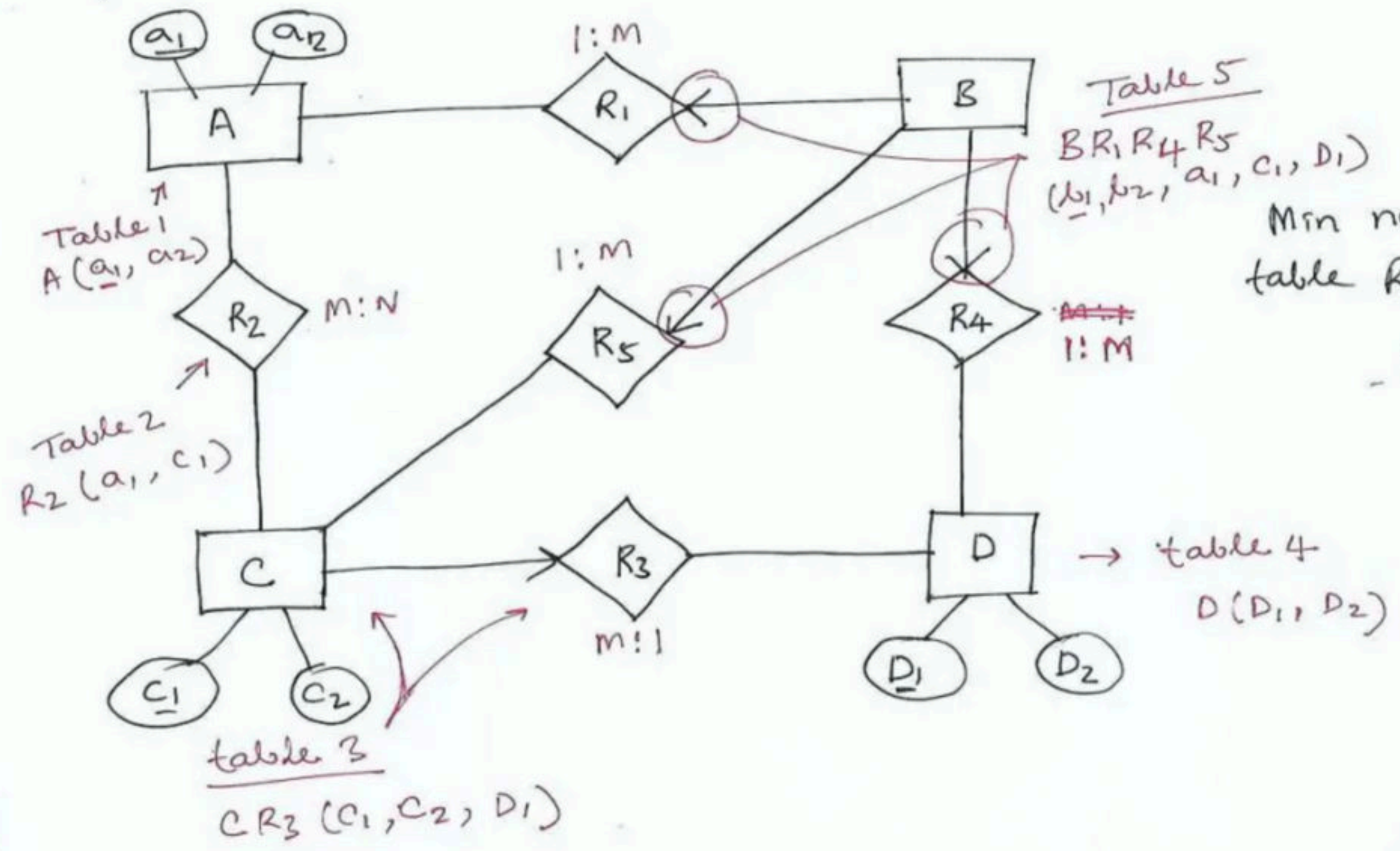
- (A) Relationship R is one-to-many and the participation of A in R is total.
- (B) Relationship R is one-to-many and the participation of A in R is partial.
- (C) Relationship R is many-to-one and the participation of A in R is total.
- (D) Relationship R is many-to-one and the participation of A in R is partial.

An ER model of a database consists of entity types A and B. These are connected by a relationship R which does not have its own attribute. Under which one of the following conditions, can the relational table for R be merged with that of A?

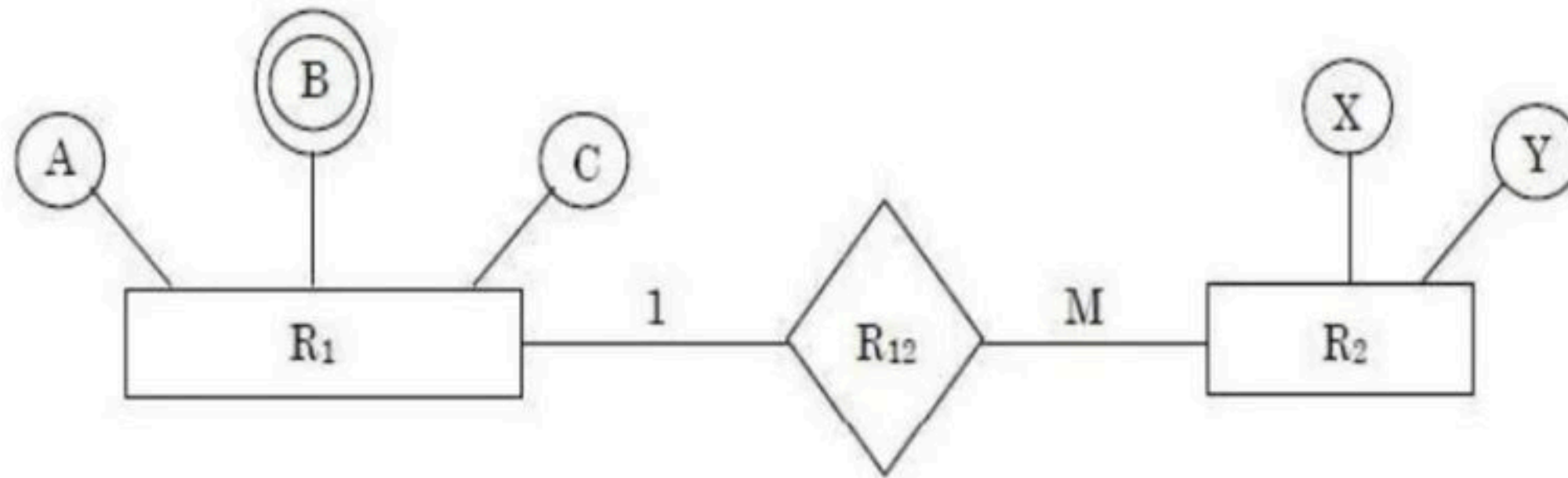
- (A) Relationship R is one-to-many and the participation of A in R is total.
- (B) Relationship R is one-to-many and the participation of A in R is partial.
- (C) Relationship R is many-to-one and the participation of A in R is total.**
- (D) Relationship R is many-to-one and the participation of A in R is partial.



Min no of
table Required?



Q.13 Find minimum number of tables required for converting the following entity relationship diagram into relational database?



(1) 2

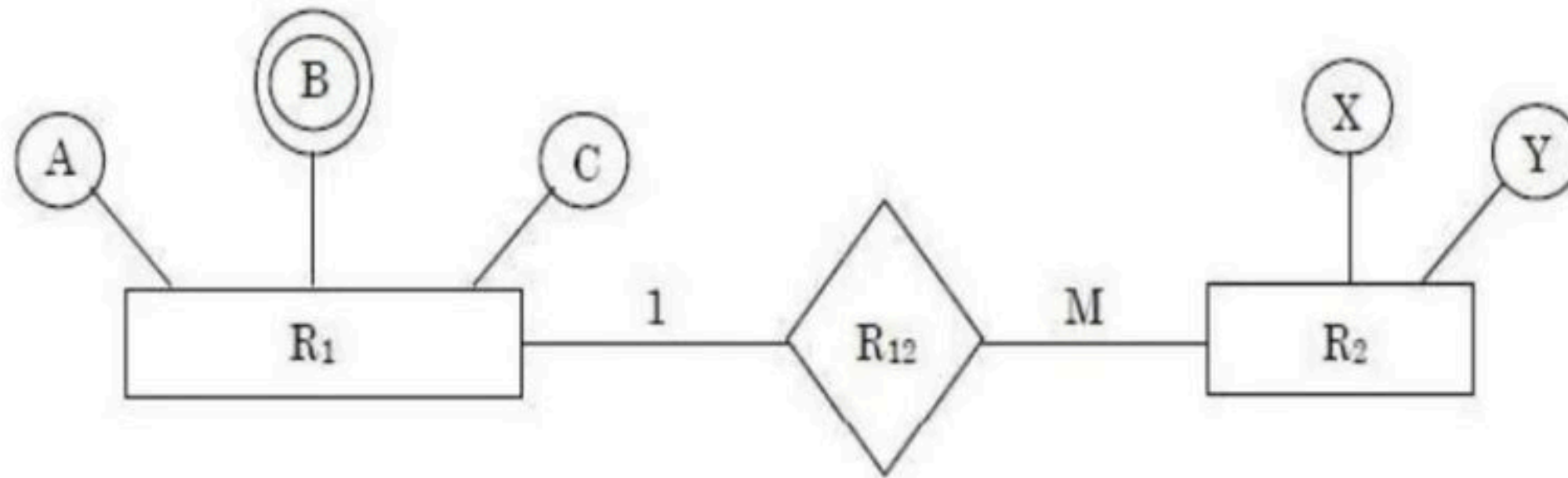
(2) 4

(3) 3

(4) 5

NET Dec 2019

Q.13 Find minimum number of tables required for converting the following entity relationship diagram into relational database?



(1) 2

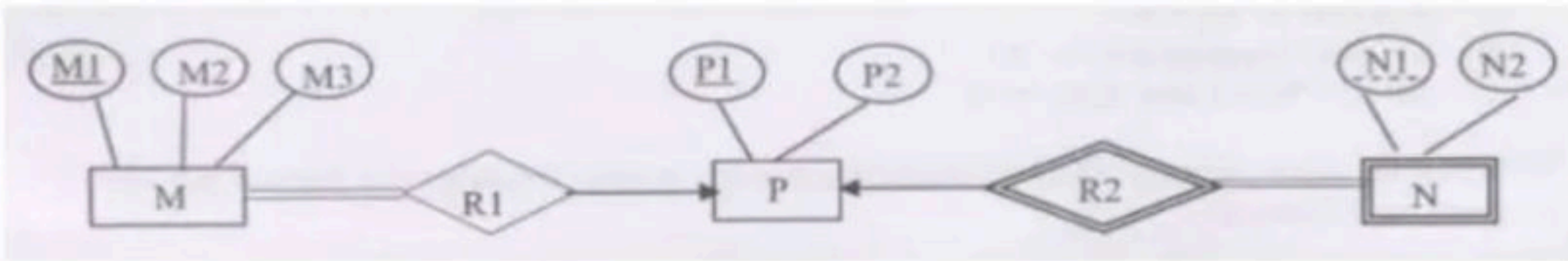
(3) 3

(2) 4

(4) 5



Consider the following ER diagram



The minimum number of tables needed to represent M, N, P, R1, R2 is

- (A) 2
- (B) 3
- (C) 4
- (D) 5

First strong entity types are made to tables. So, we get two tables M and P
assume R1 is 1:1 or 1:n
as that would minimize the number of tables as asked in question.

Now participation of M
in R1 is total (indicated by double arrow) meaning every entity of M participate in R1.
Since R1 is not having an attribute, we can simple add the primary key of P to the table
M and add a foreign key reference to M. This handles R1 and we don't need an extra
table. So, M becomes M1,M2,M3,P1

N here is a weak entity weakly related to P
. So, we form a new table N, and includes the primary key of P(P1) as foreign key
reference. Now (P1,N1) becomes the primary key of N

Thus we get 3 tables.

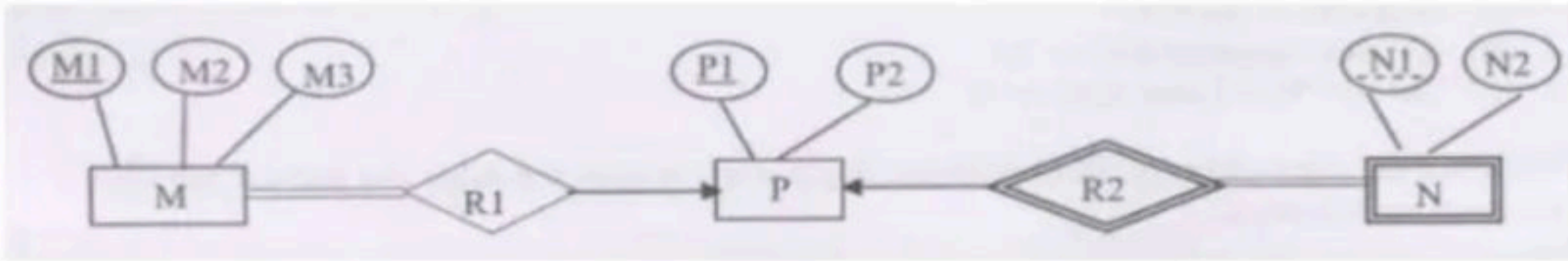
M:M1,M2,M3,P1 - M1 primary key, P1 references P

P:P1,P2 - P1 primary key

N:P1,N1,N2 - (P1,N1) primary key, P1 references P

So, answers is B.

Consider the following ER diagram



The minimum number of tables needed to represent M, N, P, R1, R2 is

- (A) 2
- (B) 3**
- (C) 4
- (D) 5

Consider an Entity-Relationship (ER) model in which entity sets E1 and E2 are connected by an $m : n$ relationship R_{12} , E1 and E3 are connected by a $1 : n$ (1 on the side of E1 and n on the side of E3) relationship R_{13} .

E1 has two single-valued attributes a_{11} and a_{12} of which a_{11} is the key attribute. E2 has two single-valued attributes a_{21} and a_{22} of which a_{22} is the key attribute. E3 has two single-valued attributes a_{31} and a_{32} of which a_{31} is the key attribute. The relationships do not have any attributes.

If a relational model is derived from the above ER model, then the minimum number of relations that would be generated if all the relations are in 3NF is _____.

- (A) 2
- (B) 3
- (C) 4
- (D) 5

Consider an Entity-Relationship (ER) model in which entity sets E1 and E2 are connected by an $m : n$ relationship R_{12} , E1 and E3 are connected by a $1 : n$ (1 on the side of E1 and n on the side of E3) relationship R_{13} .

E1 has two single-valued attributes a_{11} and a_{12} of which a_{11} is the key attribute. E2 has two single-valued attributes a_{21} and a_{22} of which a_{22} is the key attribute. E3 has two single-valued attributes a_{31} and a_{32} of which a_{31} is the key attribute. The relationships do not have any attributes.

If a relational model is derived from the above ER model, then the minimum number of relations that would be generated if all the relations are in 3NF is _____.

- (A) 2
- (B) 3
- (C) 4
- (D) 5

Which of the following $(\min(M), \max(N))$ conditions means a partial participation of the relation in a relationship?

- A. 0:N
- B. 0:M
- C. 1:N
- D. Both A and B above

Which of the following $(\min(M), \max(N))$ conditions means a partial participation of the relation in a relationship?

- A. 0:N
- B. 0:M
- C. 1:N
- D. Both A and B above

Relational Algebra and SQL

With Practical Example and PYQs

Introduction of Relational Algebra in DBMS

- **Relational Algebra** is procedural query language, which takes **Relation as input and generate relation as output**. Relational algebra mainly provides theoretical foundation for relational databases and SQL.
- It uses **operators** to perform queries. An operator can be either **unary or binary**.
- Relational algebra is **performed recursively** on a relation and **intermediate results are also considered relations**.

Basic Relational Algebra Operations:

Relational Algebra divided in various groups

Unary Relational Operations

- SELECT (symbol: σ)
- PROJECT (symbol: π)
- RENAME (symbol:)

Relational Algebra Operations From Set Theory

- UNION (\cup)
- INTERSECTION (\cap),
- DIFFERENCE ($-$)
- CARTESIAN PRODUCT (\times)

Binary Relational Operations

- JOIN
- DIVISION

Q)With respect to relational algebra, which of the following operations are included from mathematical set theory

- (a) join
- (b) Intersection
- (c) Cartesian product
- (d) Project

Option

- A) (a) and (b)
- B) (b) and (c)
- C) (c) and (d)
- D) (b) and (d)

Q)With respect to relational algebra, which of the following operations are included from mathematical set theory

- (a) join
- (b) Intersection
- (c) Cartesian product
- (d) Project

Option

A) (a) and (b)

B) (b) and (c)

C) (c) and (d)

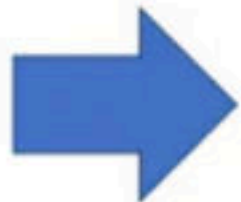
D) (b) and (d)

Operators in Relational Algebra

Projection (π)

Projection is used to project required column data from a relation.

Example :

R		π (BC)
(A B C)		B C
-----		-----
1 2 4		2 4
2 2 3		2 3
3 2 3		3 4
4 3 4		

Note: By Default projection removes duplicate data.

Selection (σ)

Selection is used to select required tuples of the relations.
for the above relation

$$\sigma (c>3)R$$

will select the tuples which have c more than 3.

Note: selection operator only selects the required tuples but does not display them. For displaying, data projection operator is used.

For the above selected tuples, to display we need to use projection also.

Example :

R
(A B C)

1 2 4
2 2 3
3 2 3
4 3 4



$\pi (\sigma (c>3)R)$ will
show following tuples.

A B C

1 2 4
4 3 4

Union compatibility: The two relations are said to be **union compatible** if both the relations have the same number of attributes and the domain of the similar attributes is same.

Union, Intersection and Difference operation required Union Compatibility.

EG: Consider the following relations:

- **Person**(FirstName, LastName)
- **Country**(Name, Population)

In this case, Person and Country are not union-compatible as they do not share the same set of attributes, even though they share the same amount of attributes.

Union Operation (U)

It performs binary union between two given relations and is defined as

Notation – $r \cup s$

Where r and s are either database relations or relation result set (temporary relation). For a union operation to be valid, the following conditions must hold

- r , and s must have the same number of attributes.
- Attribute domains must be compatible or Union compatible.
- Duplicate tuples are automatically eliminated.

Example

Consider the following tables.

Table A

column 1	column 2
1	1
1	2

Table B

column 1	column 2
1	1
1	3

$A \cup B$ gives

Table $A \cup B$

column 1	column 2
1	1
1	2
1	3

Set Difference ($-$)

The result of set difference operation is tuples, which are present in one relation but are not in the second relation.

- Symbol denotes it. The result of $A - B$, is a relation which includes all tuples that are in A but not in B .

- The attribute name of A has to match with the attribute name in B .
- The two-operand relations A and B should be either compatible or Union compatible.
- It should be defined relation consisting of the tuples that are in relation A , but not in B .

Example

Consider the following tables.

Table A

column 1	column 2
1	1
1	2

A-B

Table A - B

column 1	column 2
1	2

Table B

column 1	column 2
1	1
1	3

Intersection

An intersection is defined by the symbol \cap

$A \cap B$

Defines a relation consisting of a set of all tuple that are in both A and B. However, A and B must be union-compatible.

Example:

Table A

column 1	column 2
1	1
1	2

Table A \cap B

column 1	column 2
1	1

Table B

column 1	column 2
1	1
1	3

Cross Product OR Cartesian product (X)

Cross product between two relations let say A and B, so cross product between A X B will results all the attributes of A followed by each attribute of B. Each record of A will pairs with every record of B. below is the example

A		
(Name	Age	Sex)
Ram	14	M
Sona	15	F
kim	20	M

B	
(Id	Course)
1	DS
2	DBMS

A X B				
Name	Age	Sex	Id	Course
Ram	14	M	1	DS
Ram	14	M	2	DBMS
Sona	15	F	1	DS
Sona	15	F	2	DBMS
Kim	20	M	1	DS
Kim	20	M	2	DBMS

Note: if A has 'n' tuples and B has 'm' tuples then A X B will have 'n*m' tuples.

Natural Join (\bowtie)

Natural join is a binary operator. Natural join between two or more relations will result set of all combination of tuples where they have **equal common attribute**.

Let us see below example

Emp		
(Name	Id	Dept_name)
A	120	IT
B	125	HR
C	110	Sale
D	111	IT

Dep	
(Dept_name	Manager)
Sale	Y
Prod	Z
IT	A

Emp \bowtie Dep

Name	Id	Dept_name	Manager
A	120	IT	A
C	110	Sale	Y
D	111	IT	A

Q)

Consider the following relation schema pertaining to a students database:

Student (rollno, name, address)

Enroll (rollno, courseno, coursename)

where the primary keys are shown underlined. The number of tuples in the Student and Enroll tables are 120 and 8 respectively. What are the maximum and minimum number of tuples that can be present in (Student * Enroll), where '*' denotes natural join ?

(A) 8, 8

(B) 120, 8

(C) 960, 8

(D) 960, 120

Q)

Consider the following relation schema pertaining to a students database:

Student (rollno, name, address)

Enroll (rollno, courseno, coursename)

where the primary keys are shown underlined. The number of tuples in the Student and Enroll tables are 120 and 8 respectively. What are the maximum and minimum number of tuples that can be present in (Student * Enroll), where '*' denotes natural join ?

(A) 8, 8

(B) 120, 8

(C) 960, 8

(D) 960, 120

Rollno in students is key, students table has 120 tuples,

In Enroll table rollno is FK referencing to Students table.

In natural join it'll return the records where the rollno value of enroll matches with the rollno of students

so, in both conditions min and max records will be resulted (8,8).(8,8).

hence A is the answer.

Explanation: The result of the natural join is the set of all combinations of tuples in R and S that are equal on their common attribute names.

What is the maximum possible number of tuples?

The result of natural join becomes equal to the Cartesian product when there are no common attributes. The given tables have a common attribute, and Primary key Foreign Key Relationship. so the result of natural join is 8.

What is the minimum possible number of tuples?

It might be possible that there is no rollno common or null values .
In that case, the number of tuples would be 0.

But in the question rollno is a primary key in Student table and it is a prime attribute in the Enroll table.

So there is no chance of having null values in the rollno column in enroll table and Student table and every tuple in Enroll will have a parent.

So minimum number of tuples possible are 8 (only 8 tuples in the Enroll table).

Option (A) is correct.

Q.18

Given two tables $R1(x, y)$ and $R2(y, z)$ with 50 and 30 number of tuples respectively. Find maximum number of tuples in the output of natural join between tables $R1$ and $R2$ i.e.

$R1 \bowtie R2$? (# - Natural Join)

(1) 30

(2) 20

(3) 50

(4) 1500

Options 1. 1

2. 2

3. 3

4. 4

Q.18

Given two tables $R1(x, y)$ and $R2(y, z)$ with 50 and 30 number of tuples respectively. Find maximum number of tuples in the output of natural join between tables $R1$ and $R2$ i.e. $R1 \bowtie R2$? (* - Natural Join)

(1) 30

(2) 20

(3) 50

(4) 1500

Options 1. 1

2. 2

3. 3

4. 4

Question Type : MCQ

Question ID : 61547510511

Option 1 ID : 61547540981

Option 2 ID : 61547540982

Option 3 ID : 61547540983

Option 4 ID : 61547540984

Status : Answered

Chosen Option : 4

National Testing Agency

UGC NET DECEMBER 2019

Exam Date :04.12.2019

Final Answer Key on which result compiled and declared on 31.12.2019 Shift:First

Subject :(087) Computer Science and Applications

QUESTION ID	CORRECT OPTION(S) ID	QUESTION ID	CORRECT OPTION(S) ID	QUESTION ID	CORRECT OPTION(S) ID
61547510439	61547540702	61547510496	61547540922	61547510550	61547541138
61547510440	61547540705	61547510497	61547540925	61547510551	61547541142
61547510441	61547540711	61547510498	61547540930	61547510552	61547541146
61547510442	61547540716	61547510499	61547540936	61547510553	61547541150
61547510443	61547540718	61547510500	61547540939	61547510554	61547541154
61547510444	61547540724	61547510501	61547540941	61547510555	61547541160
61547510445	61547540725	61547510502	61547540948	61547510556	61547541163
61547510446	61547540732	61547510503	61547540952	61547510557	61547541168
61547510447	61547540733	61547510504	61547540953	61547510558	61547541171
61547510448	61547540739	61547510505	61547540960	61547510559	61547541174
61547510449	61547540741	61547510506	61547540962	61547510560	61547541179
61547510450	61547540748	61547510507	61547540966	61547510561	61547541183
61547510451	61547540750	61547510508	61547540970	61547510562	61547541187
61547510452	61547540755	61547510509	61547540975	61547510563	61547541192
61547510453	61547540760	61547510510	61547540978	61547510564	61547541193
61547510454	61547540761	61547510511	61547540984	61547510565	61547541197
61547510455	61547540767	61547510512	61547540986	61547510566	61547541202

In this case A cartesian product of two tables will be returned.

This is because when we perform any JOIN operation on two tables a **cartesian product** of those tables is performed and then based on any select condition **in WHERE clause** the resultant rows are returned.

But **if there are no common columns or no common data** the process stops after Cartesian product.

Maximum and Minimum No of Tuples after Natural Join

The result of the natural join is the set of all combinations of tuples in R and S that are equal on their common attribute names.

What is the maximum possible number of tuples?

The result of natural join becomes equal to the **Cartesian product** when there are **no common attributes and no common data** or if nothing is mentioned

What is the maximum possible number of tuples?

The result of natural join becomes equal to the **Cartesian product** when there are **no common attributes and no common data.**

What is the minimum possible number of tuples?

For **minimum can be zero** becoz we have to take worst case condition if nothing is mentioned.....

Conditional Join

Conditional join works similar to natural join. In natural join, by default condition is equal between common attribute while in conditional join we can specify the any condition such as greater than, less than, not equal

Let us see below example

R		
(ID	Sex	Marks)
1	F	45
2	F	55
3	F	60

S		
(ID	Sex	Marks)
10	M	20
11	M	22
12	M	59

Join between R And S with condition **R.marks >= S.marks**

R.ID	R.Sex	R.Marks	S.ID	S.Sex	S.Marks
1	F	45	10	M	20
1	F	45	11	M	22
2	F	55	10	M	20
2	F	55	11	M	22
3	F	60	10	M	20
3	F	60	11	M	22
3	F	60	12	M	59

Join Operations:

A Join operation combines related tuples from different relations, if and only if a given join condition is satisfied. It is denoted by \bowtie .

EMPLOYEE

EMP_CODE	EMP_NAME
101	Stephan
102	Jack
103	Harry

SALARY

EMP_CODE	SALARY
101	50000
102	30000
103	25000

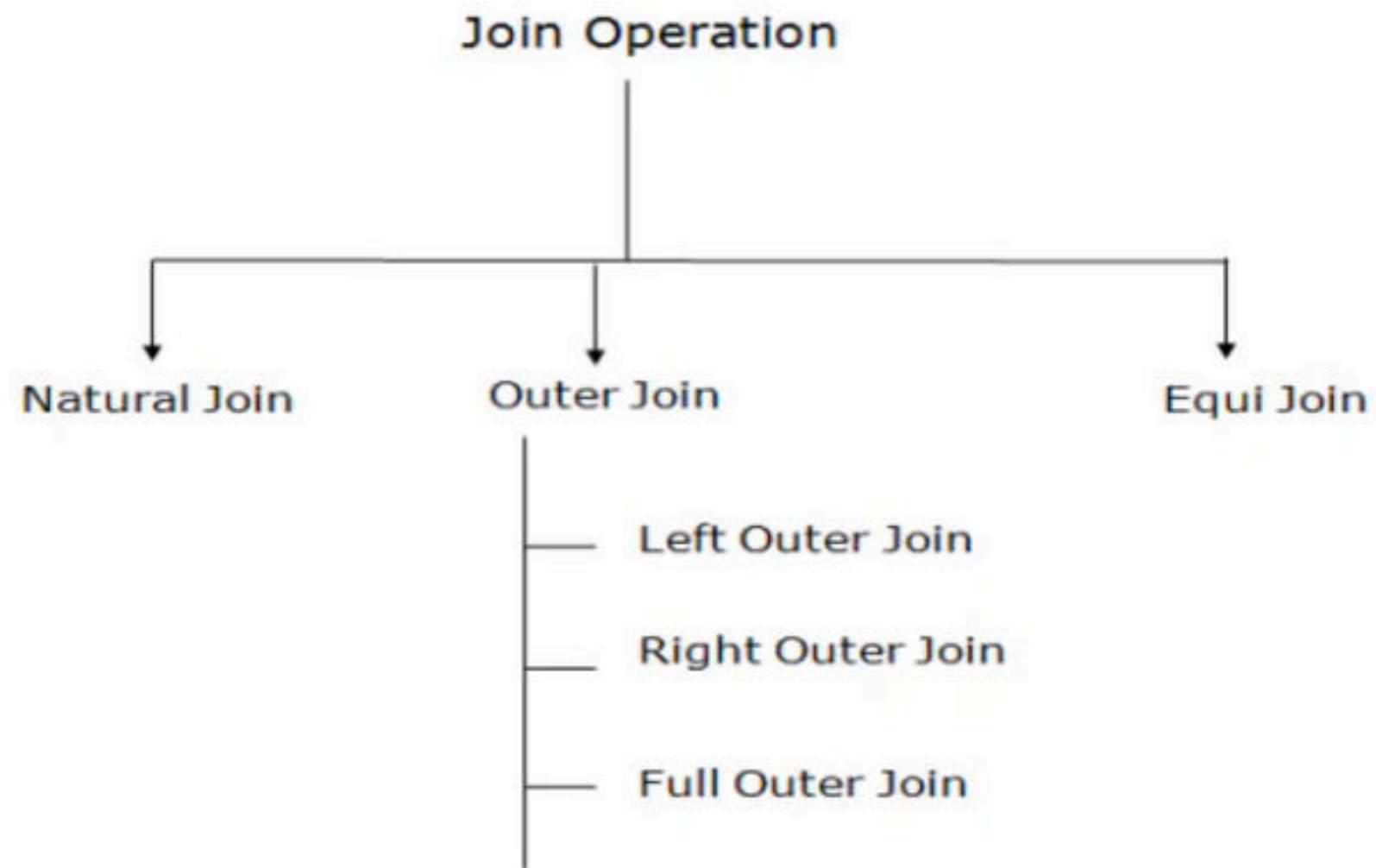
Operation: (EMPLOYEE ⋈ SALARY)



Result:

EMP_CODE	EMP_NAME	SALARY
101	Stephan	50000
102	Jack	30000
103	Harry	25000

Types of Join operations:



OUTER JOINS

Notice that much of the data is lost when applying a join to two relations. In some cases this lost data might hold useful information. An outer join retains the information that would have been lost from the tables, replacing missing data with nulls.

There are three forms of the outer join, depending on which data is to be kept.

- **LEFT OUTER JOIN** - keep data from the left-hand table
- **RIGHT OUTER JOIN** - keep data from the right-hand table
- **FULL OUTER JOIN** - keep data from both tables

R ColA ColB

A	1
B	2
D	3
F	4
E	5

R LEFT OUTER JOIN

R.ColA = S.SColA

S

A	1	A	1
D	3	D	3
E	5	E	4
B	2	-	-
F	4	-	-

S SColA SColB

A	1
C	2
D	3
E	4

R RIGHT OUTER JOIN

R.ColA = S.SColA

S

A	1	A	1
D	3	D	3
E	5	E	4
-	-	C	2

R *Co1A* *Co1B*

A	1
B	2
D	3
F	4
E	5

S *S*Co1A *S*Co1B

A	1
C	2
D	3
E	4

R FULL OUTER JOIN R.Co1A = S.SCo1A S

A	1	A	1
D	3	D	3
E	5	E	4
B	2	-	-
F	4	-	-
-	-	C	2

Consider two relations $R_1(A,B)$ with the tuples $(1,5)$, $(3,7)$ and $R_2(A,C) = (1,7)$, $(4,9)$. Assume that $R(A,B,C)$ is the full natural outer join of R_1 and R_2 . Consider the following tuples of the form (A,B,C) : $a = (1,5,\text{null})$, $b = (1,\text{null},7)$, $c = (3,\text{null},9)$, $d = (4,7,\text{null})$, $e = (1,5,7)$, $f = (3,7,\text{null})$, $g = (4,\text{null},9)$. Which one of the following statements is correct?

R contains a,b,e,f,g but not c, d .

R contains all of a,b,c,d,e,f,g

R contains e,f,g but not a,b

R contains e but not f,g

★ R_1

A	B
1	5
3	7

★ R_2

A	C
1	7
4	7

★ $R = R_1 \bowtie R_2$

A	B	C
1	5	7
3	7	Null
4	Null	9

So, from the above resultant table, R contains e, f, g only but not a, b.

Consider two relations $R_1(A,B)$ with the tuples $(1,5)$, $(3,7)$ and $R_2(A,C) = (1,7)$, $(4,9)$. Assume that $R(A,B,C)$ is the full natural outer join of R_1 and R_2 . Consider the following tuples of the form (A,B,C) : $a = (1.5, \text{null})$, $b = (1, \text{null}, 7)$, $c = (3, \text{null}, 9)$, $d = (4, 7, \text{null})$, $e = (1, 5, 7)$, $f = (3, 7, \text{null})$, $g = (4, \text{null}, 9)$. Which one of the following statements is correct?

R contains a,b,e,f,g but not c, d.

R contains all of a,b,c,d,e,f,g

R contains e,f,g but not a,b

R contains e but not f,g

Q.

Consider the following relations P(X,Y,Z), Q(X,Y,T) and R(Y,V).

P		
X	Y	Z
X1	Y1	Z1
X1	Y1	Z2
X2	Y2	Z2
X2	Y4	Z4

Q		
X	Y	T
X2	Y1	2
X1	Y2	5
X1	Y1	6
X3	Y3	1

R	
Y	V
Y1	V1
Y3	V2
Y2	V3
Y2	V2

How many tuples will be returned by the following relational algebra query?

$$\Pi_X(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)) - \Pi_X(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

Options:

- A 0
- B 1
- C 2
- D 3

Q.

Consider the following relations P(X,Y,Z), Q(X,Y,T) and R(Y,V).

P		
X	Y	Z
X1	Y1	Z1
X1	Y1	Z2
X2	Y2	Z2
X2	Y4	Z4

Q		
X	Y	T
X2	Y1	2
X1	Y2	5
X1	Y1	6
X3	Y3	1

R	
Y	V
Y1	V1
Y3	V2
Y2	V3
Y2	V2

How many tuples will be returned by the following relational algebra query?

$$\Pi_X(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)) - \Pi_X(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

Options:

- A 0
- B 1
- C 2
- D 3

$$\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)$$

X	Y	Z	V
X2	Y2	Z2	V2

$$\Pi_x(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R))$$

X
X2

$$\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R)$$

X	Y	T	V
X1	Y1	6	V1
X1	Y2	5	V3
X1	Y2	5	V2

$$\Pi_x(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

X
X1

$$\Pi_x(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)) - \Pi_x(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

$$= \begin{array}{|c|} \hline X \\ \hline X2 \\ \hline \end{array} - \begin{array}{|c|} \hline X \\ \hline X1 \\ \hline \end{array} = \begin{array}{|c|} \hline X \\ \hline X2 \\ \hline \end{array}$$

Only One Tuple will be returned

Consider the relations $r(A, B)$ and $s(B, C)$, where $s.B$ is a primary key and $r.B$ is a foreign key referencing $s.B$. Consider the query

Q: $r \bowtie (\sigma_{B < 5}(s))$

Let LOJ denote the natural left outer-join operation. Assume that r and s contain no null values. Which one of the following is NOT equivalent to Q?

- A $\sigma_{B < 5}(r \bowtie s)$
- B $\sigma_{B < 5}(r \text{ LOJ } s)$
- C $r \text{ LOJ } (\sigma_{B < 5}(s))$
- D $\sigma_{B < 5}(r) \text{ LOJ } s$

Consider the relations $r(A, B)$ and $s(B, C)$, where $s.B$ is a primary key and $r.B$ is a foreign key referencing $s.B$. Consider the query

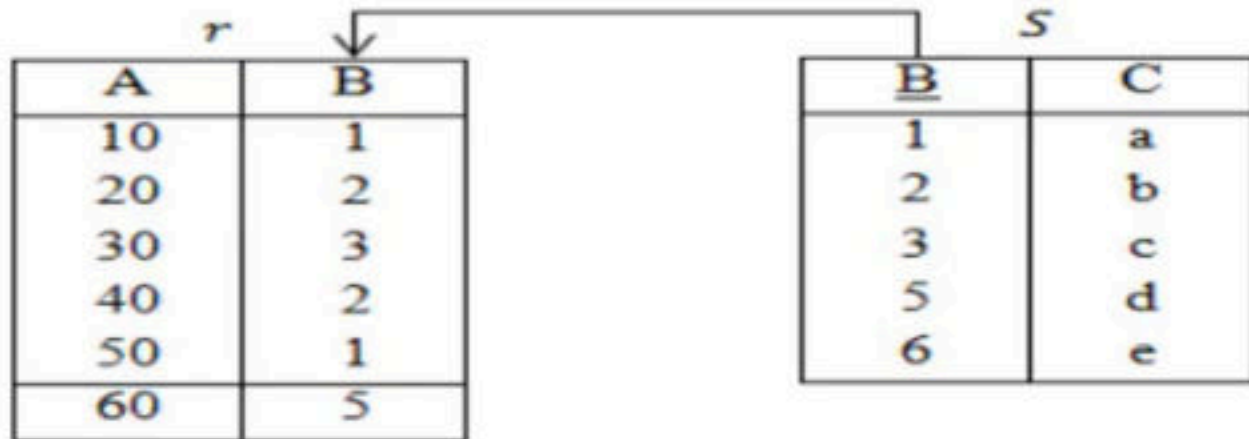
Q: $r \bowtie (\sigma_{B < 5}(s))$

Let LOJ denote the natural left outer-join operation. Assume that r and s contain no null values. Which one of the following is NOT equivalent to Q?

- A $\sigma_{B < 5}(r \bowtie s)$
- B $\sigma_{B < 5}(r \text{ LOJ } s)$
- C $r \text{ LOJ } (\sigma_{B < 5}(s))$**
- D $\sigma_{B < 5}(r) \text{ LOJ } s$

Consider the following relations $r(A, B)$ and $S(B, C)$, where $S.B$ is a primary key and $r.B$ is a foreign key referencing $S.B$

Consider the following tables without NULL values.



Q: $r \bowtie (\sigma_{B < 5}(S))$

The result of $\sigma_{B < 5}(S)$ is

B	C
1	a
2	b
3	c

$r \bowtie (\sigma_{B < 5}(S))$

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a

Option (A):

The result of $r \bowtie S$ is

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a
60	5	d

Option (B):

The result of $r \text{ LOJ } S$ is

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a
NULL	6	e
NULL	5	d

The result of $\sigma_{B < 5}(r \bowtie S)$ is

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a

The result of $\sigma_{B < 5}(r \text{ LOJ } S)$ is

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a

Option (C):
The result of $\sigma_{B < 5}(S)$ is

B	C
1	a
2	b
3	c

Now, the result of $r \text{ LOJ}(\sigma_{B < 5}(S))$

A	B
10	1
20	2
30	3
40	2
50	1
60	5

B	C
1	a
2	b
3	c

=

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a
60	NULL	NULL

Option (D):
The result of $\sigma_{B < 5}(r)$ is

A	B
10	1
20	2
30	3
40	2
50	1

Now, the result of $\sigma_{B < 5}(r) \text{ LOJ} S$ is

A	B	C
10	1	a
20	2	b
30	3	c
40	2	b
50	1	a

Therefore, from the output of above four options, the results of options (A), (B) and (D) are equivalent to Q.

Q 1.

Consider the following relations P(X,Y,Z), Q(X,Y,T) and R(Y,V).

P		
X	Y	Z
X1	Y1	Z1
X1	Y1	Z2
X2	Y2	Z2
X2	Y4	Z4

Q		
X	Y	T
X2	Y1	2
X1	Y2	5
X1	Y1	6
X3	Y3	1

R	
Y	V
Y1	V1
Y3	V2
Y2	V3
Y2	V2

How many tuples will be returned by the following relational algebra query?

$$\Pi_X(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)) - \Pi_X(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

Options:

- A 0
- B 1
- C 2
- D 3

Q 1.

Consider the following relations P(X,Y,Z), Q(X,Y,T) and R(Y,V).

P		
X	Y	Z
X1	Y1	Z1
X1	Y1	Z2
X2	Y2	Z2
X2	Y4	Z4

Q		
X	Y	T
X2	Y1	2
X1	Y2	5
X1	Y1	6
X3	Y3	1

R	
Y	V
Y1	V1
Y3	V2
Y2	V3
Y2	V2

How many tuples will be returned by the following relational algebra query?

$$\Pi_X(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)) - \Pi_X(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

Options:

- A 0
- B 1
- C 2
- D 3

$$\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)$$

X	Y	Z	V
X2	Y2	Z2	V2

$$\Pi_x(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R))$$

X
X2

$$\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R)$$

X	Y	T	V
X1	Y1	6	V1
X1	Y2	5	V3
X1	Y2	5	V2

$$\Pi_x(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

X
X1

$$\Pi_x(\sigma_{(P.Y=R.Y \wedge R.V=V2)}(P \times R)) - \Pi_x(\sigma_{(Q.Y=R.Y \wedge Q.T>2)}(Q \times R))$$

$$= \begin{array}{|c|} \hline X \\ \hline X2 \\ \hline \end{array} - \begin{array}{|c|} \hline X \\ \hline X1 \\ \hline \end{array} = \begin{array}{|c|} \hline X \\ \hline X2 \\ \hline \end{array}$$

Only One Tuple will be returned

SQL

Structured Query Language or **SQL** is a standard Database language which is used to create, maintain and retrieve the data from relational databases like MySQL, Oracle, SQL Server, PostGre, etc.

As the name suggests, it is used when we have structured data (in the form of tables). All databases that are not relational (or do not use fixed structure tables to store data) and therefore do not use SQL, are called NoSQL databases. Examples of NoSQL are MongoDB, DynamoDB, Cassandra, etc

What is Relational Database?

Relational database means the data is stored as well as retrieved in the form of relations (tables). Table 1 shows the relational database with only one relation called **STUDENT** which stores **ROLL_NO**, **NAME**, **ADDRESS**, **PHONE** and **AGE** of students.

STUDENT

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	RAM	DELHI	9455123451	18
2	RAMESH	GURGAON	9652431543	18
3	SUJIT	ROHTAK	9156253131	20
4	SURESH	DELHI	9156768971	18

TABLE 1

These are some important terminologies that are used in terms of relation.

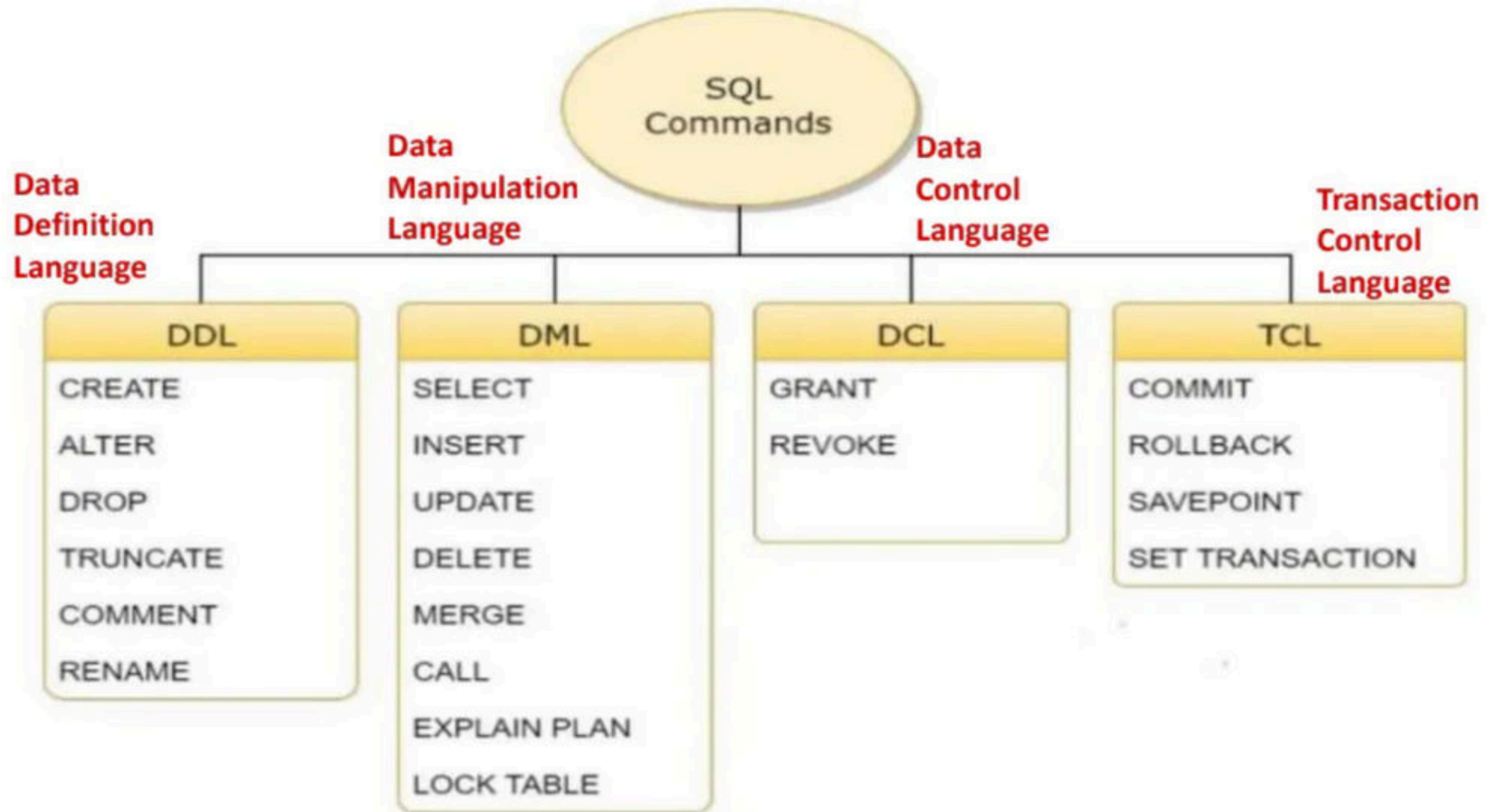
Attribute: Attributes are the properties that define a relation. e.g.; **ROLL_NO**, **NAME** etc.

Tuple: Each row in the relation is known as tuple. The above relation contains 4 tuples, one of which is shown as:

Degree: The number of attributes in the relation is known as degree of the relation. The **STUDENT** relation defined above has degree 5.

Cardinality: The number of tuples in a relation is known as cardinality. The **STUDENT** relation defined above has cardinality 4.

Column: Column represents the set of values for a particular attribute. The column **ROLL_NO** is extracted from relation **STUDENT**.



Data Definition Language: It is used to define the structure of the database. e.g; CREATE TABLE, ADD COLUMN, DROP COLUMN and so on.

Data Manipulation Language: It is used to manipulate data in the relations. e.g.; INSERT, DELETE, UPDATE and so on.

Data Query Language: It is used to extract the data from the relations. e.g.; SELECT
So first we will consider the Data Query Language. A generic query to retrieve from a relational database is:

1. **SELECT [DISTINCT] Attribute_List FROM R1,R2....RM**
2. **[WHERE condition]**
3. **[GROUP BY (Attributes)[HAVING condition]]**
4. **[ORDER BY(Attributes)[DESC]];**

Part of the query represented by statement 1 is compulsory if you want to retrieve from a relational database. The statements written inside [] are optional. We will look at the possible query combination on relation shown in Table 1.

Difference Between TRUNCATE, DELETE, And DROP In SQL Server

TRUNCATE

TRUNCATE SQL query removes all rows from a table, without logging the individual row deletions. TRUNCATE is faster than the DELETE query.

The following example removes all data from the Customers table.

```
TRUNCATE TABLE Customers;
```


1. TRUNCATE is a DDL command
2. TRUNCATE is executed using a table lock and the whole table is locked to remove all records.
3. We cannot use the WHERE clause with TRUNCATE.
4. TRUNCATE removes all rows from a table.
5. Minimal logging in the transaction log, so it is faster performance-wise.
6. TRUNCATE TABLE removes the data by deallocating the data pages used to store the table data and records only the page deallocations in the transaction log.
7. Identify the column is reset to its seed value if the table contains an identity column.
8. To use Truncate on a table you need at least ALTER permission on the table.
9. Truncate uses less transaction space than the Delete statement.
10. Truncate cannot be used with indexed views.
11. TRUNCATE is faster than DELETE.

DELETE

SQL DELETE query deletes all records from a database table. To execute a DELETE query, delete permissions are required on the target table. If you need to use a WHERE clause in a DELETE, select permissions are required as well.

The following query deletes all rows from the Customers table.

```
DELETE FROM Customers;
```

The following SQL query deletes all rows from the Customers table where OrderID is greater than 1000.

```
DELETE FROM Customers WHERE OrderId > 1000;
```

- 1.DELETE is a DML command.
- 2.DELETE is executed using a row lock, each row in the table is locked for deletion.
- 3.We can use where clause with DELETE to filter & delete specific records.
- 4.The DELETE command is used to remove rows from a table based on WHERE condition.
- 5.It maintains the log, so it slower than TRUNCATE.
- 6.The DELETE statement removes rows one at a time and records an entry in the transaction log for each deleted row.
- 7.Identity of column keep DELETE retains the identity.
- 8.To use Delete you need DELETE permission on the table.
- 9.Delete uses more transaction space than the Truncate statement.
- 10.The delete can be used with indexed views.

DROP

DROP table query removes one or more table definitions and all data, indexes, triggers, constraints, and permission specifications for those tables. DROP command requires to ALTER permission on the schema to which the table belongs, CONTROL permission on the table, or membership in the db_ddl admin fixed database role.

The following SQL query drops the Customers table and its data and indexes from the current database.

```
DROP TABLE Customers ;
```

- 1.The DROP command removes a table from the database.
- 2.All the tables' rows, indexes, and privileges will also be removed.
- 3.No DML triggers will be fired.
- 4.The operation cannot be rolled back.
- 5.DROP and TRUNCATE are DDL commands, whereas DELETE is a DML command.
- 6.DELETE operations can be rolled back (undone), while DROP and TRUNCATE operations cannot be rolled back

What is a NULL Value?

A field with a NULL value is a field with no value.

If a field in a table is optional, it is possible to insert a new record or update a record without adding a value to this field. Then, the field will be saved with a NULL value.

Note: A NULL value is different from a zero value or a field that contains spaces. A field with a NULL value is one that has been left blank during record creation!

IS NULL Syntax

```
SELECT column_names
FROM table_name
WHERE column_name IS NULL;
```

IS NOT NULL Syntax

```
SELECT column_names
FROM table_name
WHERE column_name IS NOT NULL;
```

The IS NULL operator is used to test for empty values (NULL values).

The IS NOT NULL operator is used to test for non-empty values (NOT NULL values).

Case 1: If we want to retrieve attributes **ROLL_NO** and **NAME** of all students, the query will be:

```
SELECT ROLL_NO, NAME FROM STUDENT;
```

ROLL_NO	NAME
1	RAM
2	RAMESH
3	SUJIT
4	SURESH

Case 2: If we want to retrieve **ROLL_NO** and **NAME** of the students whose **ROLL_NO** is greater than 2, the query will be:

```
SELECT ROLL_NO, NAME FROM STUDENT  
WHERE ROLL_NO>2;
```

ROLL_NO	NAME
3	SUJIT
4	SURESH

CASE 3: If we want to retrieve all attributes of students, we can write * in place of writing all attributes as:

```
SELECT * FROM STUDENT  
WHERE ROLL_NO>2;
```

ROLL_NO	NAME	ADDRESS	PHONE	AGE
3	SUJIT	ROHTAK	9156253131	20
4	SURESH	DELHI	9156768971	18

CASE 4: If we want to represent the relation in ascending order by **AGE**, we can use ORDER BY clause as:

```
SELECT * FROM STUDENT ORDER BY AGE;
```

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	RAM	DELHI	9455123451	18
2	RAMESH	GURGAON	9652431543	18
4	SURESH	DELHI	9156768971	18
3	SUJIT	ROHTAK	9156253131	20

Note: ORDER BY AGE is equivalent to ORDER BY AGE ASC. If we want to retrieve the results in descending order of AGE, we can use ORDER BY AGE DESC.

CASE 5: If we want to retrieve distinct values of an attribute or group of attribute, DISTINCT is used as in:

```
SELECT DISTINCT ADDRESS FROM STUDENT;
```

ADDRESS

DELHI

GURGAON

ROHTAK

If DISTINCT is not used, DELHI will be repeated twice in result set.

Aggregate Functions

An aggregate function allows you to perform a calculation on a set of values to return a single scalar value. We often use aggregate functions with the **GROUP BY** and **HAVING** clauses of the **SELECT** statement.

The following are the most commonly used SQL aggregate functions:

- **AVG** – calculates the average of a set of **values**.
- **COUNT** – counts **rows** in a specified table or view.
- **MIN** – gets the minimum value in a set of **values**.
- **MAX** – gets the maximum value in a set of **values**.
- **SUM** – calculates the **sum of values**.

COUNT: Count function is used to count the number of rows in a relation.

e.g;

```
SELECT COUNT (PHONE) FROM STUDENT;
```

```
COUNT(PHONE)
```

```
4
```

SUM: SUM function is used to add the values of an attribute in a relation.

e.g;

```
SELECT SUM (AGE) FROM STUDENT;
```

```
SUM(AGE)
```

```
74
```

GROUP BY: Group by is used to group the tuples of a relation based on an attribute or group of attribute. It is always combined with aggregation function which is computed on group. e.g.;

```
SELECT ADDRESS, SUM(AGE) FROM STUDENT  
GROUP BY (ADDRESS);
```

In this query, SUM(AGE) will be computed but not for entire table but for each address. i.e.; sum of AGE for address DELHI(18+18=36) and similarly for other address as well. The output is:

ADDRESS	SUM(AGE)
DELHI	36
GURGAON	18
ROHTAK	20

NOTE: An attribute which is not a part of GROUP BY clause can't be used for selection. Any attribute which is part of GROUP BY CLAUSE can be used for selection but it is not mandatory. But we could use attributes which are not a part of the GROUP BY clause in an aggregate function.

**SELECT ROLL_NO, ADDRESS, SUM(AGE) FROM STUDENT
GROUP BY (ADDRESS);**

STUDENT

ROLL_NO	NAME	ADDRESS	PHONE	AGE
1	RAM	DELHI	9455123451	18
2	RAMESH	GURGAON	9652431543	18
3	SUJIT	ROHTAK	9156253131	20
4	SURESH	DELHI	9156768971	18

If we try to execute the above query, it will result in error because although we have computed SUM(AGE) for each address, there are more than 1 ROLL_NO for each address we have grouped. So it can't be displayed in result set. We need to use aggregate functions on columns after SELECT statement to make sense of the resulting set whenever we are using GROUP BY.

SQL | EXISTS

The EXISTS condition in SQL is used to check whether the result of a correlated nested query is empty (contains no tuples) or not. The result of EXISTS is a boolean value True or False. It can be used in a SELECT, UPDATE, INSERT or DELETE statement.

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
  (SELECT column_name(s)
   FROM table_name
   WHERE condition);
```


Examples:

Consider the following two relation "Customers" and "Orders".

Customers

customer_id	lname	fname	website
401	Singh	Dolly	abc.com
402	Chauhan	Anuj	def.com
403	Kumar	Niteesh	ghi.com
404	Gupta	Shubham	jkl.com
405	Walecha	Divya	abc.com
406	Jain	Sandeep	jkl.com
407	Mehta	Rajiv	abc.com
408	Mehra	Anand	abc.com

Orders

order_id	c_id	order_date
1	407	2017-03-03
2	405	2017-03-05
3	408	2017-01-18
4	404	2017-02-05

1. Using EXISTS condition with SELECT statement

To fetch the first and last name of the customers who placed atleast one order.

```
SELECT fname, lname
FROM Customers
WHERE EXISTS (SELECT *
              FROM Orders
              WHERE Customers.customer_id = Orders.c_id);
```

Output:

fname	lname
Shubham	Gupta
Divya	Walecha
Rajiv	Mehta
Anand	Mehra

The SQL IN Operator

The IN operator allows you to specify multiple values in a WHERE clause. The IN operator is a shorthand for multiple OR conditions.

IN Syntax

```
SELECT  column_name(s)
FROM    table_name
WHERE   column_name IN (value1, value2, ...);
```

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```

The SQL ANY and ALL Operators

The ANY and ALL operators are used with a WHERE or HAVING clause.

The ANY operator returns true if **any of the subquery** values meet the condition.

The ALL operator returns true if **all of the subquery** values meet the condition.

```
SELECT ProductName
FROM Products
WHERE ProductID
= ANY (SELECT ProductID FROM OrderDetails WHERE
Quantity > 99);
```

```
SELECT ProductName
FROM Products
WHERE ProductID
= ANY (SELECT ProductID FROM OrderDetails WHERE
Quantity = 10);
```

SQL | EXISTS

The EXISTS condition in SQL is used to check whether the result of a correlated nested query is empty (contains no tuples) or not. The result of EXISTS is a boolean value True or False. It can be used in a SELECT, UPDATE, INSERT or DELETE statement.

```
SELECT column_name(s)
FROM table_name
WHERE EXISTS
  (SELECT column_name(s)
   FROM table_name
   WHERE condition);
```

Examples:

Consider the following two relation "Customers" and "Orders".

Customers

customer_id	lname	fname	website
401	Singh	Dolly	abc.com
402	Chauhan	Anuj	def.com
403	Kumar	Niteesh	ghi.com
404	Gupta	Shubham	jkl.com
405	Walecha	Divya	abc.com
406	Jain	Sandeep	jkl.com
407	Mehta	Rajiv	abc.com
408	Mehra	Anand	abc.com

Orders

order_id	c_id	order_date
1	407	2017-03-03
2	405	2017-03-05
3	408	2017-01-18
4	404	2017-02-05

1. Using EXISTS condition with SELECT statement

To fetch the first and last name of the customers who placed atleast one order.

```
SELECT fname, lname
FROM Customers
WHERE EXISTS (SELECT *
              FROM Orders
              WHERE Customers.customer_id = Orders.c_id);
```

Output:

fname	lname
Shubham	Gupta
Divya	Walecha
Rajiv	Mehta
Anand	Mehra

The SQL IN Operator

The IN operator allows you to specify multiple values in a WHERE clause. The IN operator is a shorthand for multiple OR conditions.

IN Syntax

```
SELECT  column_name(s)
FROM    table_name
WHERE   column_name IN (value1, value2, ...);
```

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```


The SQL BETWEEN Operator

The BETWEEN operator selects values within a given range. The values can be numbers, text, or dates.

The BETWEEN operator is inclusive: begin and end values are included.

```
SELECT column_name(s)
FROM table_name
WHERE column_name BETWEEN value1 AND value2;
```

ProductID	ProductName	SupplierID	CategoryID	Unit	Price
1	Chais	1	1	10 boxes x 20 bags	18
2	Chang	1	1	24 - 12 oz bottles	19
3	Aniseed Syrup	1	2	12 - 550 ml bottles	10
4	Chef Anton's Cajun Seasoning	1	2	48 - 6 oz jars	22
5	Chef Anton's Gumbo Mix	1	2	36 boxes	21.35

```
SELECT * FROM Products  
WHERE Price BETWEEN 10 AND 20;
```

S.No	Operator	Description	Sample Query
1.	LIKE	Used for matching or checking.	Select * from student where <u>subject_stream</u> LIKE 'Science';
2.	AND	Used for combining two conditions together.	Select * from student where <u>subject_stream</u> LIKE 'Science' AND marks > 100;
3.	IN	Used to compare values.	Select * from student where <u>subject_stream</u> IN ('Science', 'Commerce', 'Arts');
4.	BETWEEN	Used to compare value within a range.	Select * from student where marks BETWEEN 100 AND 200;
5.	OR	Used to check either of the two comparison.	Select * from student where marks = "100" OR subject LIKE 'DBMS';
6.	NOT	Used to fetch value other than values which are not required.	Select * from student where subject NOT LIKE 'DBMS';

The SQL ANY and ALL Operators

The ANY and ALL operators are used with a WHERE or HAVING clause.

The ANY operator returns true if **any of the subquery** values meet the condition.

The ALL operator returns true if **all of the subquery** values meet the condition.

```
SELECT ProductName
FROM Products
WHERE ProductID
= ANY (SELECT ProductID FROM OrderDetails WHERE
Quantity > 99);
```

```
SELECT ProductName
FROM Products
WHERE ProductID
= ANY (SELECT ProductID FROM OrderDetails WHERE
Quantity = 10);
```

SQL Logical Operators:

In SQL, we use Logical operators such as AND, OR, NOT, IN, BETWEEN, ALL, EXISTS, LIKE. Check the below image for the description of each operator.

Operator	Description
ALL	TRUE if all of the subquery values meet the condition
AND	TRUE if all the conditions separated by AND is TRUE
ANY	TRUE if any of the subquery values meet the condition
BETWEEN	TRUE if the operand is within the range of comparisons
IN	TRUE if the operand is equal to one of a list of expressions
NOT	Displays a record if the condition(s) is NOT TRUE
OR	TRUE if any of the conditions separated by OR is TRUE
EXISTS	TRUE if the subquery returns one or more records
LIKE	TRUE if the operand matches a pattern

Commands	Description	Syntax Format
SELECT	<p>Retrieving and display data, value or even a database object.</p> <p>This can be followed by a SQL FROM clause that specifies the table name or the source of the data.</p>	<p>SELECT [* field(s)] FROM nameTable WHERE conditions(s);</p>
INSERT	<p>Entering new data into the database table that has previously been created.</p>	<p>INSERT INTO nameTable VALUES (value(s));</p>
UPDATE	<p>Updating data that exists in the database table.</p>	<p>UPDATE nameTable SET field(s)=value(s) WHERE condition(s);</p>
DELETE	<p>Deleting data that exists in the database table</p>	<p>DELETE FROM nameTable WHERE condition(s);</p>

QUERYING DATA FROM A TABLE

SELECT c1, c2 FROM t;

Query data in columns c1, c2 from a table

SELECT * FROM t;

Query all rows and columns from a table

SELECT c1, c2 FROM t

WHERE condition;

Query data and filter rows with a condition

SELECT DISTINCT c1 FROM t

WHERE condition;

Query distinct rows from a table

SELECT c1, c2 FROM t

ORDER BY c1 ASC [DESC];

Sort the result set in ascending or descending order

SELECT c1, c2 FROM t

ORDER BY c1

LIMIT n OFFSET offset;

Skip *offset* of rows and return the next *n* rows

SELECT c1, aggregate(c2)

FROM t

GROUP BY c1;

Group rows using an aggregate function

SELECT c1, aggregate(c2)

FROM t

GROUP BY c1

HAVING condition;

Filter groups using HAVING clause

QUERYING FROM MULTIPLE TABLES

SELECT c1, c2

FROM t1

INNER JOIN t2 ON condition;

Inner join t1 and t2

SELECT c1, c2

FROM t1

LEFT JOIN t2 ON condition;

Left join t1 and t2

SELECT c1, c2

FROM t1

RIGHT JOIN t2 ON condition;

Right join t1 and t2

SELECT c1, c2

FROM t1

FULL OUTER JOIN t2 ON condition;

Perform full outer join

SELECT c1, c2

FROM t1

CROSS JOIN t2;

Produce a Cartesian product of rows in tables

SELECT c1, c2

FROM t1, t2

Another way to perform cross join

SELECT c1, c2

FROM t1 A

INNER JOIN t2 B ON condition;

Join t1 to itself using INNER JOIN clause

USING SQL OPERATORS

SELECT c1, c2 FROM t1

UNION [ALL]

SELECT c1, c2 FROM t2;

Combine rows from two queries

SELECT c1, c2 FROM t1

INTERSECT

SELECT c1, c2 FROM t2;

Return the intersection of two queries

SELECT c1, c2 FROM t1

MINUS

SELECT c1, c2 FROM t2;

Subtract a result set from another result set

SELECT c1, c2 FROM t1

WHERE c1 [NOT] LIKE pattern;

Query rows using pattern matching %, _

SELECT c1, c2 FROM t

WHERE c1 [NOT] IN value_list;

Query rows in a list

SELECT c1, c2 FROM t

WHERE c1 BETWEEN low AND high;

Query rows between two values

SELECT c1, c2 FROM t

WHERE c1 IS [NOT] NULL;

Check if values in a table is NULL or not

MANAGING TABLES

```
CREATE TABLE t (  
  id INT PRIMARY KEY,  
  name VARCHAR NOT NULL,  
  price INT DEFAULT 0  
);  
Create a new table with three columns
```

```
DROP TABLE t;  
Delete the table from the database
```

```
ALTER TABLE t ADD column;  
Add a new column to the table
```

```
ALTER TABLE t DROP COLUMN c;  
Drop column c from the table
```

```
ALTER TABLE t ADD constraint;  
Add a constraint
```

```
ALTER TABLE t DROP constraint;  
Drop a constraint
```

```
ALTER TABLE t1 RENAME TO t2;  
Rename a table from t1 to t2
```

```
ALTER TABLE t1 RENAME c1 TO c2;  
Rename column c1 to c2
```

```
TRUNCATE TABLE t;  
Remove all data in a table
```

USING SQL CONSTRAINTS

```
CREATE TABLE t(  
  c1 INT, c2 INT, c3 VARCHAR,  
  PRIMARY KEY (c1,c2)  
);  
Set c1 and c2 as a primary key
```

```
CREATE TABLE t1(  
  c1 INT PRIMARY KEY,  
  c2 INT,  
  FOREIGN KEY (c2) REFERENCES t2(c2)  
);  
Set c2 column as a foreign key
```

```
CREATE TABLE t(  
  c1 INT, c1 INT,  
  UNIQUE(c2,c3)  
);  
Make the values in c1 and c2 unique
```

```
CREATE TABLE t(  
  c1 INT, c2 INT,  
  CHECK(c1 > 0 AND c1 >= c2)  
);  
Ensure c1 > 0 and values in c1 >= c2
```

```
CREATE TABLE t(  
  c1 INT PRIMARY KEY,  
  c2 VARCHAR NOT NULL  
);  
Set values in c2 column not NULL
```

MODIFYING DATA

```
INSERT INTO t(column_list)  
VALUES(value_list);  
Insert one row into a table
```

```
INSERT INTO t(column_list)  
VALUES (value_list),  
      (value_list), ...;  
Insert multiple rows into a table
```

```
INSERT INTO t1(column_list)  
SELECT column_list  
FROM t2;  
Insert rows from t2 into t1
```

```
UPDATE t  
SET c1 = new_value;  
Update new value in the column c1 for all rows
```

```
UPDATE t  
SET c1 = new_value,  
    c2 = new_value  
WHERE condition;  
Update values in the column c1, c2 that match  
the condition
```

```
DELETE FROM t;  
Delete all data in a table
```

```
DELETE FROM t  
WHERE condition;  
Delete subset of rows in a table
```


MANAGING VIEWS

```
CREATE VIEW v(c1,c2)
AS
SELECT c1, c2
FROM t;
```

Create a new view that consists of c1 and c2

```
CREATE VIEW v(c1,c2)
AS
SELECT c1, c2
FROM t;
```

```
WITH [CASCADED | LOCAL] CHECK OPTION;
```

Create a new view with check option

```
CREATE RECURSIVE VIEW v
AS
```

select-statement *-- anchor part*

```
UNION [ALL]
```

select-statement; *-- recursive part*

Create a recursive view

```
CREATE TEMPORARY VIEW v
AS
```

```
SELECT c1, c2
FROM t;
```

Create a temporary view

```
DROP VIEW view_name;
```

Delete a view

MANAGING INDEXES

```
CREATE INDEX idx_name
```

```
ON t(c1,c2);
```

Create an index on c1 and c2 of the table t

```
CREATE UNIQUE INDEX idx_name
```

```
ON t(c3,c4);
```

Create a unique index on c3, c4 of the table t

```
DROP INDEX idx_name;
```

Drop an index

SQL AGGREGATE FUNCTIONS

AVG returns the average of a list

COUNT returns the number of elements of a list

SUM returns the total of a list

MAX returns the maximum value in a list

MIN returns the minimum value in a list

MANAGING TRIGGERS

```
CREATE OR MODIFY TRIGGER trigger_name
```

```
WHEN EVENT
```

```
ON table_name TRIGGER_TYPE
```

```
EXECUTE stored_procedure;
```

Create or modify a trigger

WHEN

- **BEFORE** – Invoke before the event occurs
- **AFTER** – Invoke after the event occurs

EVENT

- **INSERT** – invoke for INSERT
- **UPDATE** – invoke for UPDATE
- **DELETE** – invoke for DELETE

TRIGGER_TYPE

- **FOR EACH ROW**
- **FOR EACH STATEMENT**

```
CREATE TRIGGER before_insert_person
BEFORE INSERT
```

```
ON person FOR EACH ROW
```

```
EXECUTE stored_procedure;
```

Create a trigger invoked before a new row is inserted into the person table

```
DROP TRIGGER trigger_name;
```

Delete a specific trigger

SQL ALTER TABLE Statement

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table. The ALTER TABLE statement is also used to add and drop various constraints on an existing table.

ALTER TABLE - ADD Column

To add a column in a table, use the following syntax:

```
ALTER TABLE table_name  
ADD column_name datatype;
```

ALTER TABLE - DROP COLUMN

To delete a column in a table, use the following syntax (notice that some database systems don't allow deleting a column):

```
ALTER TABLE table_name  
DROP COLUMN column_name;
```

ALTER TABLE - ALTER/MODIFY COLUMN

To change the data type of a column in a table, use the following syntax:

```
ALTER TABLE table_name  
MODIFY COLUMN column_name datatype;
```

The SQL UPDATE Statement

The UPDATE statement is used to modify the existing records in a table.

UPDATE Syntax

```
UPDATE table_name
```

```
SET column1 = value1, column2 = value2, ...
```

```
WHERE condition;
```

Note: Be careful when updating records in a table! Notice the WHERE clause in the UPDATE statement. The WHERE clause specifies which record(s) that should be updated. If you omit the WHERE clause, all records in the table will be updated!

Q.94 Consider the following tables (relations) :

Students

<u>Roll-No</u>	Name
18CS101	Ramesh
18CS102	Mukesh
18CS103	Ramesh

Performance

<u>Roll-No</u>	<u>Course</u>	Marks
18CS101	DBMS	60
18CS101	Compiler Design	65
18CS102	DBMS	80
18CS103	DBMS	85
18CS102	Compiler Design	75
18CS103	Operating System	70

Primary keys in the tables are shown using underline. Now, consider the following query :

```
SELECT S.Name, Sum (P.Marks)
FROM Students S, Performance P
WHERE S.Roll-No = P.Roll-No
GROUP BY S.Name
```

The number of rows returned by above query is

- Options**
1. 0
 2. 3
 3. 1
 4. 2

```
SELECT S.Name, Sum (P.Marks)
FROM Students S, Performance P
WHERE S.Roll-No = P.Roll-No
```



Roll-No	Name	Course	Marks
18CS101	Ramesh	DBMS	60
18CS101	Ramesh	Compiler Design	65
18CS102	Mukesh	DBMS	80
18CS102	Mukesh	Compiler Design	75
18CS103	Ramesh	DBMS	85
18CS103	Ramesh	Operating System	70

```
SELECT S.Name, Sum (P.Marks)
FROM Students S, Performance P
WHERE S.Roll-No = P.Roll-No
GROUP BY S.Name
```



Name	Marks
Ramesh	280
Mukesh	155

Q.94 Consider the following tables (relations) :

Students

<u>Roll-No</u>	Name
18CS101	Ramesh
18CS102	Mukesh
18CS103	Ramesh

Performance

<u>Roll-No</u>	<u>Course</u>	Marks
18CS101	DBMS	60
18CS101	Compiler Design	65
18CS102	DBMS	80
18CS103	DBMS	85
18CS102	Compiler Design	75
18CS103	Operating System	70

Primary keys in the tables are shown using underline. Now, consider the following query :

```
SELECT S.Name, Sum (P.Marks)
FROM Students S, Performance P
WHERE S.Roll-No = P.Roll-No
GROUP BY S.Name
```

The number of rows returned by above query is

- Options**
1. 0
 2. 3
 3. 1
 4. 2 ✓

NTA NET Dec 2018

A relational database contains two tables Student and Performance as shown below:

Student		Performance		
Roll_no.	Student_name	Roll_no.	Subject_code	Marks
1	Amit	1	A	86
2	Priya	1	B	95
3	Vinit	1	C	90
4	Rohan	2	A	89
5	Smita	2	C	92
		3	C	80

The primary key of the Student table is Roll_no. For the Performance table, the columns Roll_no. and Subject_code together form the primary key. Consider the SQL query given below:

```
SELECT S.Student_name, sum (P.Marks)  
FROM Student S, Performance P  
WHERE P.Marks > 84  
GROUP BY S.Student_name;
```

- A 0**
- B 9**
- C 7**
- D 5**

The number of rows returned by the above SQL query is _____.

A relational database contains two tables Student and Performance as shown below:

Student		Performance		
Roll_no.	Student_name	Roll_no.	Subject_code	Marks
1	Amit	1	A	86
2	Priya	1	B	95
3	Vinit	1	C	90
4	Rohan	2	A	89
5	Smita	2	C	92
		3	C	80

The primary key of the Student table is Roll_no. For the Performance table, the columns Roll_no. and Subject_code together form the primary key. Consider the SQL query given below:

```
SELECT S.Student_name, sum (P.Marks)
FROM Student S, Performance P
WHERE P.Marks > 84
GROUP BY S.Student_name;
```

Student	sum(P.Marks)
-----	-----
Rohan	452
Vinit	452
Priya	452
Amit	452
Smita	452

- A 0
- B 9
- C 7
- D 5

The number of rows returned by the above SQL query is ____.

Consider a relational database containing the following schemas.

Catalogue

<u>sno</u>	<u>pno</u>	cost
S1	P1	150
S1	P2	50
S1	P3	100
S2	P4	200
S2	P5	250
S3	P1	250
S3	P2	150
S3	P5	300
S3	P4	250

Suppliers

<u>sno</u>	sname	location
S1	M/s Royal furniture	Delhi
S2	M/s Balaji furniture	Bangalore
S3	M/s Premium furniture	Chennai

Parts

<u>pno</u>	pname	part_spec
P1	Table	Wood
P2	Chair	Wood
P3	Table	Steel
P4	Almirah	Steel
P5	Almirah	Wood

The primary key of each table is indicated by underlying the constituent fields.

```
SELECT s.sno, s.sname
```

```
FROM Suppliers s, Catalogue c
```

```
WHERE s.sno = c.sno AND
```

```
Cost > (SELECT AVG (cost)
```

```
FROM Catalogue
```

```
WHERE pno = 'P4'
```

```
GROUP BY pno);
```

The number of rows returned by the above SQL query is

A

B

C

D

5

4

Consider a relational database containing the following schemas.

Catalogue

<u>sno</u>	<u>pno</u>	cost
S1	P1	150
S1	P2	50
S1	P3	100
S2	P4	200
S2	P5	250
S3	P1	250
S3	P2	150
S3	P5	300
S3	P4	250

Suppliers

<u>sno</u>	sname	location
S1	M/s Royal furniture	Delhi
S2	M/s Balaji furniture	Bangalore
S3	M/s Premium furniture	Chennai

Parts

<u>pno</u>	pname	part_spec
P1	Table	Wood
P2	Chair	Wood
P3	Table	Steel
P4	Almirah	Steel
P5	Almirah	Wood

The primary key of each table is indicated by underlying the constituent fields.

```
SELECT s.sno, s.sname
```

```
FROM Suppliers s, Catalogue c
```

```
WHERE s.sno = c.sno AND
```

```
Cost > (SELECT AVG (cost)
```

```
FROM Catalogue
```

```
WHERE pno = 'P4'
```

```
GROUP BY pno);
```

The number of rows returned by the above SQL query is

A

B

C

D

5

4

The inner query "select avg(cost) from catalogue where pno='P4' group by pno;" returns:

AVG(COST)

225

The outer query "select s.sno,s.sname from suppliers s, catalogue c where s.sno=c.sno" returns:

SNO SNAME

S1 M/s Royal furniture

S1 M/s Royal furniture

S1 M/s Royal furniture

S2 M/s Balaji furniture

S2 M/s Balaji furniture

S3 M/s Premium furniture

S3 M/s Premium furniture

S3 M/s Premium furniture

S3 M/s Premium furniture

So, the final result of the query is:

SN SNAME

S2 M/s Balaji furniture

S3 M/s Premium furniture

S3 M/s Premium furniture

S3 M/s Premium furniture

Therefore, 4 rows will be returned by the query.

Consider the following two tables and four queries in SQL.

Book (isbn, bname), Stock (isbn, copies)

Query 1: SELECT B.isbn, S.copies
FROM Book B INNER JOIN Stock S
ON B.isbn = S.isbn;

Query 2: SELECT B.isbn, S.copies
FROM Book B LEFT OUTER JOIN Stock S
ON B.isbn = S.isbn;

Query 3: SELECT B.isbn, S.copies
FROM Book B RIGHT OUTER JOIN Stock S
ON B.isbn = S.isbn;

Query 4: SELECT B.isbn, S.copies
FROM Book B FULL OUTER JOIN Stock S
ON B.isbn = S.isbn;

Which one of the queries above is certain to have an output that is a superset of the outputs of the other three queries?

- A Query 1
- B Query 2
- C Query 3
- D Query 4

Consider the following two tables and four queries in SQL.

Book (isbn, bname), Stock (isbn, copies)

Query 1: SELECT B.isbn, S.copies
FROM Book B INNER JOIN Stock S
ON B.isbn = S.isbn;

Query 2: SELECT B.isbn, S.copies
FROM Book B LEFT OUTER JOIN Stock S
ON B.isbn = S.isbn;

Query 3: SELECT B.isbn, S.copies
FROM Book B RIGHT OUTER JOIN Stock S
ON B.isbn = S.isbn;

Query 4: SELECT B.isbn, S.copies
FROM Book B FULL OUTER JOIN Stock S
ON B.isbn = S.isbn;

Which one of the queries above is certain to have an output that is a superset of the outputs of the other three queries?

- A Query 1
- B Query 2
- C Query 3
- D Query 4**

Consider a database that has the relation schema EMP (EmpId, EmpName, and DeptName). An instance of the schema EMP and a SQL query on it are given below.

EMP		
EmpId	EmpName	DeptName
1	XYA	AA
2	XYB	AA
3	XYC	AA
4	XYD	AA
5	XYE	AB
6	XYF	AB
7	XYG	AB
8	XYH	AC
9	XYI	AC
10	XYJ	AC
11	XYK	AD
12	XYL	AD
13	XYM	AE

```
SELECT AVG(EC. Num)
FROM EC
WHERE (DeptName, Num) IN
      (SELECT DeptName, COUNT(EmpId) AS
        EC(DeptName, Num)
       FROM EMP
        GROUP BY DeptName)
```

The output of executing the SQL query is _____.

- A 2.6
- B 2.7
- C 2.8
- D 2.9

Consider a database that has the relation schema EMP (EmpId, EmpName, and DeptName). An instance of the schema EMP and a SQL query on it are given below.

EMP		
EmpId	EmpName	DeptName
1	XYA	AA
2	XYB	AA
3	XYC	AA
4	XYD	AA
5	XYE	AB
6	XYF	AB
7	XYG	AB
8	XYH	AC
9	XYI	AC
10	XYJ	AC
11	XYK	AD
12	XYL	AD
13	XYM	AE

```
SELECT AVG(EC. Num)
FROM EC
WHERE (DeptName, Num) IN
      (SELECT DeptName, COUNT(EmpId) AS
       EC(DeptName, Num)
       FROM EMP
       GROUP BY DeptName)
```

The output of executing the SQL query is _____.

- A 2.6
- B 2.7
- C 2.8
- D 2.9

The given query is

```
SELECT AVG(EC.Num)FROM EC } Outer Query  
WHERE (DeptName, Num)
```

IN

```
(SELECT DeptName, COUNT(EmpId)as )  
EC(DeptName, Name) } Inner Query  
FROM EMP GROUP BY DeptName;
```

EC	
DeptName	Num
AA	4
AB	3
AC	3
AD	2
AE	1

The outer query will find the
 $Avg(Num) = (4+3+3+2+1)/5 = 2.6$

A relational schema for a train reservation database is given below.

Passenger (pid, pname, age)

Reservation (pid, class, tid)

Table: Passenger

pid pname age

0 Sachin 65
1 Rahul 66
2 Sourav 67
3 Anil 69

Table : Reservation

pid class tid

0 AC 8200
1 AC 8201
2 SC 8201
5 AC 8203
1 SC 8204
3 AC 8202

What pids are returned by the following SQL query for the above instance of the tables?

```
SLECT pid  
FROM Reservation ,  
WHERE class 'AC' AND  
EXISTS (SELECT *  
FROM Passenger  
WHERE age > 65 AND  
Passenger. pid = Reservation.pid)
```

- A 1, 0
- B 1, 2
- C 1, 3
- D 1, 5

A relational schema for a train reservation database is given below.

Passenger (pid, pname, age)

Reservation (pid, class, tid)

Table: Passenger

pid pname age

0 Sachin 65
1 Rahul 66
2 Sourav 67
3 Anil 69

Table : Reservation

pid class tid

0 AC 8200
1 AC 8201
2 SC 8201
5 AC 8203
1 SC 8204
3 AC 8202

What pids are returned by the following SQL query for the above instance of the tables?

```
SELECT pid
FROM Reservation ,
WHERE class 'AC' AND
EXISTS (SELECT *
FROM Passenger
WHERE age > 65 AND
Passenger. pid = Reservation.pid)
```

- A 1, 0
- B 1, 2
- C 1, 3**
- D 1, 5

Pid	Pname	Age
0	Sachin	65
1	Rahul	66
2	Sourav	67
3	Anil	69

↓
Age > 65

Pid	Class	tid
0	AC	8200
1	AC	8201
2	SC	8201
5	AC	8203
1	SC	8204
3	AC	8202

↓
Class = 'AC'

Given the following two statements:

S1: Every table with two single-valued attributes is in 1NF, 2NF, 3NF and BCNF.

S2: $AB \rightarrow C$, $D \rightarrow E$, $E \rightarrow C$ is a minimal cover for the set of functional dependencies

$AB \rightarrow C$, $D \rightarrow E$, $AB \rightarrow E$, $E \rightarrow C$.

- A S1 is TRUE and S2 is FALSE.
- B Both S1 and S2 are TRUE.
- C S1 is FALSE and S2 is TRUE.
- D Both S1 and S2 are FALSE.

Given the following two statements:

S1: Every table with two single-valued attributes is in 1NF, 2NF, 3NF and BCNF.

S2: $AB \rightarrow C, D \rightarrow E, E \rightarrow C$ is a minimal cover for the set of functional dependencies

$AB \rightarrow C, D \rightarrow E, AB \rightarrow E, E \rightarrow C$.

- A** **S1 is TRUE and S2 is FALSE.**
- B Both S1 and S2 are TRUE.
- C S1 is FALSE and S2 is TRUE.
- D Both S1 and S2 are FALSE.

S1: True

Let $R(AB)$ be a two attribute relation, then

If $\{A \rightarrow B\}$ exists then BCNF since $\{A\}^+ = AB = R$

If $\{B \rightarrow A\}$ exists then BCNF since $\{B\}^+ = AB = R$

S2: False

The canonical cover for the given FD set is $\{AB \rightarrow C, D \rightarrow E, AB \rightarrow E, E \rightarrow C\}$. As we can see $AB \rightarrow E$ is not covered in minimal cover since $\{AB\}^+ = ABC$ in the given cover $\{AB \rightarrow C, D \rightarrow E, E \rightarrow C\}$

Given the following schema:

employees(emp-id, first-name, last-name, hire-date, dept-id, salary)

departments(dept-id, dept-name, manager-id, location-id)

You want to display the last names and hire dates of all latest

hires in their respective departments in the location ID 1700. You issue the following query:

```
SQL> SELECT last-name, hire-date  
       FROM employees  
       WHERE (dept-id, hire-date) IN ( SELECT dept-id, MAX(hire-date)  
       FROM employees JOIN departments USING(dept-id)  
       WHERE location-id = 1700  
       GROUP BY dept-id);
```

What is the outcome?

- A It executes but does not give the correct result.
- B It executes and gives the correct result
- C It generates an error because of pairwise comparison.
- D It generates an error because the GROUP BY clause cannot be used with table joins in a subquery.

What is the outcome?

A It executes but does not give the correct result.

B It executes and gives the correct result

C It generates an error because of pairwise comparison.

D It generates an error because the GROUP BY clause cannot be used with table joins in a subquery.

The maximum number of superkeys for the relation schema $R(E, F, G, H)$ with E as the key is _____.

8

9

12

10

The maximum number of superkeys for the relation schema R(E, F, G, H) with E as the key is _____.

8

9

12

10

Maximum no. of possible super keys for a relation with n attributes with one candidate key (only one attribute) = 2^{n-1}

Here, n = 4.

So, the possible super keys = $2^{4-1} = 8$

The super keys are: E, EH, EG, EF, EGH, EFH, EFG, EFGH.

SQL allows tuples in relations, and correspondingly defines the multiplicity of tuples in the result of joins. Which one of the following queries always gives the same answer as the nested query shown below:

```
select * from R where a in (select S.a from S)
```

- A select R.* from R,S where R.a=S.a
- B select distinct R.* from R,S where R.a=S.a
- C select R.* from R,(select distinct a from S) as S1 where R.a=S1.a
- D select R.* from R,S where R.a=S.a and is unique R

Multiplicity = Cardinality + Participation

Cardinality: Denotes the maximum number of possible relationship occurrences in which a certain entity can participate in (in simple terms: at most).

Participation: Denotes if all or only some entity occurrences participate in a relationship (in simple terms: at least).

The solution of this question lies in the data set(tuples) of Relations R and S we define. If we miss some case then we may get wrong answer.

Let's say,

Relation R(BCA) with attributes B, C and A contains the following tuples.

BCA	AMN
-----	-----
721	167
721	284
895	596
895	553

Now ,the original Query will give result as:

“select * from R where a in (select S.a from S) ” – The query asks to display every tuple of Relation R where R.a is present in the complete set S.a.

BCA

721
721
895
895

Option A query will result in :
"select R.* from R, S where R.a=S.a"

B	C	A
7	2	1
7	2	1
8	9	5
8	9	5
8	9	5
8	9	5

Option D query will result in : NULL
set
"select R.* from R,S where R.a=S.a
and is unique R"

Option B query will result in :
" select distinct R.* from R,S where R.a=S.a"

B	C	A
7	2	1
8	9	5

Option C query will result in :
"select R.* from R,(select distinct a from S) as S1
where
R.a=S1.a"

B	C	A
7	2	1
7	2	1
8	9	5
8	9	5

SQL allows tuples in relations, and correspondingly defines the multiplicity of tuples in the result of joins. Which one of the following queries always gives the same answer as the nested query shown below:

`select * from R where a in (select S.a from S)`

- A `select R.* from R,S where R.a=S.a`
- B `select distinct R.* from R,S where R.a=S.a`
- C `select R.* from R,(select distinct a from S) as S1 where R.a=S1.a`**
- D `select R.* from R,S where R.a=S.a and is unique R`

Multiplicity of duplicate tuples will be distributed when there is a match between R.a and S.a; and for that match, S.a's value is repeated in each cases except the third case. So, the output of query given in the question matches with the output of (C).

What is the optimized version of the relation algebra expression

$$\pi_{A1}(\pi_{A2}(\sigma_{F1}(\sigma_{F2}(r))))$$

where $A1, A2$ are sets of attributes in r with $A1 \subset A2$ and $F1, F2$ are Boolean expressions

based on the attributes in r ?

$$\pi_{A1}(\sigma_{(F1 \wedge F2)}(r))$$
$$\pi_{A1}(\sigma_{(F1 \vee F2)}(r))$$
$$\pi_{A2}(\sigma_{(F1 \wedge F2)}(r))$$
$$\pi_{A2}(\sigma_{(F1 \vee F2)}(r))$$

What is the optimized version of the relation algebra expression

$\pi_{A1}(\pi_{A2}(\sigma_{F1}(\sigma_{F2}(r))))$

where $A1, A2$ are sets of attributes in r with $A1 \subset A2$ and $F1, F2$ are Boolean expressions

based on the attributes in r ?

$\pi_{A1}(\sigma_{(F1 \wedge F2)}(r))$

$\pi_{A1}(\sigma_{(F1 \vee F2)}(r))$

$\pi_{A2}(\sigma_{(F1 \wedge F2)}(r))$

$\pi_{A2}(\sigma_{(F1 \vee F2)}(r))$

Consider the following relational schema:

employee(empId,empName,empDept)

customer(custId,custName,salesRepId,rating)

salesRepId is a foreign key referring to empId of the employee relation. Assume that each employee makes a sale to at least one customer. What does the following query return?

```
SELECT empName FROM employee E
WHERE NOT EXISTS (SELECT custId FROM customer C
                  WHERE C.salesRepId = E.empId
                  AND(GATEC.rating <> 'GOOD');
```

- A Names of all the employees with at least one of their customers having a 'GOOD' rating.
- B Names of all the employees with at most one of their customers having a 'GOOD' rating.
- C Names of all the employees with none of their customers having a 'GOOD' rating.
- D Names of all the employees with all their customers having a 'GOOD' rating.

Consider the following relational schema:

employee(empId,empName,empDept)

customer(custId,custName,salesRepId,rating)

salesRepId is a foreign key referring to empId of the employee relation. Assume that each employee makes a sale to at least one customer. What does the following query return?

```
SELECT empName FROM employee E
WHERE NOT EXISTS (SELECT custId FROM customer C
                  WHERE C.salesRepId = E.empId
                  AND(GATEC.rating <> 'GOOD');
```

- A Names of all the employees with at least one of their customers having a 'GOOD' rating.
- B Names of all the employees with at most one of their customers having a 'GOOD' rating.
- C Names of all the employees with none of their customers having a 'GOOD' rating.
- D Names of all the employees with all their customers having a 'GOOD' rating.

Suppose $R_1(A, B)$ and $R_2(C, D)$ are two relation schemas. Let r_1 and r_2 be the corresponding relation instances. B is a foreign key that refers to C in R_2 . If data in r_1 and r_2 satisfy referential integrity constraints, which of the following is ALWAYS TRUE?

$$\Pi_B(r_1) - \Pi_C(r_2) = \emptyset$$

$$\Pi_C(r_2) - \Pi_B(r_1) = \emptyset$$

$$\Pi_B(r_1) = \Pi_C(r_2)$$

$$\Pi_B(r_1) - \Pi_C(r_2) \neq \emptyset$$

Suppose $R_1(A, B)$ and $R_2(C, D)$ are two relation schemas. Let r_1 and r_2 be the corresponding relation instances. B is a foreign key that refers to C in R_2 . If data in r_1 and r_2 satisfy referential integrity constraints, which of the following is ALWAYS TRUE?

$$\prod B (r_1) - \prod C (r_2) = \emptyset$$

$$\prod C (r_2) - \prod B (r_1) = \emptyset$$

$$\prod B (r_1) = \prod C (r_2)$$

$$\prod B (r_1) - \prod C (r_2) \neq \emptyset$$

Referential integrity means, all the values in foreign key should be present in primary key.
So we can say that $r_2(C)$ is the superset of $r_1(B)$.
So (subset - superset) is always empty.

Consider a relational table with a single record for each registered student with the following attributes.

1. Registration_Num: Unique registration number of each registered student
2. UID: Unique identity number, unique at the national level for each citizen
3. BankAccount_Num: Unique account number at the bank. A student can have multiple accounts or joint accounts. This attribute stores the primary account number.
4. Name: Name of the student
5. Hostel_Room: Room number of the hostel

Which one of the following options is INCORRECT?

A BankAccount_Num is a candidate key

B Registration_Num can be a primary key

C UID is a candidate key if all students are from the same country

D If S is a superkey such that $S \cap \text{UID}$ is NULL then $S \cup \text{UID}$ is also a superkey

Consider a relational table with a single record for each registered student with the following attributes.

1. Registration_Num: Unique registration number of each registered student
2. UID: Unique identity number, unique at the national level for each citizen
3. BankAccount_Num: Unique account number at the bank. A student can have multiple accounts or join accounts. This attribute stores the primary account number.
4. Name: Name of the student
5. Hostel_Room: Room number of the hostel

Which one of the following option is INCORRECT?

- A **BankAccount_Num is a candidate key**
- B Registration_Num can be a primary key
- C UID is a candidate key if all students are from the same country
- D If S is a superkey such that $S \cap \text{UID}$ is NULL then $S \cup \text{UID}$ is also a superkey

Database table by name Loan_Records is given below.

Borrower Bank_Manager Loan_Amount

Ramesh Sunderajan 10000.00

Suresh Ramgopal 5000.00

Mahesh Sunderajan 7000.00

What is the output of the following SQL query?

```
SELECT Count(*)
```

```
FROM ( ( SELECT Borrower, Bank_Manager
```

```
FROM Loan_Records) AS S
```

```
NATURAL JOIN ( SELECT Bank_Manager,  
Loan_Amount
```

```
FROM Loan_Records) AS T );
```

A 3

B 9

C 5

D 6

S

Borrower	Bank – Manager
Ramesh	Sunderajan
Suresh	Ramgopal
Mahesh	Sunderjan

T

Bank – Manager	Loan - Amount
Sunderajan	10000.00
Ramgopal	5000.00
Sunderjan	7000.00

After executing the given query, the output would be

Borrower	Bank – Manager	Load – Amount
Ramesh	Sunderajan	10000.00
Ramesh	Sunderajan	7000.00
Suresh	Ramgopal	5000.00
Mahesh	Sunderajan	10000.00
Mahesh	Sunderajan	7000.00

Database table by name Loan_Records is given below.

Borrower Bank_Manager Loan_Amount

Ramesh Sunderajan 10000.00

Suresh Ramgopal 5000.00

Mahesh Sunderajan 7000.00

What is the output of the following SQL query?

```
SELECT Count(*)
```

```
FROM ( ( SELECT Borrower, Bank_Manager
```

```
        FROM Loan_Records) AS S
```

```
        NATURAL JOIN ( SELECT Bank_Manager,  
Loan_Amount
```

```
                FROM Loan_Records) AS T );
```

A 3

B 9

C 5

D 6

62. Consider a relation book (title, price) which 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 as B  
where (select count (*)  
       from book as T  
       where T.price > B.price) < 7
```

- (a) Titles of the six most expensive books.
- (b) Title of the sixth most expensive books.
- (c) Titles of the seven most expensive books.
- (d) Title of the seventh most expensive books.

62. Consider a relation book (title, price) which 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 as B  
where (select count (*)  
       from book as T  
       where T.price > B.price) < 7
```

- (a) Titles of the six most expensive books.
- (b) Title of the sixth most expensive books.
- (c) Titles of the seven most expensive books. ✓
- (d) Title of the seventh most expensive books.

Transaction Management

Transaction Management

A sequence of many actions which are considered to be one atomic unit of work. A transaction is a collection of operations involving data items in a database. There are four important properties of transactions that a DBMS must ensure to maintain data in the face of concurrent access and system failures.

Concurrency Control: An Overview

In the context of database systems, a **transaction** is a set of database operations induced by a single user or application that should be **considered as one undividable unit of work**. For a database management system to have a valid transaction mechanism, it is important that it has the **ACID (Atomicity, Consistency, Isolation, Durability)** properties.

A **transaction manager** in the database management system takes responsibility for coordinating transaction executions. When new, to-be-executed transactions come, the transaction manager will put them in a **scheduler**.

Atomicity

Atomicity requires that each transaction is all or nothing. If one part of the transaction fails, the entire transaction fails and the database state is left unchanged.

Consistency

If each transaction is consistent and the data base starts on as consistent, it ends up as consistent

Isolation

Execution of one transaction is isolated from that of another transactions. It ensures that concurrent execution of transaction results in a system state that would be obtained if transaction were executed serially, i.e., one after the other.

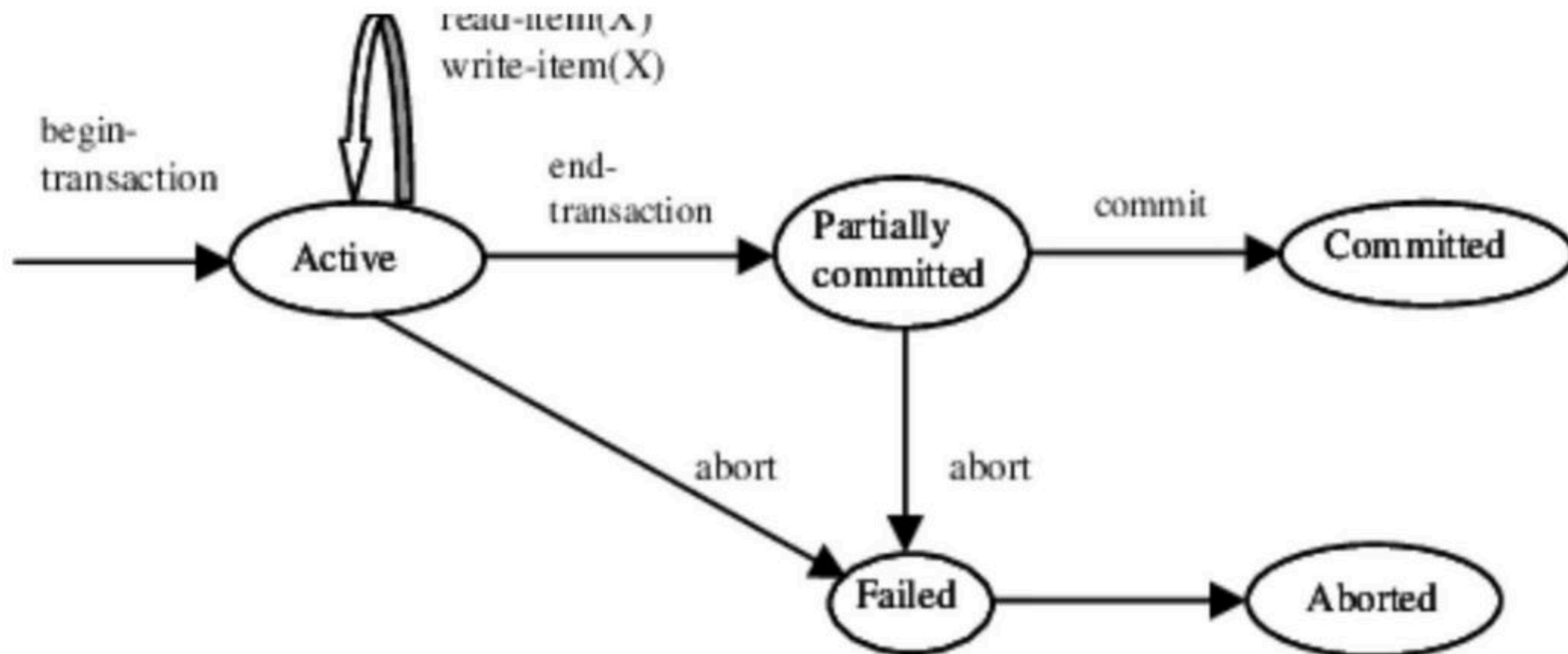
Durability

Durability means that once a transaction has been committed, it will remain even in the event of power loss, crashes, or errors. In a relational database, for instance, once a group of SQL statements execute, the results need to be stored permanently.

If a transaction commits, its effects persist.

- A transaction starts with any SQL statement and ends with a COMMIT or ROLLBACK.
- COMMIT statement makes changes permanent to the database.
- ROLLBACK statement reverses changes.

A transaction includes one or more database access operations. These can include insertion, deletion, modification or retrieval operations.



Definition of SCHEDULE

The sequence of the read/write operations of several transactions as they are executed in the Database

Serial Schedule and Serializable Schedule

There are two types of a scheduler, **serial schedule** and **serializable schedule**. Yes, they do seem like synonyms but are in fact more like antonyms.

In a serial schedule, all statements of the same transaction are executed consecutively, without any interleave with statements from a different transaction.

On the other hand, **serializable schedules** have two characteristics:

- (1) they are non-serial schedules, which allow concurrent executions of different transactions
- (2) they still produce correct results.

Serial Schedule

- Transactions execute fully.
- One at a time.
- No interleaving.
- Different orders of execution may produce different final values

Serializable Schedule

- Interleaved.
- Equivalent to SOME serial schedule.
- Equivalence does NOT mean "ending up with the same values as".
- Equivalence cannot depend on initial values of database items.
- Cannot depend on values written
DB doesn't know logic of transaction.
- Depends only on order of operations.

The transactions are interleaved, means second transaction is started before the first one could end. And execution can switch between the transactions back and forth. It can also switch between multiple transactions. This could actually cause inconsistency in the system. But they are handled by the database systems.

The Goal

With a serial schedule, the ACID properties can be easily achieved. However, in our current world where **parallelism and concurrency** is the norm, using a serial schedule would seriously under-exploit the potential these hardware technologies provide.

So **we aim at serializable schedules**, which have the integrity of a serial schedule as well as the advantage of efficiency.

The study of **concurrency control**, then, is to coordinate transactions that execute simultaneously in a way that guarantees these transactions do not cause any data inconsistencies from mutual interference.

Typical Concurrency Problems

Any design in the field of concurrency control would be tested against these five typical interference problems:

- 1.The Lost Update Problem
- 2.The Uncommitted Dependency Problem
- 3.The Inconsistent Analysis Problem
- 4.Non-repeatable Read
- 5.Phantom Read

Lost Update Problem

happens when two transactions access and update the same data simultaneously, and the result of one is overwritten by another. For example, both transaction T1 and T2 is adding 3 to a variable x, whose original value is 0. The operations in both transactions are as follows:

1. Read the value of x.
2. Add 3 to the value.
3. Assign it back to x.

Now, in a concurrent environment, what might happen is T1 and T2 read the value of x, which is 0, at the same time. They both add 3 to that value and return. When all operations are done, our x now has the value 3, when it should have been 6. This problem results in an inconsistent database state.

Uncommitted Dependency Problem (Dirty Read Problem)

Suppose today we have a data that is being updated and eventually rolled back in a transaction. **A dirty read** occurs when another transaction, running concurrently, reads that data at the exact point when it is updated but not yet rolled back.

This problem results in either an inconsistent database or incorrect query result, depending on what of that dirty-reading transaction is.

Inconsistent Analysis Problem

Inconsistent analysis happens when a **transaction reads a data which another transaction is still updating**. It is almost the same problem with lost update, except that the final database state is actually consistent, because that other transaction does not modify the data. Nevertheless, the query result is still incorrect.

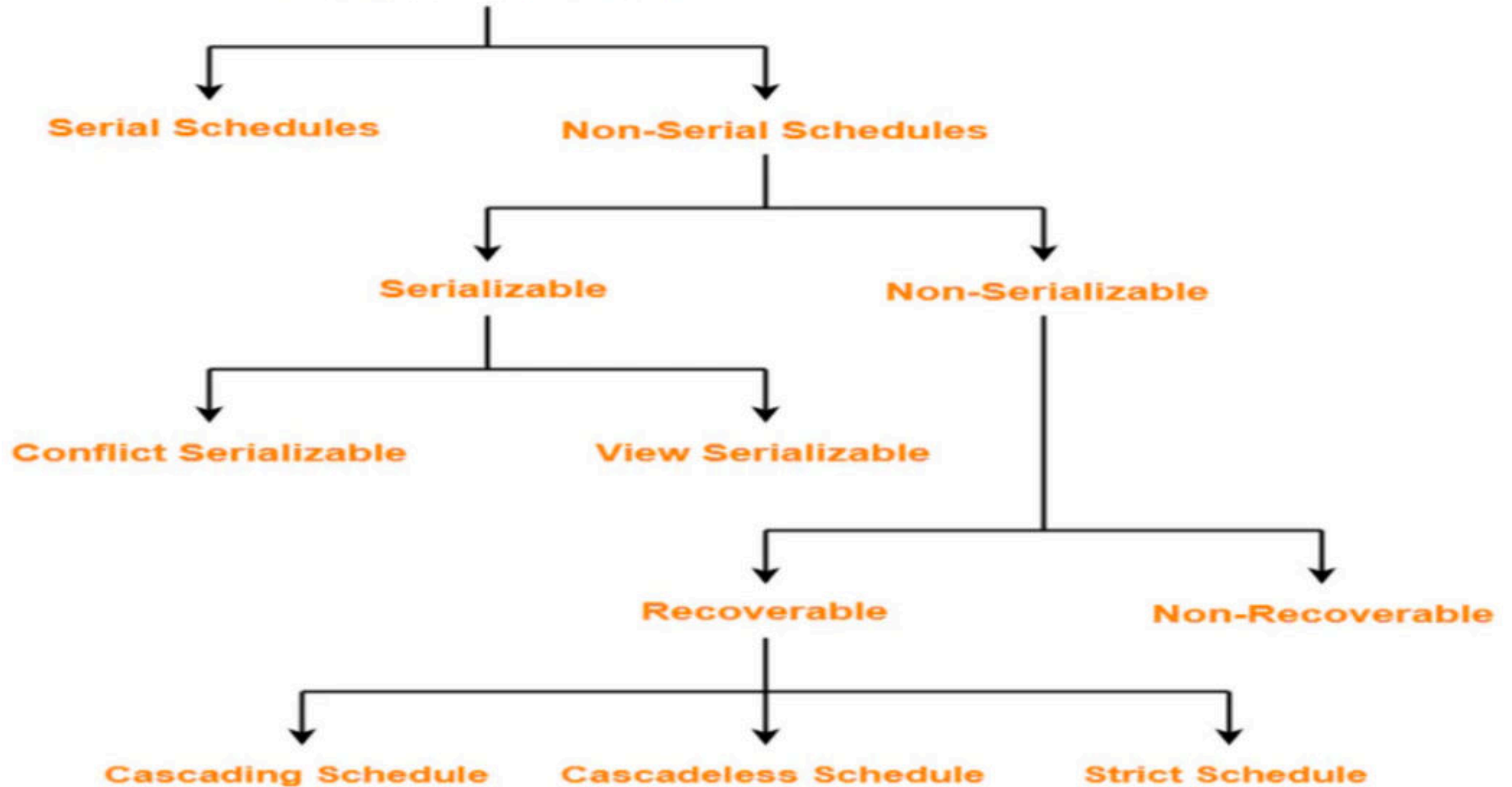
Non-repeatable Read

Non-repeatable read occurs when a transaction **reads the same row multiple times but obtains different subsequent values**, because another transaction is updating the row at the same time.

Phantom Read

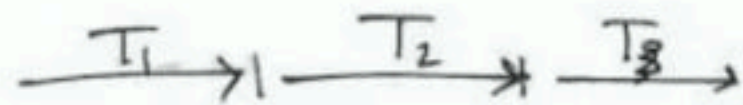
Phantom read is a problem when a transaction is executing **insert or delete operations on a set of rows that are being read by another transaction**.

Schedules in DBMS



Serial Schedule

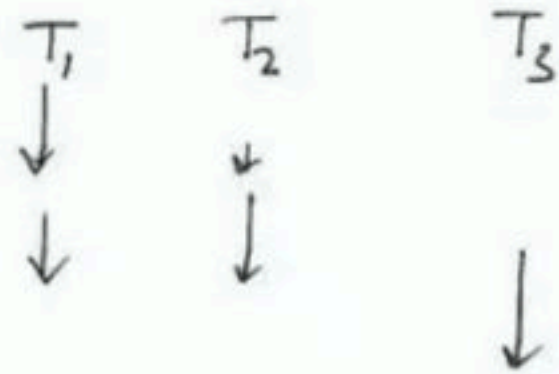
If one transaction have started then others will wait till its completion.



Assure consistency but more waiting time.

Parallel Schedule

Multiple transactions are executed parallelly by switching.

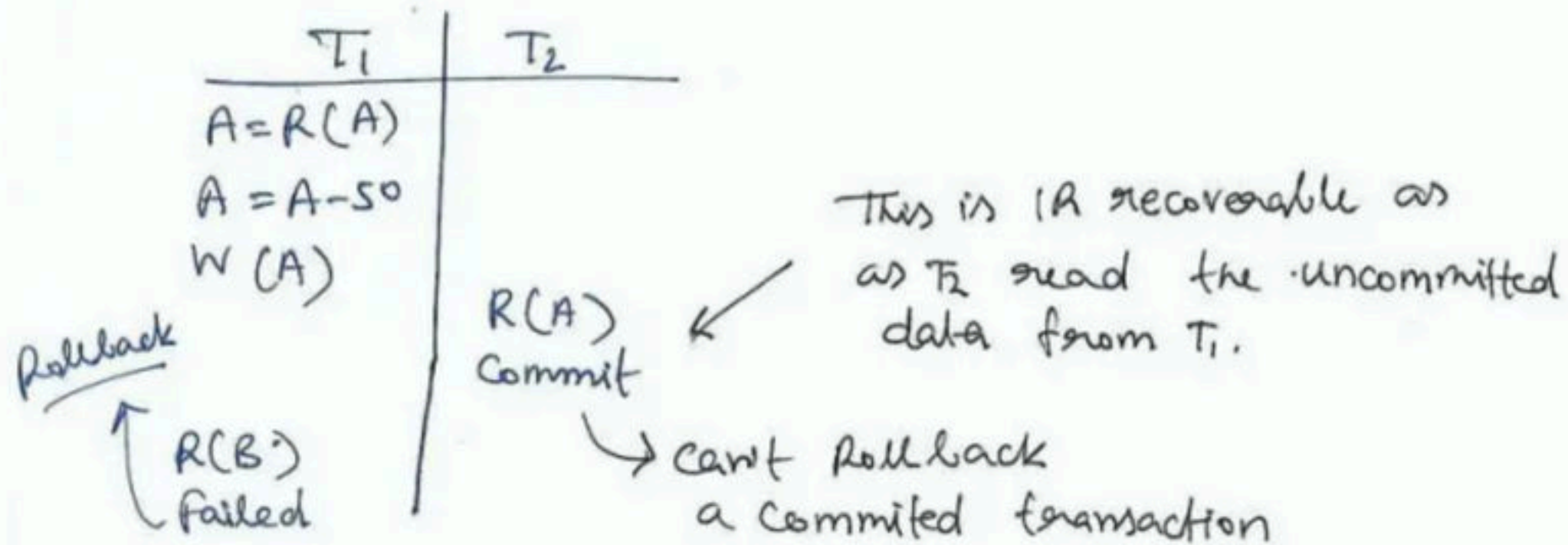


Less waiting time

Recoverability

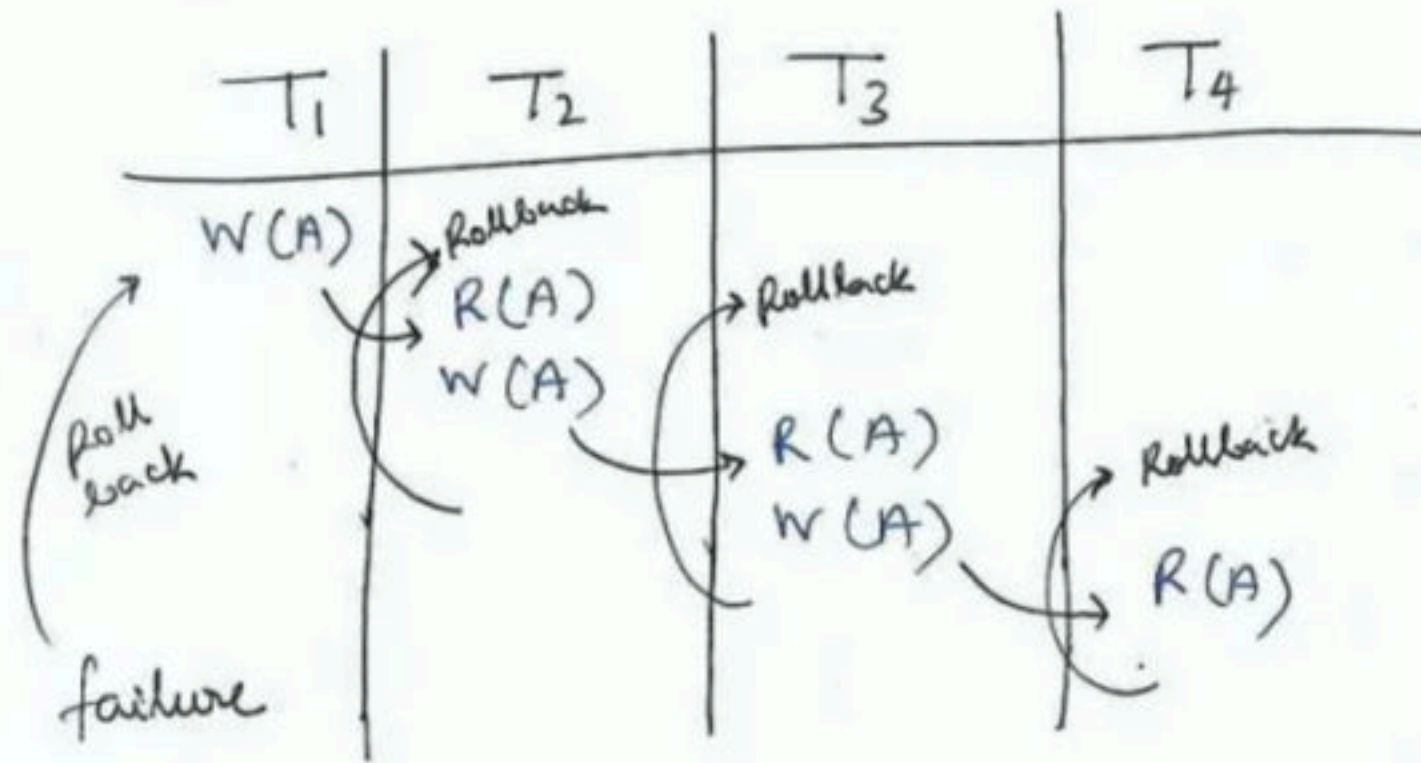
If T_2 is reading a data item which is written by T_1 then T_2 should commit after T_1

IR-Recoverable



Cascading Rollback

Failure of one transaction results in rollback of other transactions.



IR-Recoverable
IR-REC

T ₁	T ₂
W(A)	R(A) Commit

Commit

T₂ commit
before T₁

Recoverable
REC

T ₁	T ₂
W(A) Commit	R(A) Commit

T₂ commit
after T₁

Cascades
Recoverable
CLR

T ₁	T ₂
W(A) Commit	R(A)

Should
not be any
Uncommitted
Read

Strict Recoverable

T ₁	T ₂
W(A) Commit	R(A) OR W(A)

No uncommitted
Read or
write is
allowed.

Q.43

Two concurrent executing transactions T_1 and T_2 are allowed to update same stock item say 'A' in an uncontrolled manner. In such scenario, following problems may occur :

- (a) Dirty read problem
- (b) Lost update problem
- (c) Transaction failure
- (d) Inconsistent database state

Which of the following option is correct if database system has no concurrency module and allows concurrent execution of above two transactions?

- (1) (a), (b) and (c) only
- (2) (c) and (d) only
- (3) (a) and (b) only
- (4) (a), (b) and (d) only

NET Dec 2019

Q.43

Two concurrent executing transactions T_1 and T_2 are allowed to update same stock item say 'A' in an uncontrolled manner. In such scenario, following problems may occur :

- (a) Dirty read problem
- (b) Lost update problem
- (c) Transaction failure
- (d) Inconsistent database state

Which of the following option is correct if database system has no concurrency module and allows concurrent execution of above two transactions?

- (1) (a), (b) and (c) only
- (2) (c) and (d) only
- (3) (a) and (b) only
- (4) (a), (b) and (d) only



NET Dec 2019

Serializability in DBMS-

- Some non-serial schedules may lead to inconsistency of the database.
- Serializability is a concept that helps to identify which non-serial schedules are correct and will maintain the consistency of the database.

Serializable Schedules-

If a given non-serial schedule of 'n' transactions is equivalent to some serial schedule of 'n' transactions, then it is called as a **serializable schedule**.

Characteristics-

Serializable schedules behave exactly same as serial schedules. Thus, serializable schedules are always-

- Consistent
- Recoverable
- Casacadeless
- Strict



Conflict Serializability-

If a given non-serial schedule can be converted into a serial schedule by swapping its non-conflicting operations, then it is called as a **conflict serializable schedule**.

Conflicting Operations-

Two operations are called as **conflicting operations** if all the following conditions hold true for them-

- Both the operations belong to different transactions
- Both the operations are on the same data item
- At least one of the two operations is a write operation

Example: –

- **Conflicting** operations pair $(R_1(A), W_2(A))$ because they belong to two different transactions on same data item A and one of them is write operation.
- Similarly, $(W_1(A), W_2(A))$ and $(W_1(A), R_2(A))$ pairs are also **conflicting**.
- On the other hand, $(R_1(A), W_2(B))$ pair is **non-conflicting** because they operate on different data item.
- Similarly, $(W_1(A), W_2(B))$ pair is **non-conflicting**.

Example:

Swapping is possible only if S1 and S2 are logically equal.

1. T1: Read(A) T2: Read(A)

T1	T2
Read(A)	
	Read(A)

Swapped



T1	T2
	Read(A)
Read(A)	

Schedule S1

Schedule S2

Here, $S1 = S2$. That means it is non-conflict.

2. T1: Read(A) T2: Write(A)

T1	T2
Read(A)	Write(A)

Swapped



T1	T2
Read(A)	Write(A)

Schedule S1

Schedule S2

Here, $S1 \neq S2$. That means it is conflict.

Conflict Equivalent

In the conflict equivalent, one can be transformed to another by swapping non-conflicting operations. In the given example, S2 is conflict equivalent to S1 (S1 can be converted to S2 by swapping non-conflicting operations).

Two schedules are said to be conflict equivalent if and only if:

- 1.They contain the same set of the transaction.
- 1.If each pair of conflict operations are ordered in the same way.

Non-serial schedule

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

Schedule S1

Serial Schedule

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

Schedule S2

Schedule S2 is a serial schedule because, in this, all operations of T1 are performed before starting any operation of T2. Schedule S1 can be transformed into a serial schedule by swapping non-conflicting operations of S1.

After swapping of non-conflict operations, the schedule S1 becomes:

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

Since, S1 is conflict serializable.

Q) Consider the following schedules involving two transactions.

S1: r1(X) ; r1(Y) ; r2(X) ; r2(Y) ; w2(Y) ; w1(X)

S2 : r1(X) ; r2(X) ; r2(Y) ; w2(Y) ; r1(Y) ; w1(X)

Which one of the following statements is correct with respect to above ?

A) Both S1 and S2 are conflict serializable.

B) Both S1 and S2 are not conflict serializable.

C) S1 is conflict serializable and S2 is not conflict serializable.

D) S1 is not conflict serializable and S2 is conflict serializable.

Q) Consider the following schedules involving two transactions.

S1: r1(X) ; r1(Y) ; r2(X) ; r2(Y) ; w2(Y) ; w1(X)

S2 : r1(X) ; r2(X) ; r2(Y) ; w2(Y) ; r1(Y) ; w1(X)

Which one of the following statements is correct with respect to above ?

A) Both S1 and S2 are conflict serializable.

B) Both S1 and S2 are not conflict serializable.

C) S1 is conflict serializable and S2 is not conflict serializable.

D) S1 is not conflict serializable and S2 is conflict serializable.

S_1

T_1	T_2
$r(X)$ $r(Y)$	$r(X)$ $r(Y)$ $w(Y)$
$w(X)$	



Cycle exists, so it is not Conflict serializable.

S_2

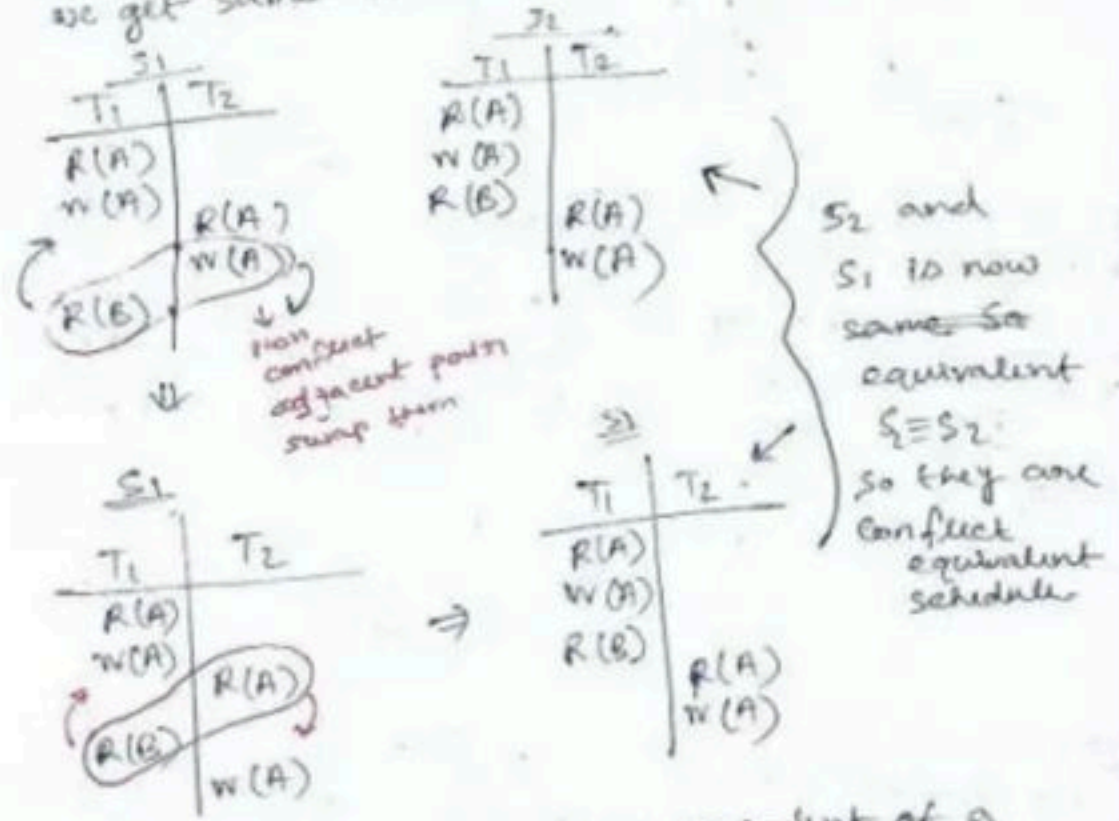
T_1	T_2
$r(X)$	$r(X)$ $r(Y)$ $w(Y)$
$r(Y)$ $w(X)$	



No cycle exists, so it is conflict serializable.

check Conflict Equivalent on mut

- check the adjacent pairs
- if they are non conflict pairs then swap them, otherwise no change in position
- After performing all possible swapping if we get same schedule then conflict equivalent



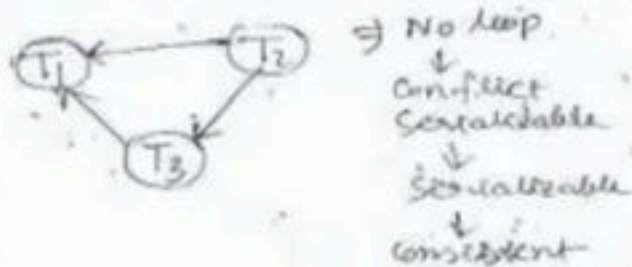
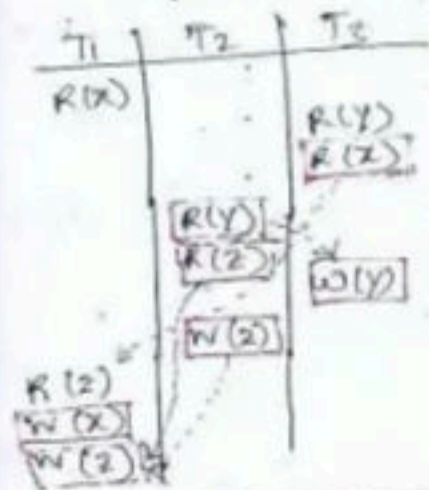
If there exists a conflict equivalent of a schedule then it is realizable.

Check conflict serializable and find the equivalent serial schedule.

→ draw precedence graph
 → check conflict pairs in other transaction and draw edges.

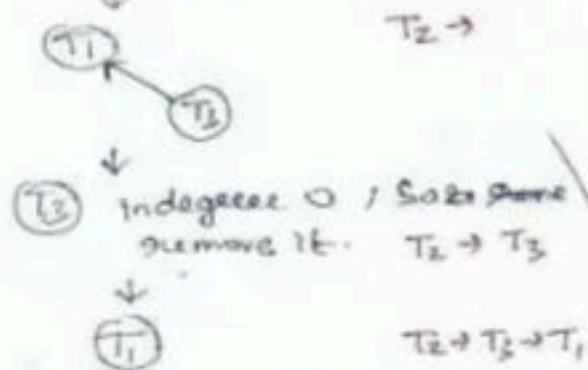
→ If **No loop/cycle** the conflict serializable.

No-loop means it can be convert in serial schedule
 loop then not conflict-serializable.



No to get equivalent serial schedule.

→ check in degree 0 vertex
 T2 is remove that



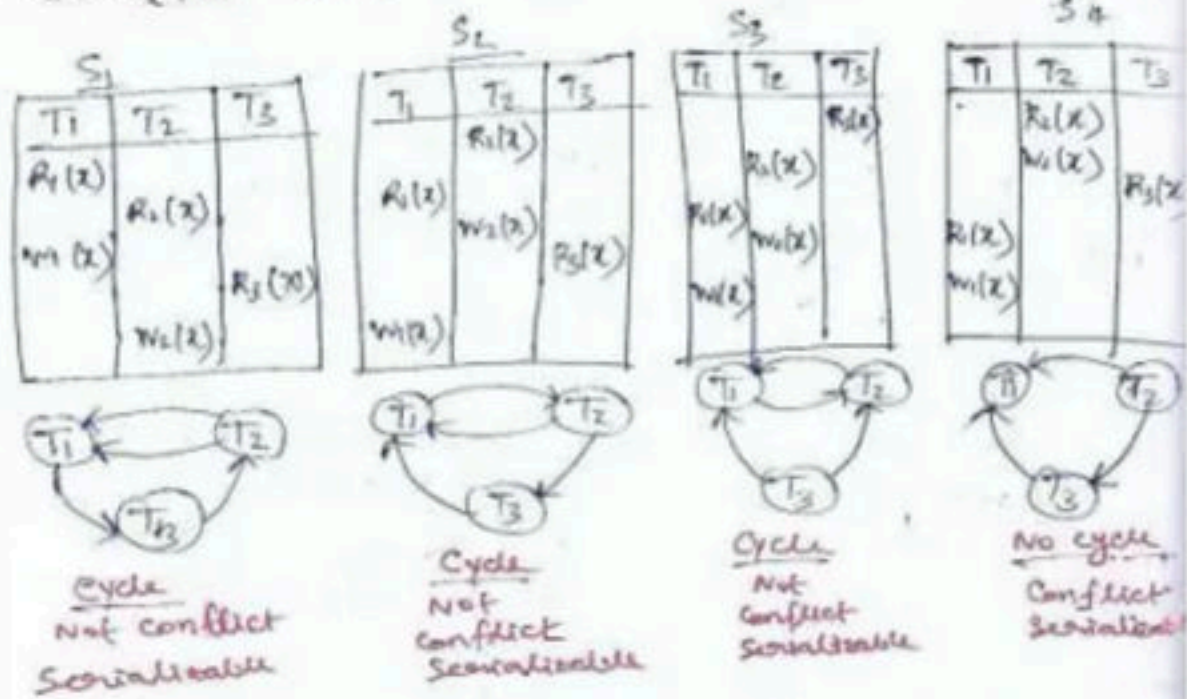
So the equivalent serial schedule is T2 → T3 → T1

If a schedule is not conflict serializable then we can't say that the schedule is not serializable.

For confirmation we have to check the view serializability.

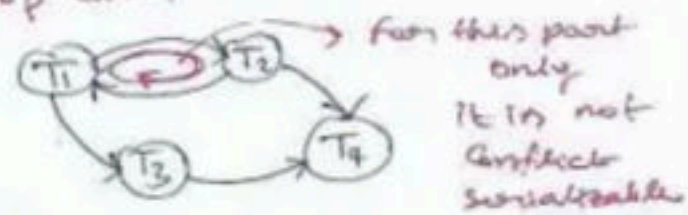
9) Which one is conflict serializable.

- a) $R_1(x) R_2(x) W_1(x) R_3(x) W_2(x) \rightarrow S_1$
- b) $R_2(x) R_1(x) W_2(x) R_3(x) W_1(x) \rightarrow S_2$
- c) $R_3(x) R_2(x) R_1(x) W_2(x) W_1(x) \rightarrow S_3$
- d) $R_2(x) W_2(x) R_3(x) R_1(x) W_1(x) \rightarrow S_4$



Only S_4 is conflict serializable.

* If any loop exist the not conflict serializable like



View Serializability

We can say that a schedule is Conflict-Serializable (means serial and consistent) iff its corresponding precedence graph does not have any loop/cycle.

There may be some schedules that are not Conflict-Serializable but still gives a consistent result because the concept of Conflict-Serializability becomes limited when the **Precedence Graph** of a schedule contains a **loop/cycle**. In such a case we cannot predict whether a schedule would be consistent or inconsistent.

So, to address such cases we brought the concept of **View-Serializability** because we did not want to confine the concept serializability only to Conflict-Serializability.

A schedule is said to be **View-Serializable** if it is view equivalent to a **Serial Schedule** (**where no interleaving of transactions is possible**).

Example

T₁

T₂

T₃

a=100

read(a)

a=a-40

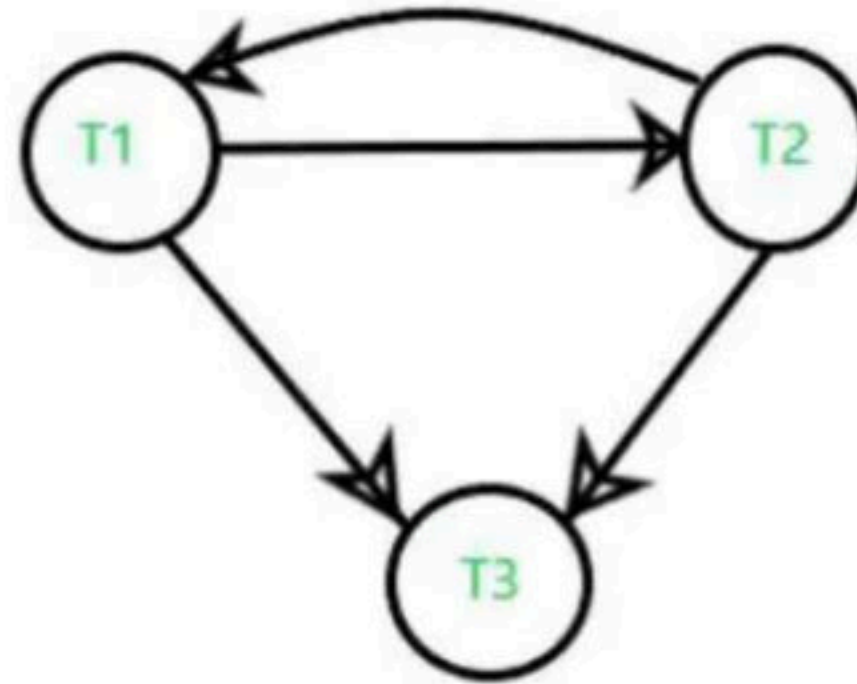
write(a) //60

a=a-40

write(a) //20

a=a-20

write(a) //0



graph contains cycle/loop which means it is not conflict-serializable but it does not mean that it cannot be consistent and equivalent to the serial schedule it may or may not be.

T₁ T₂ T₃

a=100
read(a)

a=a-40
write(a) //60

a=a-40
write(a) //20

a=a-20
write(a) //0

if we do **swapping** among some transaction's operation so our table will look like

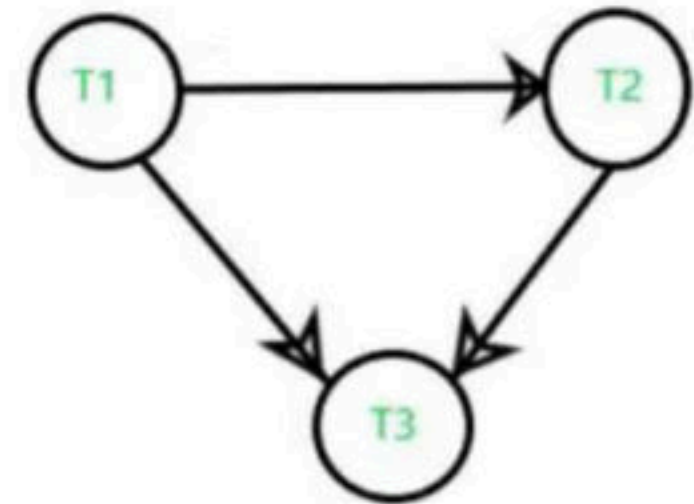
T₁ T₂ T₃

a=100
read(a) //100

a=a-40
write(a) //60

a=a-40
write(a) //20

a=a-20
write(a) //0



Now, we see that the precedence graph of the second table does not contain any cycle/loop, which means it is conflict serializable (equivalent to serial schedule, consistent) and the final result is coming the same as the first table.

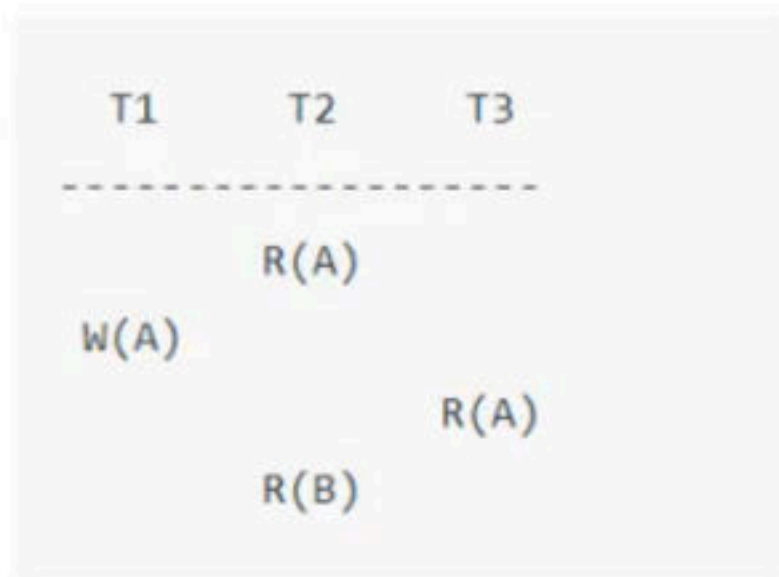
Methods to check View-Serializability of a schedule –

Method-1 :

Two schedules S1 and S2 are said to be view-equivalent if below conditions are satisfied :

1) Initial Read

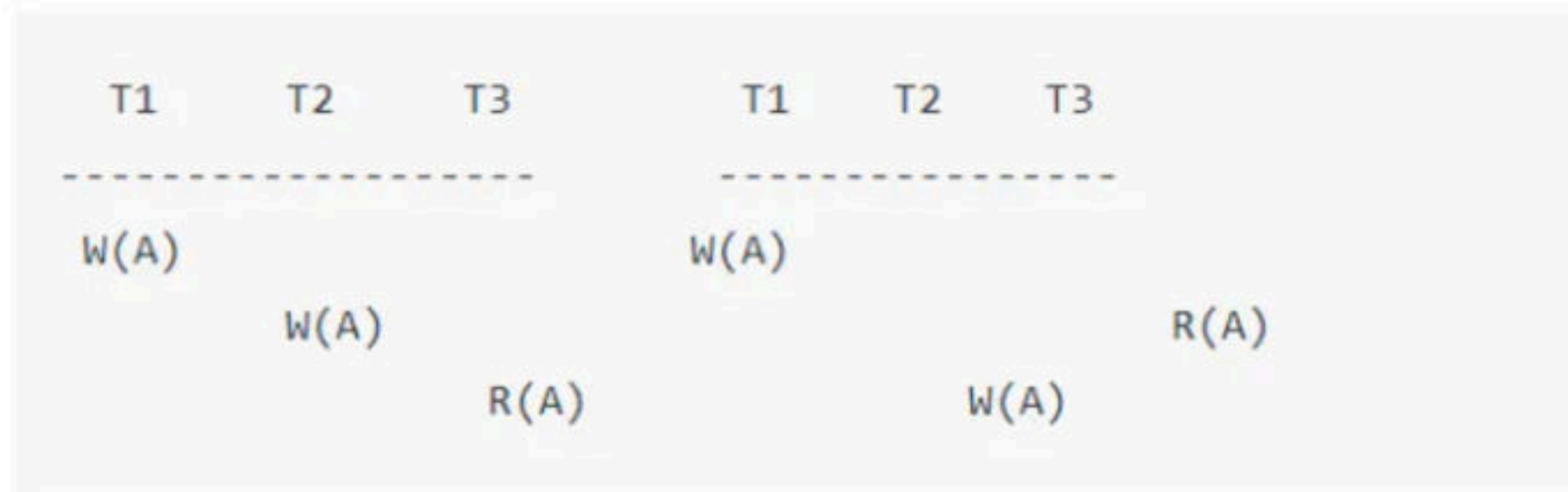
If a transaction T1 reading data item A from database in S1 then in S2 also T1 should read A from database.



Transaction T2 is reading A from database.

2) Updated Read

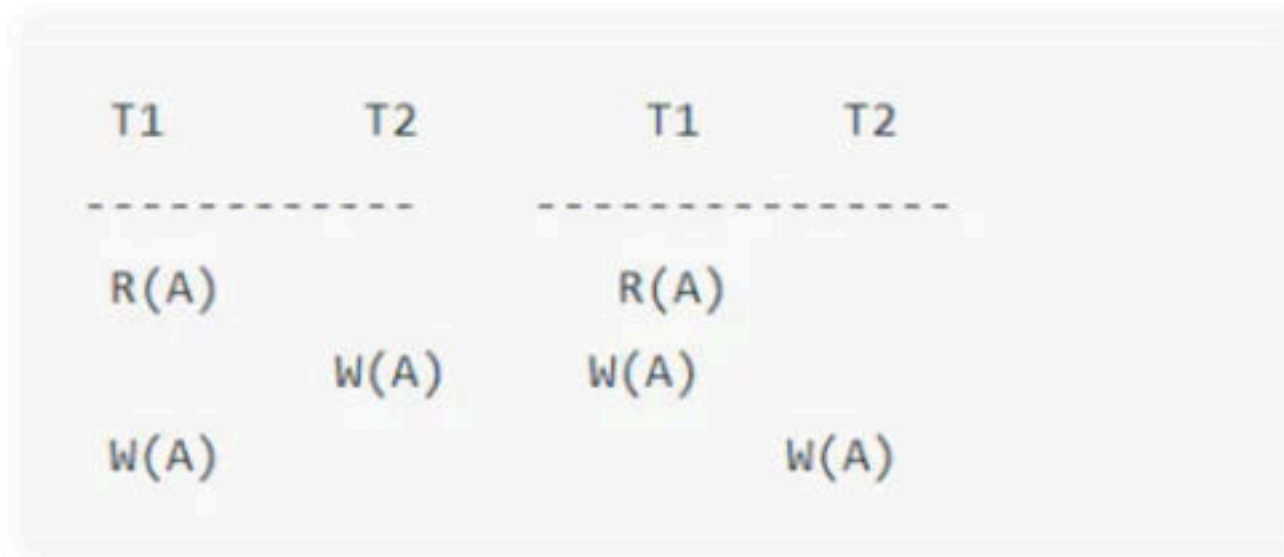
If T_i is reading A which is updated by T_j in S_1 then in S_2 also T_i should read A which is updated by T_j .



Above two schedules are not view-equivalent as in S_1 : T_3 is reading A updated by T_2 , in S_2 T_3 is reading A updated by T_1 .

3) Final Write operation

If a transaction T1 updated A at last in S1, then in S2 also T1 should perform final write operations.



Above two schedules are not view-equivalent as Final write operation in S1 is done by T1 while in S2 done by T2.

Method-2 :

First of all, check whether the given schedule is Non-Conflict Serializable or Conflict-Serializable –

- If the given schedule is conflict serializable (means its precedence graph does not contain any loop/cycle), then the given schedule must be a view serializable. Stop and submit your final answer.
- If the given schedule is non-conflict serializable, then it may or may not be view serializable. We cannot predict it just by using the concept of conflict serializability, So we need to look at the below cases.

After performing the above steps if you find the provided schedule is non-conflicting you need to perform the following steps –

Blind write : Performing the Writing operation (updatation), without reading operation, such write operation is known as a blind write.

- If no blind write exists, then the schedule must be a non-View-Serializable schedule. Stop and submit your final answer.
- If there exists any blind write, then, in that case, the schedule may or may not be view serializable. So we need to look at the below cases. Because, if it does not contain any blind-write, we can surely state that the schedule would not be View-Serializable.
- If the above two conditions do not work {means we have tried the above 2 conditions, then we have come to this step}. Then, draw a precedence graph using those dependencies. If no cycle/loop exists in the graph, then the schedule would be a View-Serializable otherwise not.

Log based Recovery

Deferred DB Modification

- always store new value in log
- Deferred means late.. it store updated new value only after commit in DB
- For recovery system must perform redo operation for the transaction that contains both start and commit in log record.
- Those transaction that contains only start but not commit log record, we perform rollback.

Immediate DB modification

- store new and old both in log
- after changing new value it immediately store in DB. didn't wait for commit.
- For recovery system must perform redo operation for the transaction that contains both start and commit in log record.
- Those transaction that contains only start but not commit log record, we perform undo taking old value from log and store in DB.

Q.65

Consider the following sequence of two transactions on a bank account (A) with initial balance 20,000 that transfers 5,000 to another account (B) and then apply 10% interest.

- (i) T1 start
- (ii) T1 A old = 20,000 new 15,000
- (iii) T1 B old = 12,000 new = 17,000
- (iv) T1 commit
- (v) T2 start
- (vi) T2 A old = 15,000 new = 16,500
- (vii) T2 commit

Suppose the database system crashes just before log record (vii) is written. When the system is restarted, which one statement is true of the recovery process ?

Options 1.

We need not redo log records (ii) and (iii) because transaction T1 has committed.

2.

Undo

We must redo log record (vi) to set A to 16,500 and then redo log records (ii) and (iii).

3.

We can apply redo and undo operations in arbitrary order because they are idempotent.

4. We must redo log record (vi) to set A to 16,500.

NET Dec 2018

In log based recovery we must perform 'Redo' operation for those transactions that contains both start and commit log record.

We perform 'undo' operation for those transaction that contains only start but not commit log record.

Therefore we perform 'Redo' of T1 and 'Undo' of T2.

Note: Actually they given option-2 is “ We must redo log record (vi) to set A to 16,500 and then redo log record (ii) and (iii)”.

But it should be undo as I have correct

Ans : option 2

Lock-Based Protocol

In this type of protocol, any transaction cannot read or write data until it acquires an appropriate lock on it. There are two types of lock:

1. Shared lock:

- It is also known as a Read-only lock. In a shared lock, the data item can only read by the transaction.
- It can be shared between the transactions because when the transaction holds a lock, then it can't update the data on the data item.

2. Exclusive lock:

- In the exclusive lock, the data item can be both reads as well as written by the transaction.
- This lock is exclusive, and in this lock, multiple transactions do not modify the same data simultaneously.

There are four types of lock protocols available:

1. Simplistic lock protocol

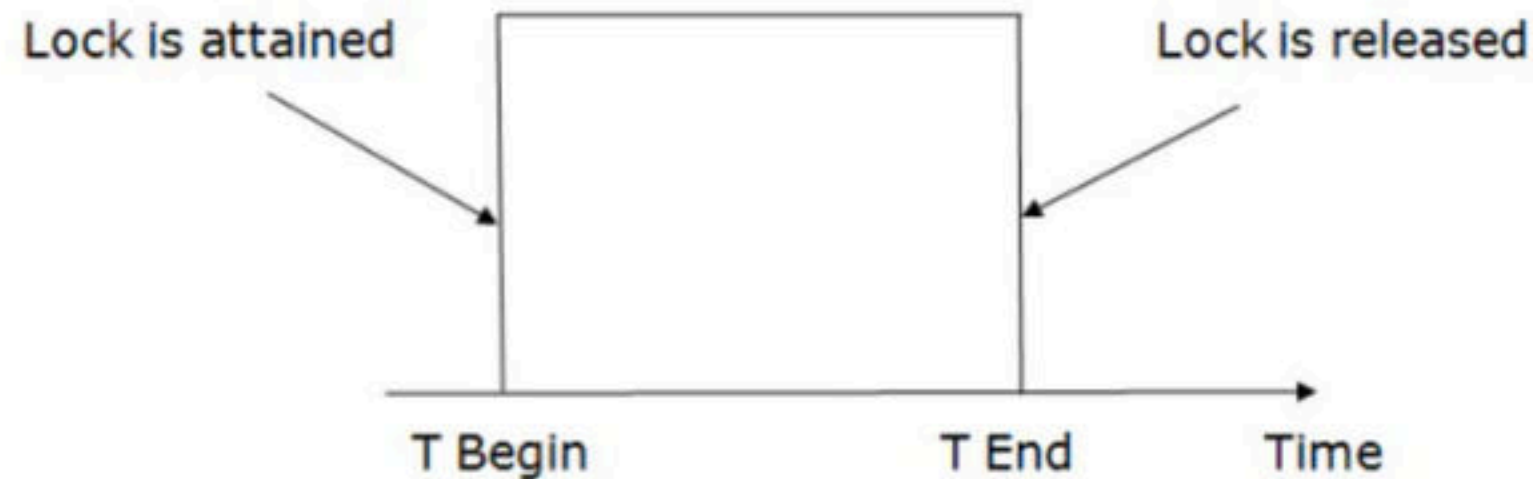
It is the simplest way of locking the data while transaction. Simplistic lock-based protocols allow all the transactions to get the lock on the data before insert or delete or update on it. It will unlock the data item after completing the transaction.

Implementing this lock system without any restrictions gives us the **Simple Lock-based protocol (or Binary Locking)**, but it has its own disadvantages, they do **not guarantee Serializability**. Schedules may follow the preceding rules but a non-serializable schedule may result.

To guarantee serializability, we must follow some additional protocol *concerning the positioning of locking and unlocking operations* in every transaction. This is where the concept of **Two-Phase Locking(2-PL)** comes into the picture, 2-PL ensures serializability.

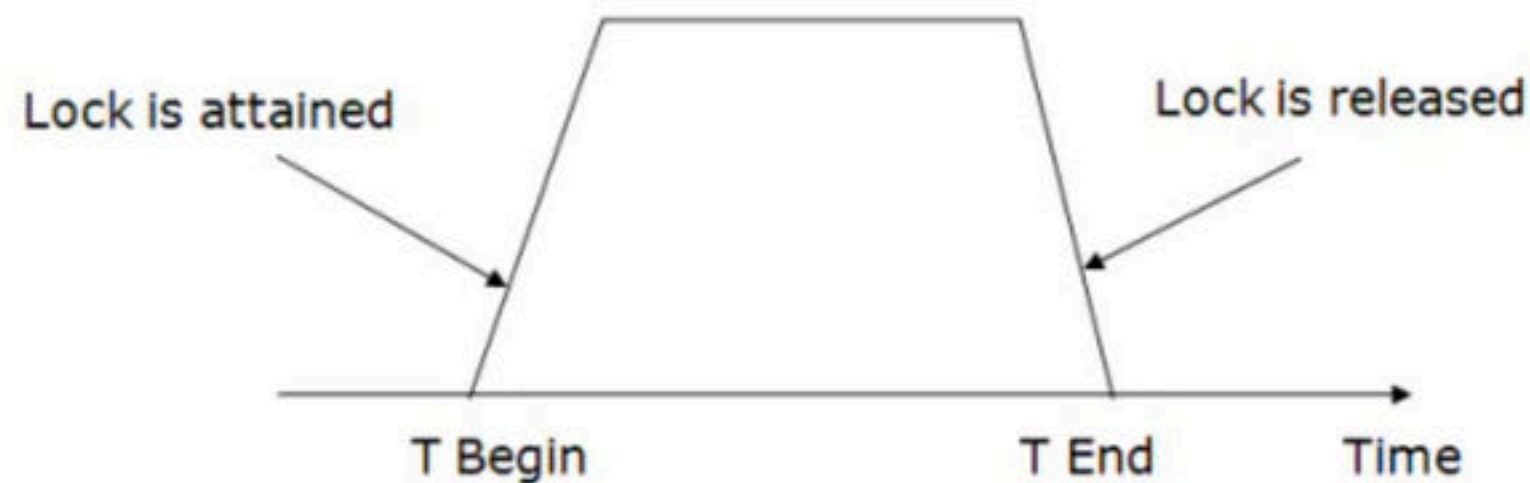
2. Pre-claiming Lock Protocol

- Pre-claiming Lock Protocols evaluate the transaction to list all the data items on which they need locks.
- Before initiating an execution of the transaction, it requests DBMS for all the lock on all those data items.
- If all the locks are granted then this protocol allows the transaction to begin. When the transaction is completed then it releases all the lock.
- If all the locks are not granted then this protocol allows the transaction to rolls back and waits until all the locks are granted.



3. Two-phase locking (2PL)

- The two-phase locking protocol divides the **execution phase of the transaction into three parts.**
- In the first part, when the execution of the transaction starts, it seeks permission for the lock it requires.
- In the second part, the transaction acquires all the locks. The third phase is started as soon as the transaction releases its first lock.
- In the third phase, the transaction cannot demand any new locks.** It only releases the acquired locks.



There are two phases of 2PL:

Growing phase: In the growing phase, a new lock on the data item may be acquired by the transaction, but none can be released.

Shrinking phase: In the shrinking phase, existing lock held by the transaction may be released, but no new locks can be acquired.

In the below example, if lock conversion is allowed then the following phase can happen:

1. Upgrading of lock (from S(a) to X (a)) is allowed in growing phase.
2. Downgrading of lock (from X(a) to S(a)) must be done in shrinking phase.

	T ₁	T ₂
1	lock-S(A)	
2		lock-S(A)
3	lock-X(B)	
4
5	Unlock(A)	
6		Lock-X(C)
7	Unlock(B)	
8		Unlock(A)
9		Unlock(C)
10

Transaction T₁:

- The growing Phase is from steps 1-3.
- The shrinking Phase is from steps 5-7.
- Lock Point at 3

Transaction T₂:

- The growing Phase is from steps 2-6.
- The shrinking Phase is from steps 8-9.
- Lock Point at 6

What is LOCK POINT? The Point at which the growing phase ends, i.e., when a transaction takes the final lock it needs to carry on its work.

2-PL ensures serializability, but there are still some drawbacks of 2-PL. Let's glance at the drawbacks:

- Cascading Rollback is possible under 2-PL.
- Deadlocks and Starvation are possible.

	T ₁	T ₂	T ₃
1	Lock-X(A)		
2	Read(A)		
3	Write(A)		
4	Lock-S(B) → LP	Rollback	
5	Read(B)		Rollback
6	Unlock(A), Unlock(B)		
7		Lock-X(A) → LP	
8		Read(A)	
9		Write(A)	
10		Unlock(A)	
11			Lock-S(A) → LP
12			Read(A)

FAIL → Rollback

LP - Lock Point

Read(A) in T₂ and T₃ denotes Dirty Read because of Write(A) in T₁.

Deadlock in 2-PL –

Schedule: Lock-X₁(A) Lock-X₂(B) Lock-X₁(B) Lock-X₂(A)

Drawing the precedence graph, you may detect the loop. So Deadlock is also possible in 2-PL.

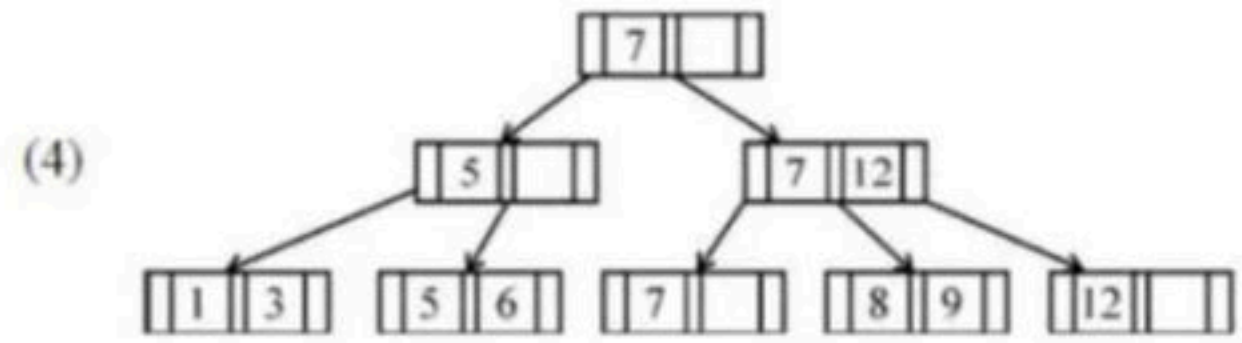
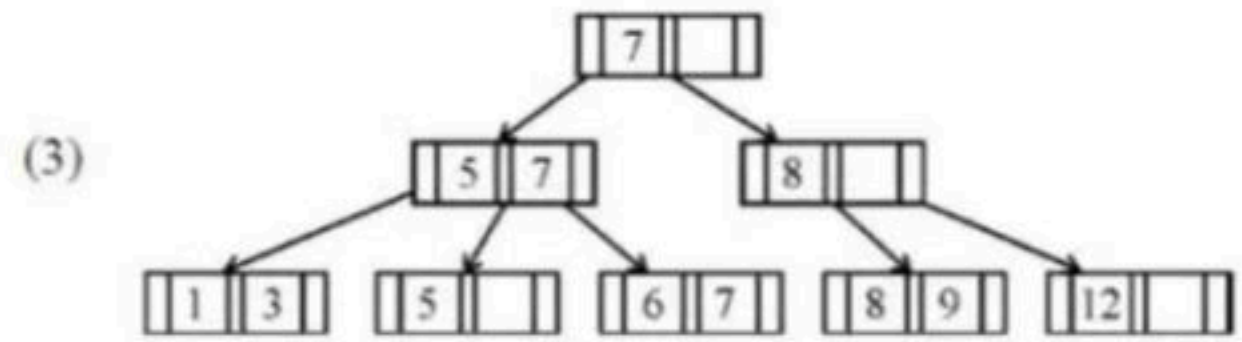
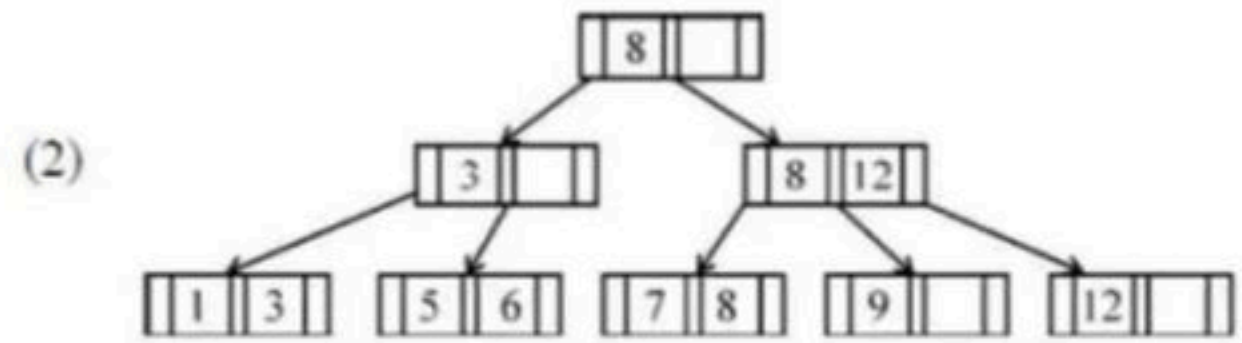
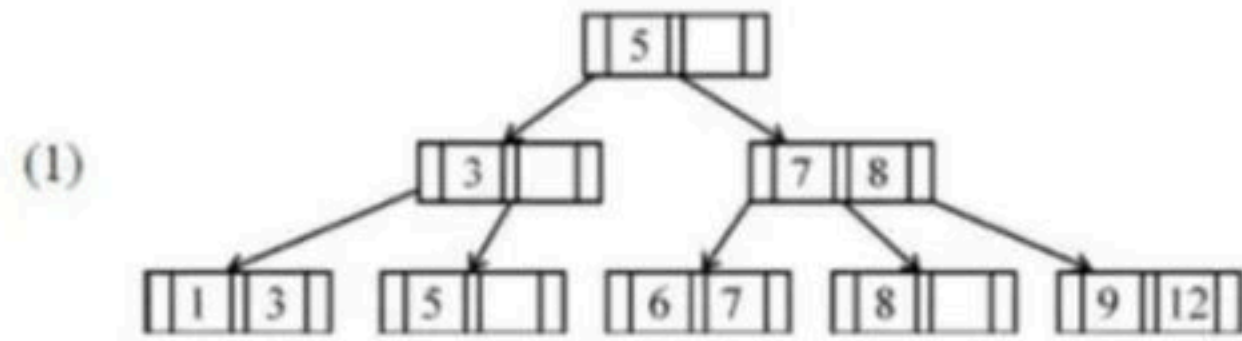
B Tree and B+ Tree

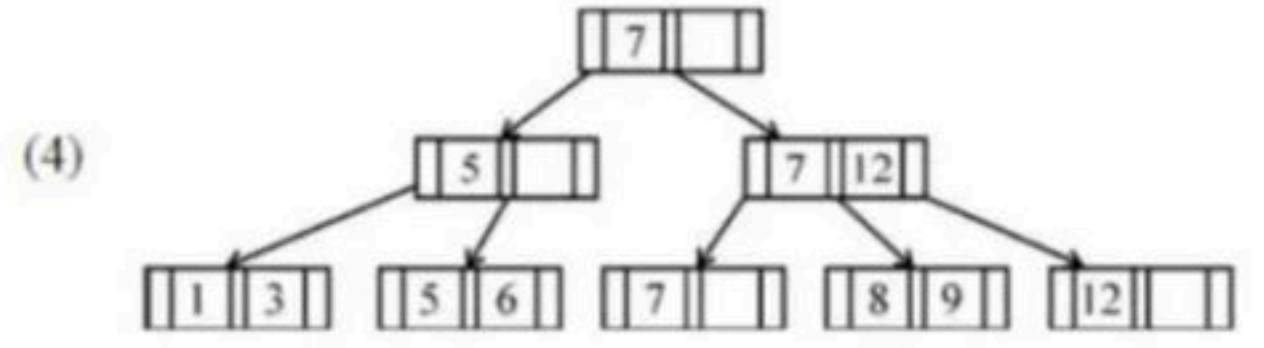
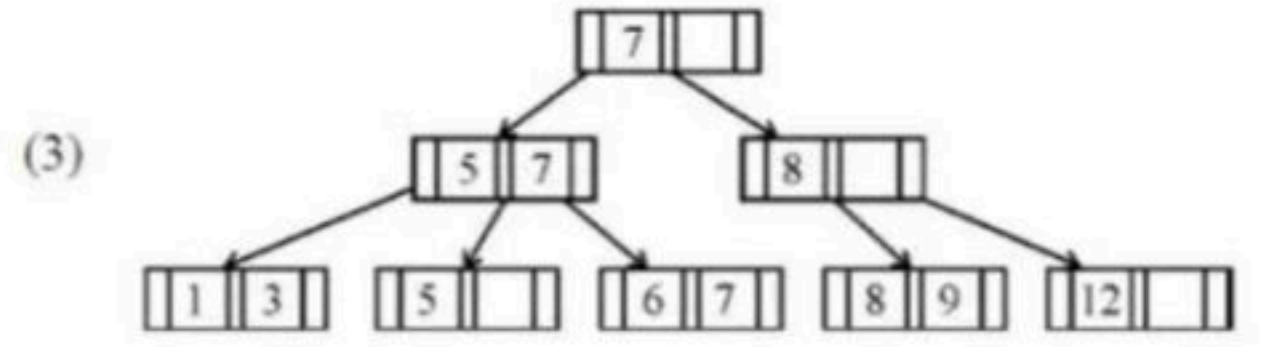
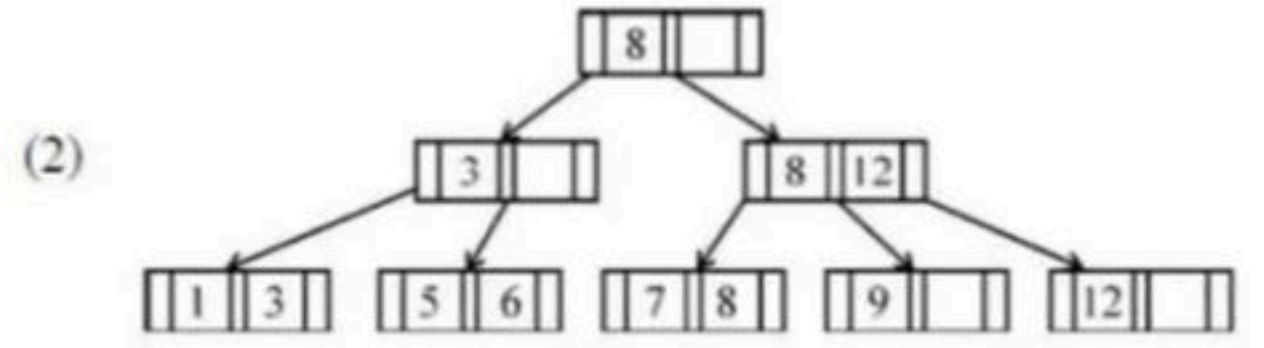
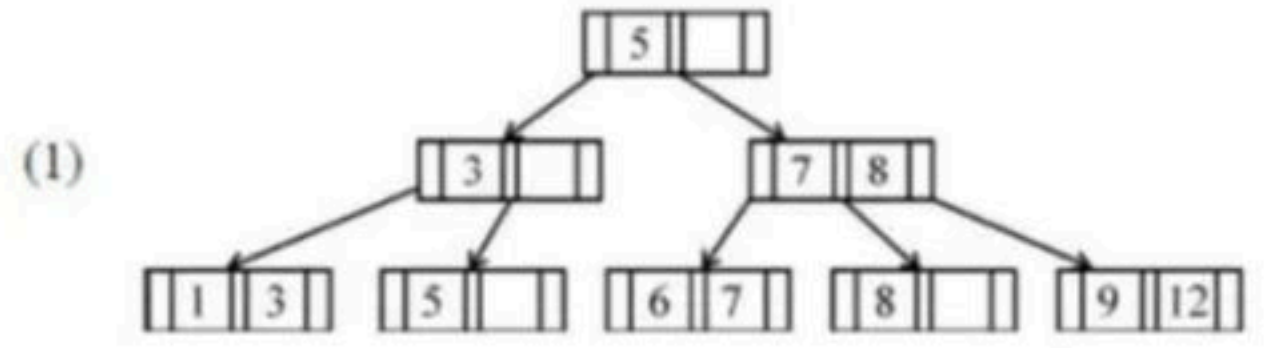
B Tree & B+ Tree

Q: If following sequence of keys are inserted in a B+ tree with $K(=3)$ pointers :

8, 5, 1, 7, 3, 12, 9, 6

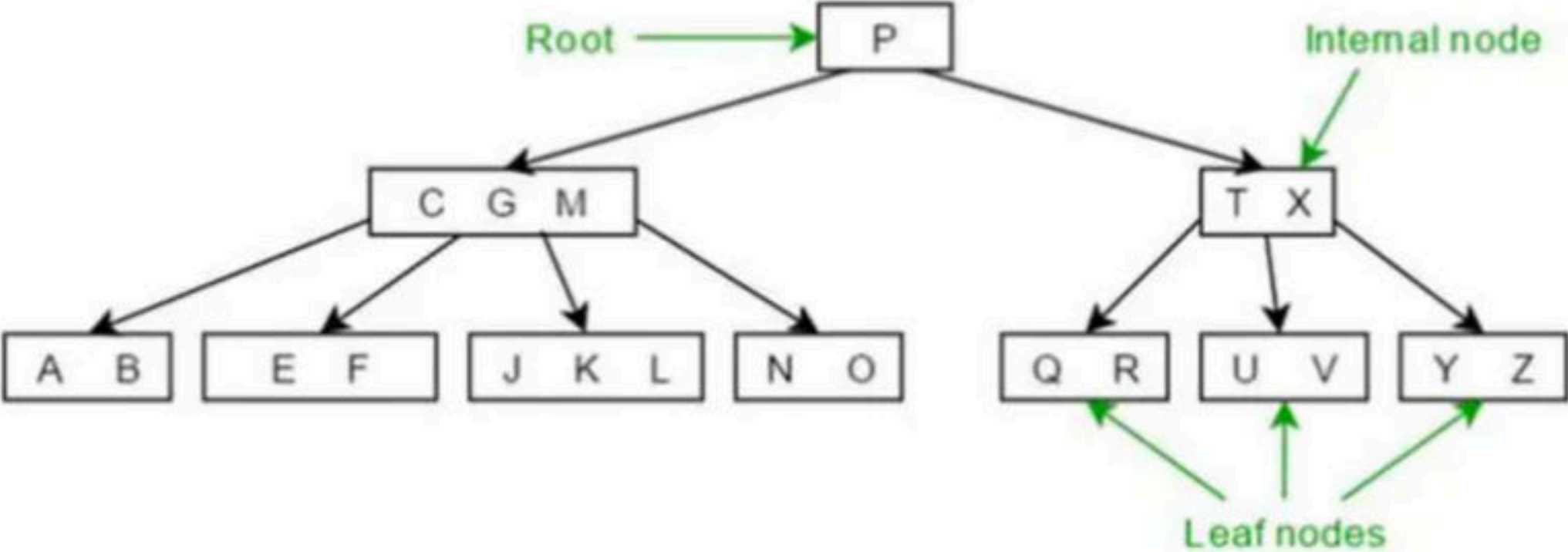
Which of the following shall be correct B+ tree?





Ans : Option 1

B TREES -DELETING KEYS RULES



B tree is a self-balancing search tree (the tree adjusts itself so that all the leaves are at the same depth) and contains multiple nodes which keep data in sorted order. Each node has 2 or more children, known as the **branching factor** and consists of multiple keys.

Following are the 5 properties of a B tree.

1. Every node x has the following:

- n – Number of keys
- key_i – The keys stored in ascending order
- leaf – Whether x is a leaf or not

2. Every node x has $x.n + 1$ children

3. The keys $x.key_i$ separate the ranges of keys stored in each sub-tree

4. All the leaves have the same depth, which is the tree height

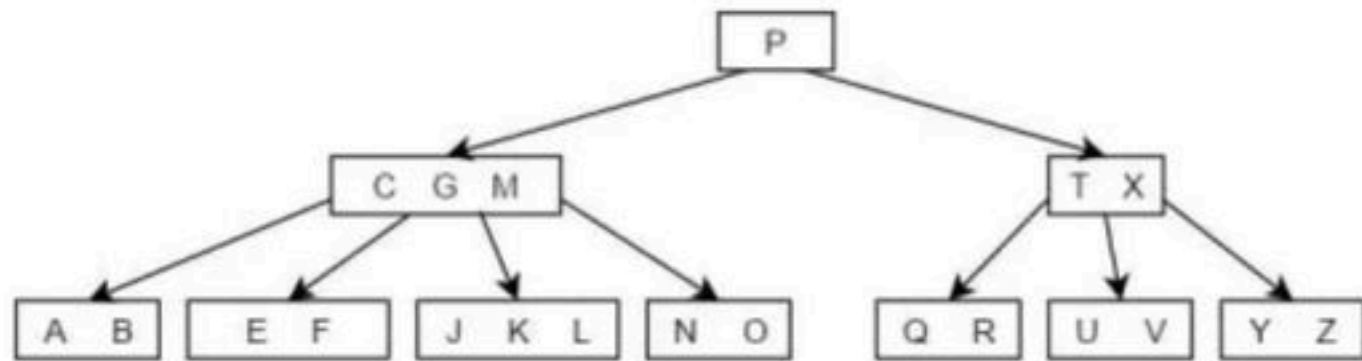
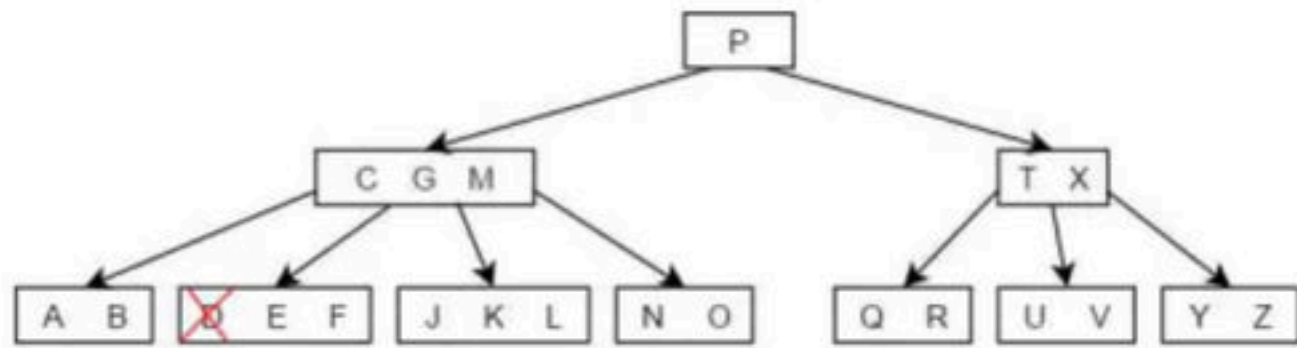
5. Nodes have lower and upper bounds on the number of keys that can be stored. We consider $t \geq 2$, called **minimum degree** (or **branching factor**) of the B tree.

- The root must have at least one key.

- Every other node must have at least $(t-1)$ keys and at most $(2t-1)$ keys. Hence, every node will have at least t children and at most $2t$ children. We say the node is full if it has $(2t-1)$ keys.

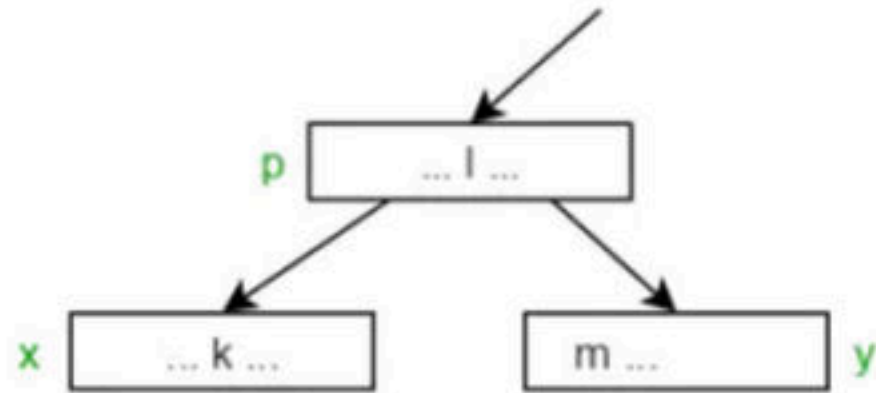
When deleting, you have to make sure that the 5 B tree properties are preserved. We consider 3 cases in deletion of B trees and we are going to delete the key k.

Case 1 : delete D from the B tree at the beginning.



Case 2

If k is in a node x which is a leaf and $x.n == t-1$



Case 2a:

Find the immediate sibling y of x , the extreme key m of y , the parent p of x and the parent key l of k

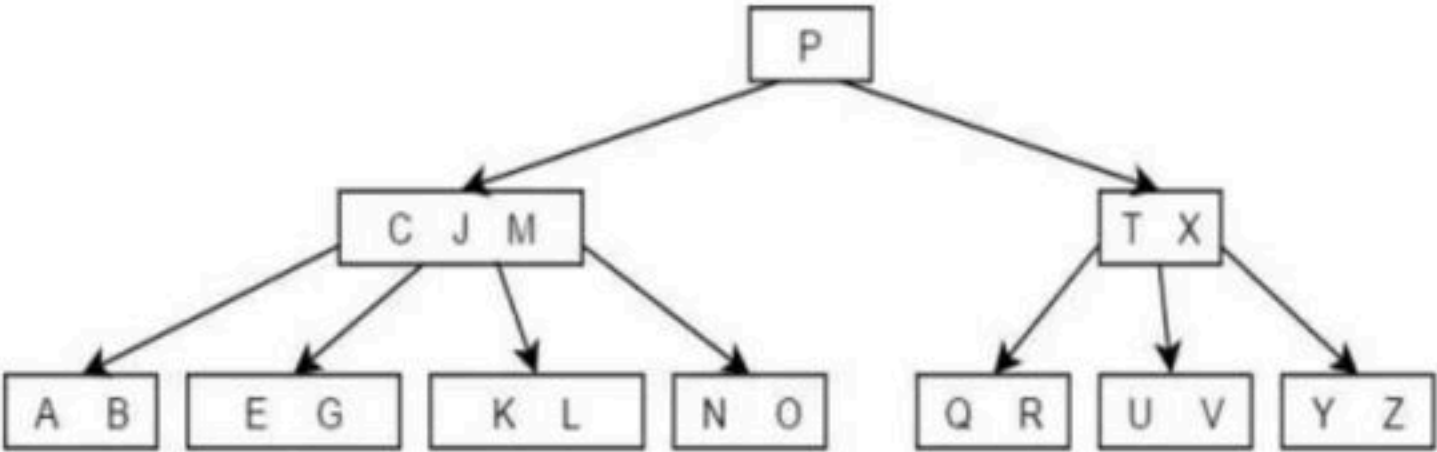
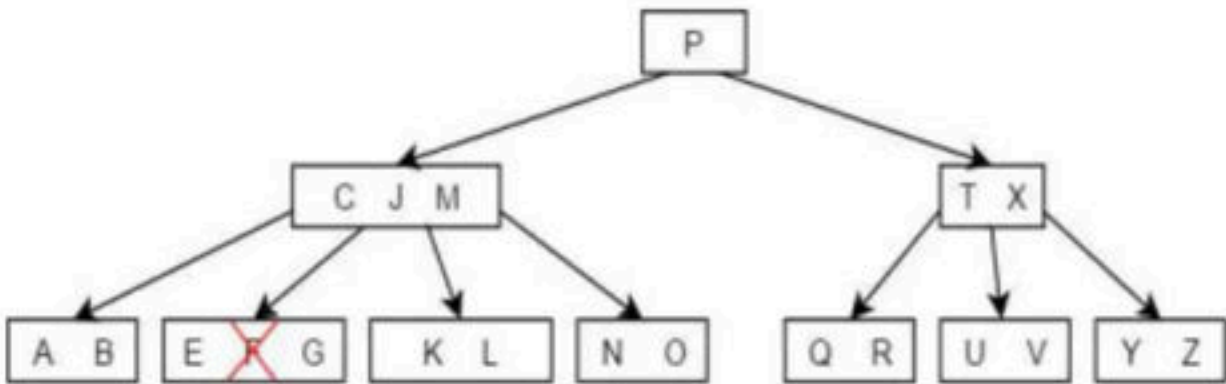
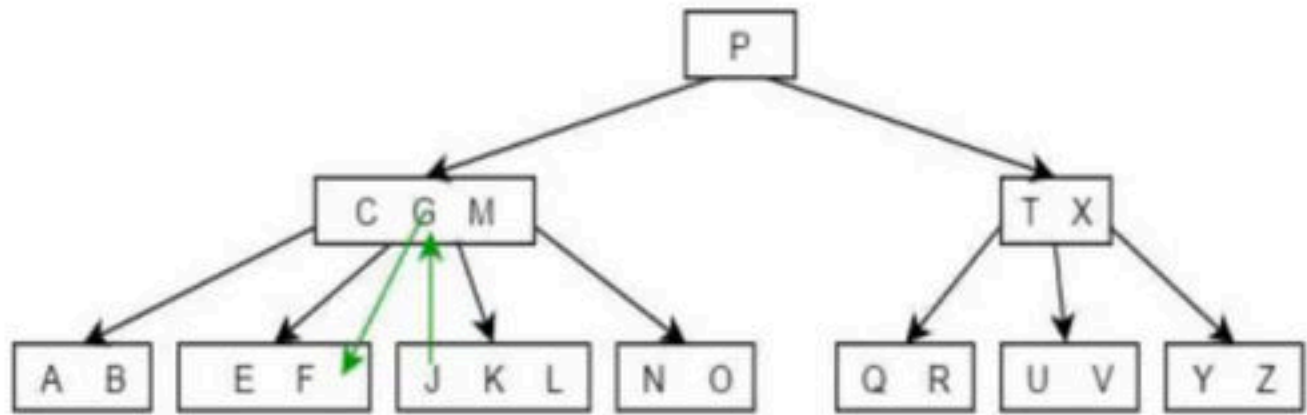
If $y.n \geq t$:

Move l of x into x

Move m of y to parent p

Delete k from x

Delete F from the B tree at the beginning.



Case 2b:

Find the immediate sibling y of x , the extreme key m of y , the parent p of x and the parent key l of k

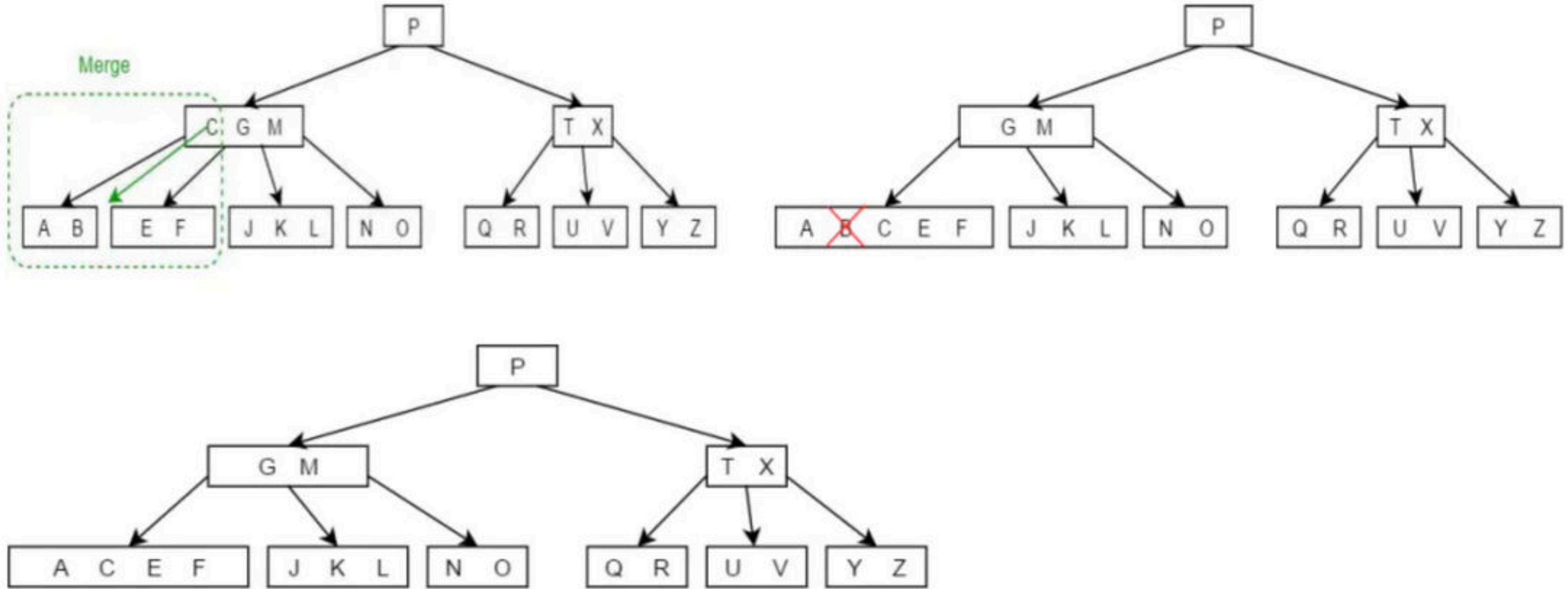
If $y.n == (t-1)$:

Merge x and y

Move down l to the new node as the median key

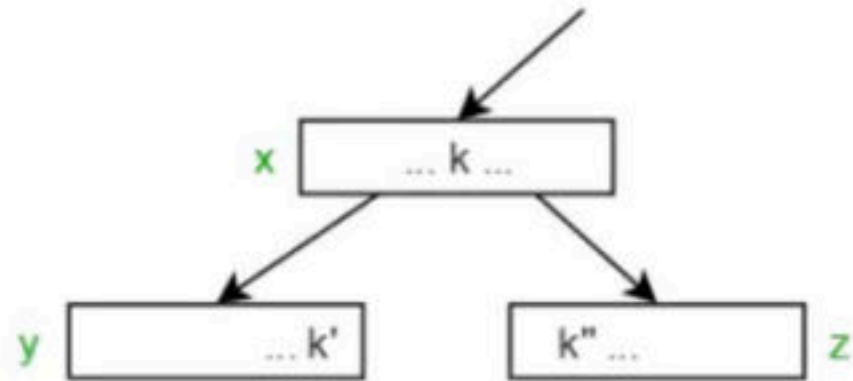
Delete k from the new node

Let's delete B from the B tree at the beginning.



Case 3

If k is in node x and x is an internal node (not a leaf)



Case 3a:

Find the child node y that precedes k (the node which is on the left side of k)

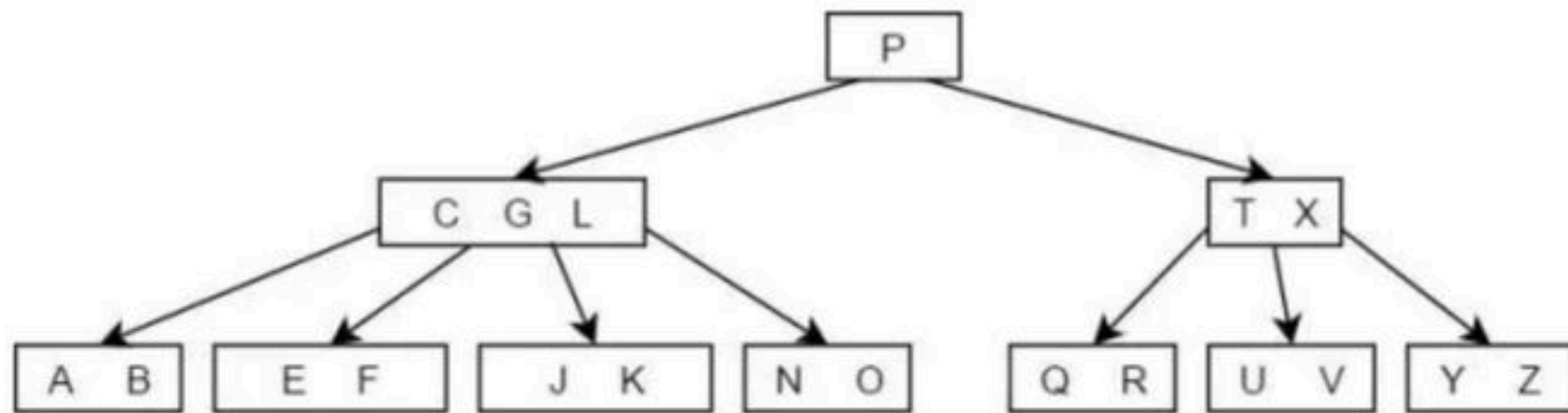
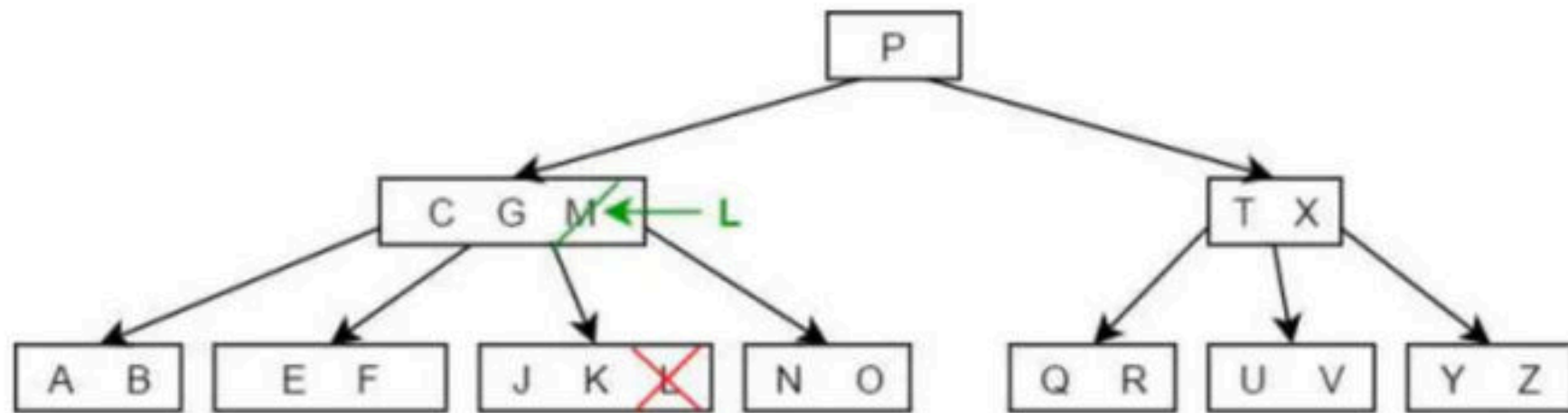
If $y.n \geq t$:

Find the key k' in y which is the predecessor of k

Delete k' recursively. (Here k' can be another internal node as well. So we have to delete it in the same way as well)

Replace k with k' in x

Let's delete M from the B tree at the beginning.



Case 3b:

Find the child node y that precedes k

If $y.n < t$ (or $y.n == (t-1)$):

Find the child node z that follows k (the node which is on the right side of k)

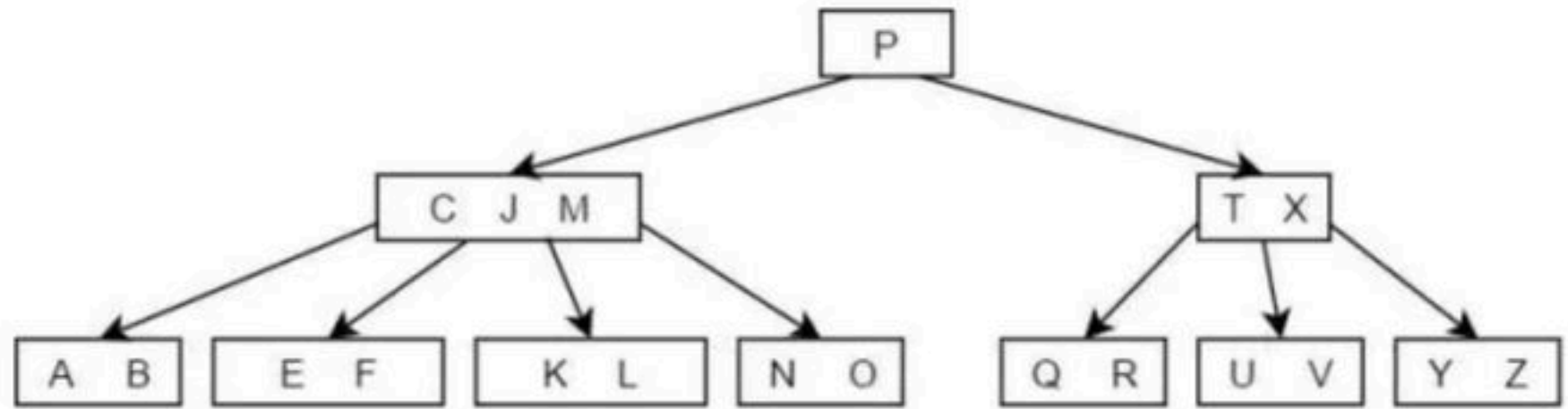
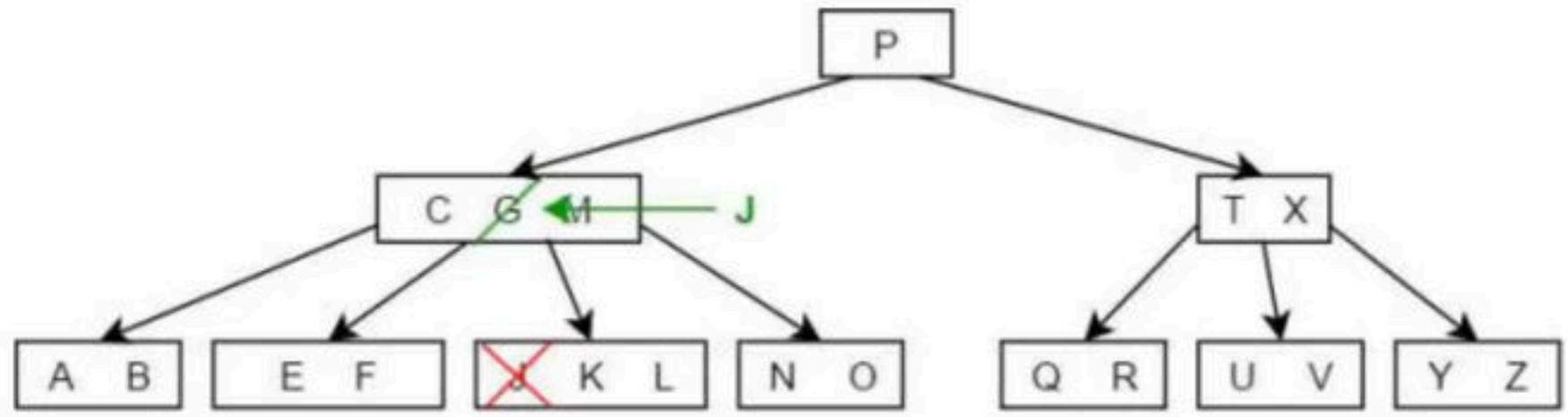
If $z.n \geq t$:

Find k'' in z which is the successor of k

Delete k'' recursively

Replace k with k'' in x

Let's delete G from the B tree at the beginning.



Case 3c:

Find the child node y that precedes k and the child node z that follows k

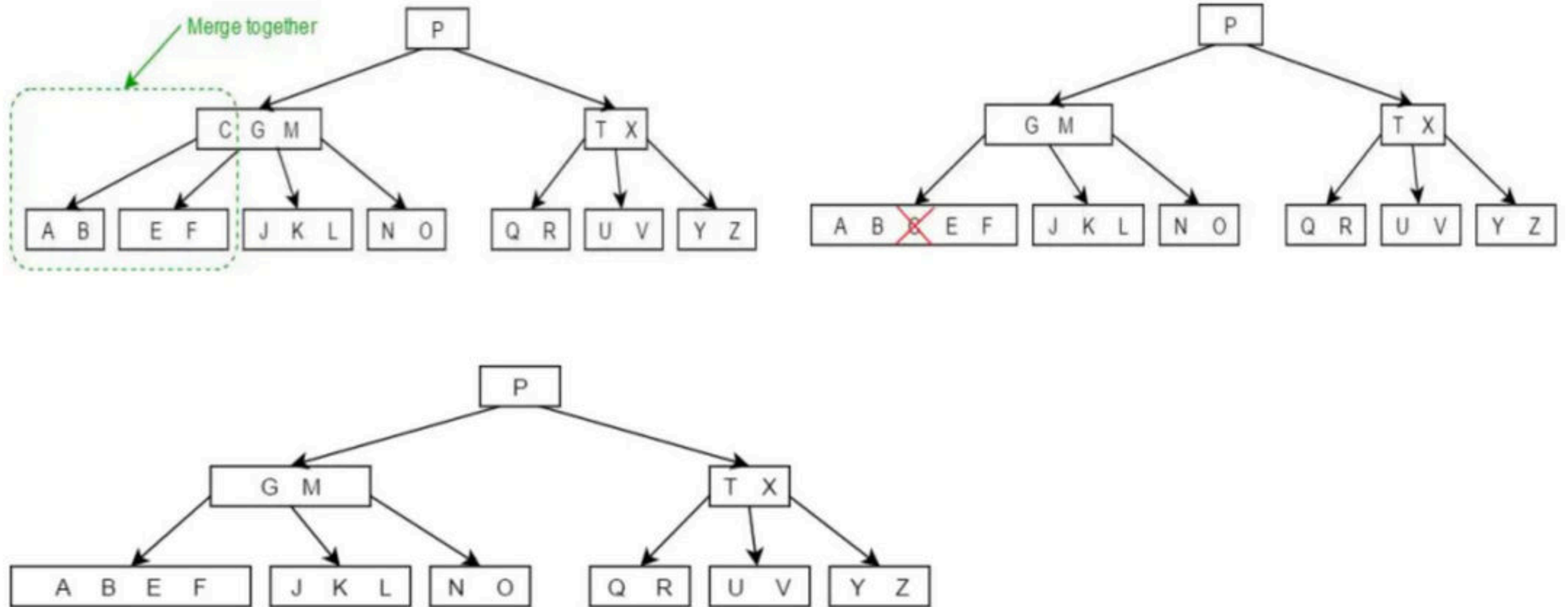
If $y.n == (t-1) \ \&\& \ z.n == (t-1)$:

Merge k and z to y

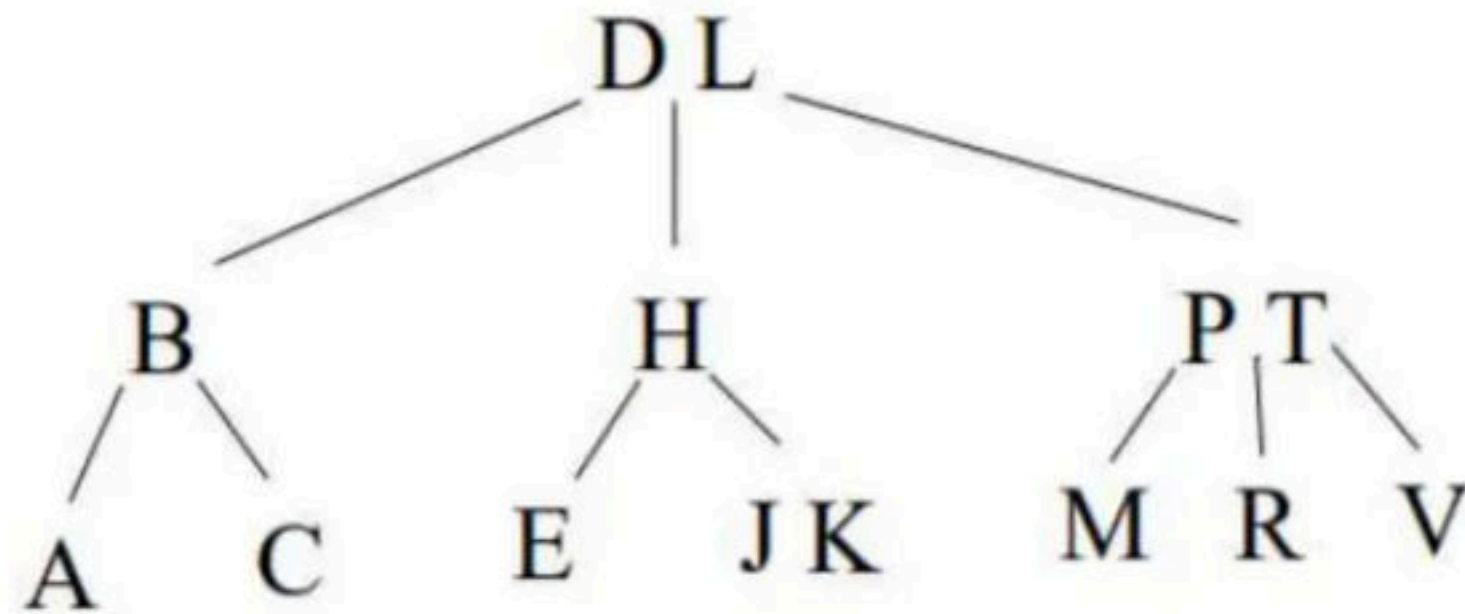
Free memory of node z

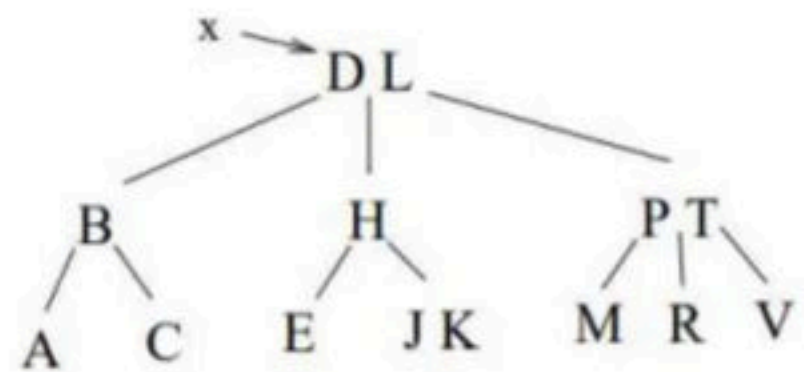
Recursively delete k from y

Let's delete C from the B tree at the beginning.

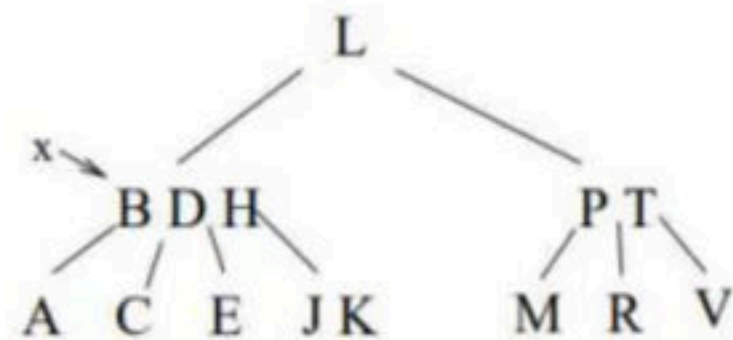


Show the B-tree the results when deleting A, then deleting V and then deleting P from the following B-tree with a minimum branching factor of $t = 2$.

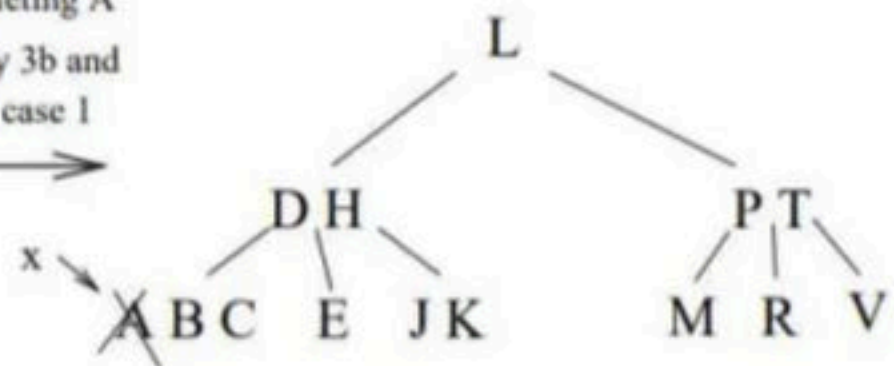




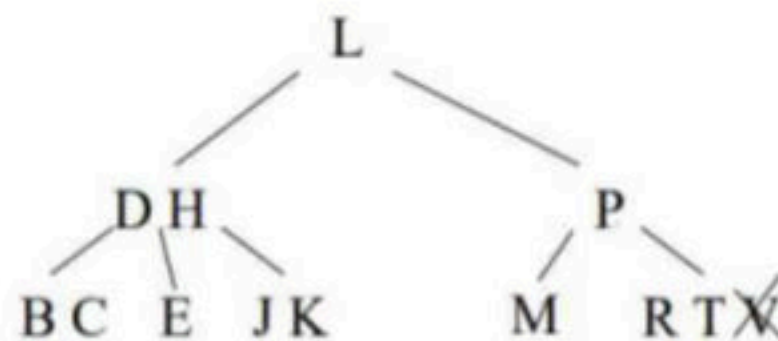
deleting A
apply case 3b



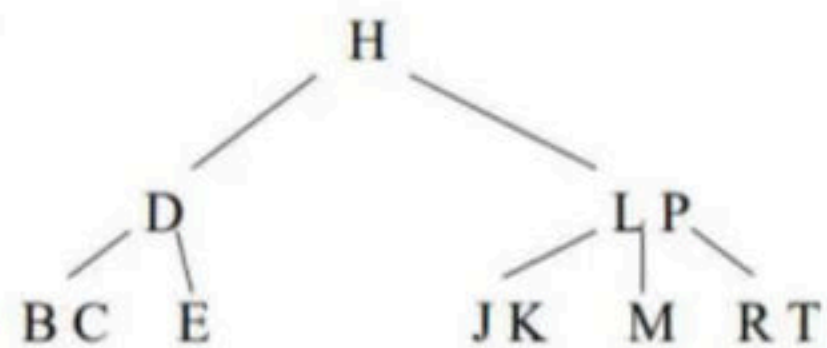
deleting A
apply 3b and
then case 1



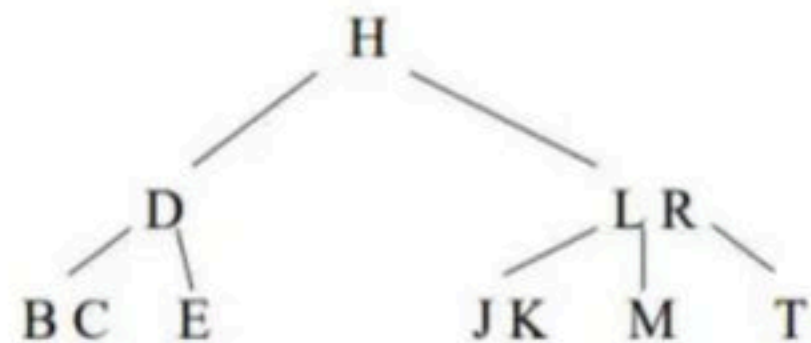
Now deleting V
go to PT then
apply case 3b and
case 1



Now deleting
P, apply 3a



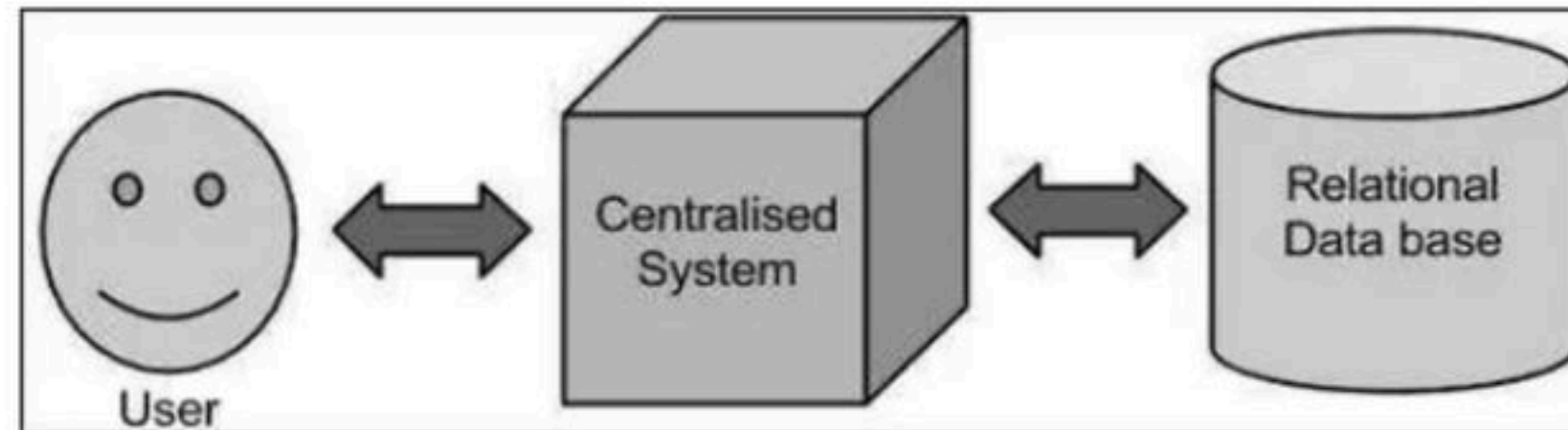
deleting P
case 2b



Advance DBMS

Traditional Approach

In this approach, an enterprise will have a computer to store and process big data. For storage purpose, the programmers will take the help of their choice of database vendors such as Oracle, IBM, etc. In this approach, the user interacts with the application, which in turn handles the part of data storage and analysis.

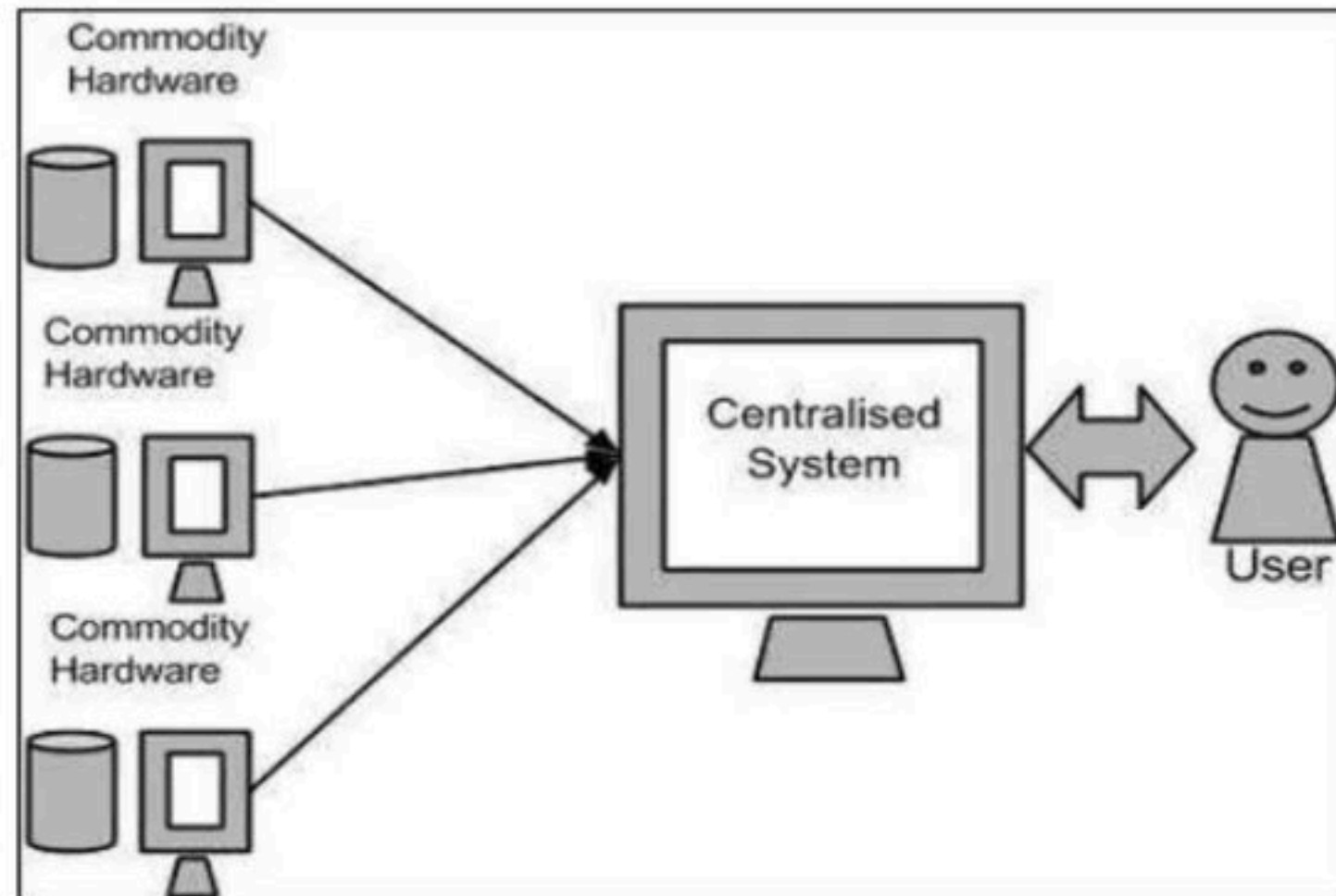


Limitation of Traditional Approach

This approach works fine with those applications that process less voluminous data that can be accommodated by standard database servers, or up to the limit of the processor that is processing the data. But when it comes to dealing with huge amounts of scalable data, it is a hectic task to process such data through a single database bottleneck.

Google's Solution

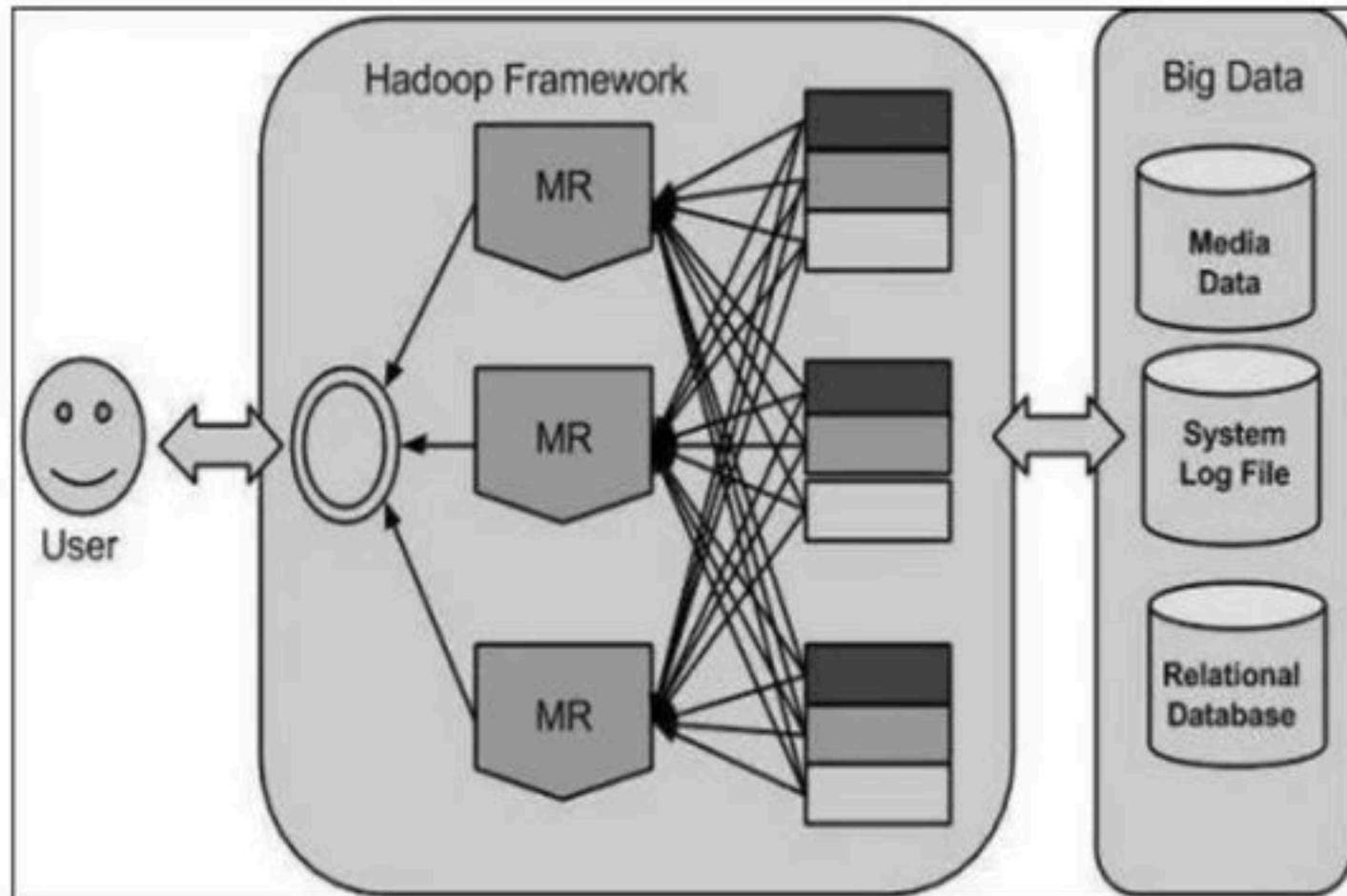
Google solved this problem using an algorithm called MapReduce. This algorithm divides the task into small parts and assigns them to many computers, and collects the results from them which when integrated, form the result dataset.



Hadoop

Using the solution provided by Google, **Doug Cutting** and his team developed an Open Source Project called **HADOOP**.

Hadoop runs applications using the **MapReduce algorithm**, where the data is processed in parallel with others. In short, Hadoop is used to develop applications that could perform complete statistical analysis on huge amounts of data.



What is Big Data?

Big data is a collection of large datasets that cannot be processed using traditional computing techniques. It is not a single technique or a tool, rather it has become a complete subject, which involves various tools, techniques and frameworks.

What Comes Under Big Data?

Big data involves the data produced by different devices and applications. Given below are some of the fields that come under the umbrella of Big Data.

- **Black Box Data** – It is a component of helicopter, airplanes, and jets, etc. It captures voices of the flight crew, recordings of microphones and earphones, and the performance information of the aircraft.
- **Social Media Data** – Social media such as Facebook and Twitter hold information and the views posted by millions of people across the globe.
- **Stock Exchange Data** – The stock exchange data holds information about the 'buy' and 'sell' decisions made on a share of different companies made by the customers.
- **Power Grid Data** – The power grid data holds information consumed by a particular node with respect to a base station.
- **Transport Data** – Transport data includes model, capacity, distance and availability of a vehicle.
- **Search Engine Data** – Search engines retrieve lots of data from different databases.

Thus Big Data includes huge volume, high velocity, and extensible variety of data. The data in it will be of three types.

- **Structured data** – Relational data.
- **Semi Structured data** – XML data.
- **Unstructured data** – Word, PDF, Text, Media Logs.

Benefits of Big Data

- Using the information kept in the social network like Facebook, the marketing agencies are learning about the response for their campaigns, promotions, and other advertising mediums.
- Using the information in the social media like preferences and product perception of their consumers, product companies and retail organizations are planning their production.
- Using the data regarding the previous medical history of patients, hospitals are providing better and quick service.

Big Data Technologies

Big data technologies are important in providing more accurate analysis, which may lead to more concrete decision-making resulting in greater operational efficiencies, cost reductions, and reduced risks for the business.

To harness the power of big data, you would require an infrastructure that can manage and process huge volumes of structured and unstructured data in real time and can protect data privacy and security.

There are various technologies in the market from different vendors including Amazon, IBM, Microsoft, etc., to handle big data. While looking into the technologies that handle big data, we examine the following two classes of technology –

Operational Big Data

This include systems like MongoDB that provide operational capabilities for real-time, interactive workloads where data is primarily captured and stored.

NoSQL Big Data systems are designed to take advantage of new cloud computing architectures that have emerged over the past decade to allow massive computations to be run inexpensively and efficiently. This makes operational big data workloads much easier to manage, cheaper, and faster to implement.

Some NoSQL systems can provide insights into patterns and trends based on real-time data with minimal coding and without the need for data scientists and additional infrastructure.

Analytical Big Data

These includes systems like Massively Parallel Processing (MPP) database systems and MapReduce that provide analytical capabilities for retrospective and complex analysis that may touch most or all of the data.

MapReduce provides a new method of analyzing data that is complementary to the capabilities provided by SQL, and a system based on MapReduce that can be scaled up from single servers to thousands of high and low end machines.

Big Data Challenges

The major challenges associated with big data are as follows –

- Capturing data
- Curation
- Storage
- Searching
- Sharing
- Transfer
- Analysis
- Presentation

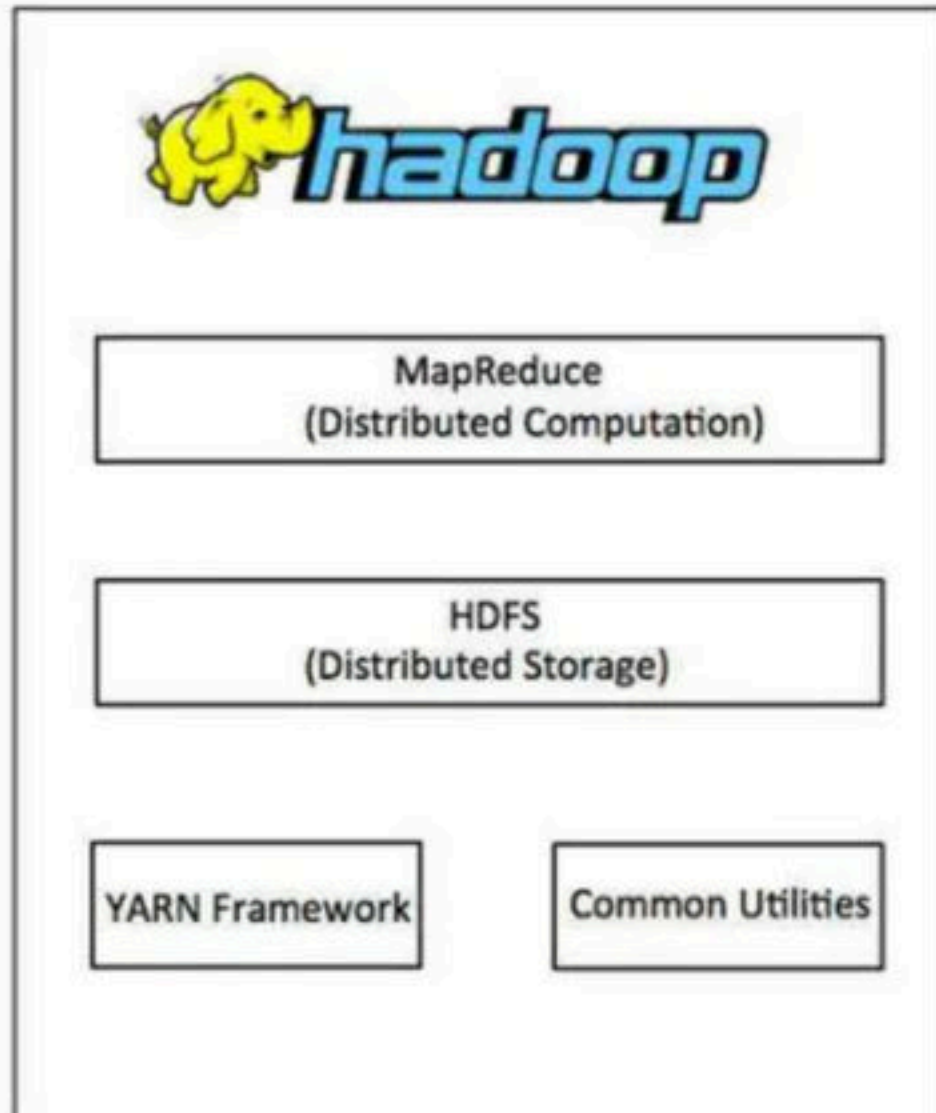
Hadoop is an Apache open source framework written in java that allows distributed processing of large datasets across clusters of computers using simple programming models.

The Hadoop framework application works in an environment that provides distributed *storage* and *computation* across clusters of computers. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

Hadoop Architecture

At its core, Hadoop has two major layers namely –

- Processing/Computation layer (MapReduce), and
- Storage layer (Hadoop Distributed File System).



MapReduce

MapReduce is a parallel programming model for writing distributed applications devised at Google for efficient processing of large amounts of data (multi-terabyte data-sets), on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner. The MapReduce program runs on Hadoop which is an Apache open-source framework.

Hadoop Distributed File System

The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS) and provides a distributed file system that is designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. It is highly fault-tolerant and is designed to be deployed on low-cost hardware. It provides high throughput access to application data and is suitable for applications having large datasets.

Apart from the above-mentioned two core components, Hadoop framework also includes the following two modules –

- **Hadoop Common** – These are Java libraries and utilities required by other Hadoop modules.
- **Hadoop YARN** – This is a framework for job scheduling and cluster resource management.

How Does Hadoop Work?

It is quite expensive to build bigger servers with heavy configurations that handle large scale processing, but as an alternative, you can tie together many commodity computers with single-CPU, as a single functional distributed system and practically, the clustered machines can read the dataset in parallel and provide a much higher throughput. Moreover, it is cheaper than one high-end server. So this is the first motivational factor behind using Hadoop that it runs across clustered and low-cost machines.

Hadoop runs code across a cluster of computers. This process includes the following core tasks that Hadoop performs –

- Data is initially divided into directories and files. Files are divided into uniform sized blocks of 128M and 64M (preferably 128M).
- These files are then distributed across various cluster nodes for further processing.
- HDFS, being on top of the local file system, supervises the processing.
- Blocks are replicated for handling hardware failure.
- Checking that the code was executed successfully.
- Performing the sort that takes place between the map and reduce stages.
- Sending the sorted data to a certain computer.
- Writing the debugging logs for each job.

Advantages of Hadoop

- Hadoop framework allows the user to quickly write and test distributed systems. It is efficient, and it automatic distributes the data and work across the machines and in turn, utilizes the underlying parallelism of the CPU cores.
- Hadoop does not rely on hardware to provide fault-tolerance and high availability (FTHA), rather Hadoop library itself has been designed to detect and handle failures at the application layer.
- Servers can be added or removed from the cluster dynamically and Hadoop continues to operate without interruption.
- Another big advantage of Hadoop is that apart from being open source, it is compatible on all the platforms since it is Java based.

Hadoop (a big data tool) works with number of related tools. Choose from the following, the common tools included into Hadoop:

- A. MySQL, Google API and Map reduce
- B. Map reduce, Scala and Hummer
- C. Map reduce, H Base and Hive
- D. Map reduce, Hummer and Heron

UGC NET 2019

Hadoop (a big data tool) works with number of related tools. Choose from the following, the common tools included into Hadoop:

- A. MySQL, Google API and Map reduce
- B. Map reduce, Scala and Hummer
- C. Map reduce, H Base and Hive
- D. Map reduce, Hummer and Heron

UGC NET 2019

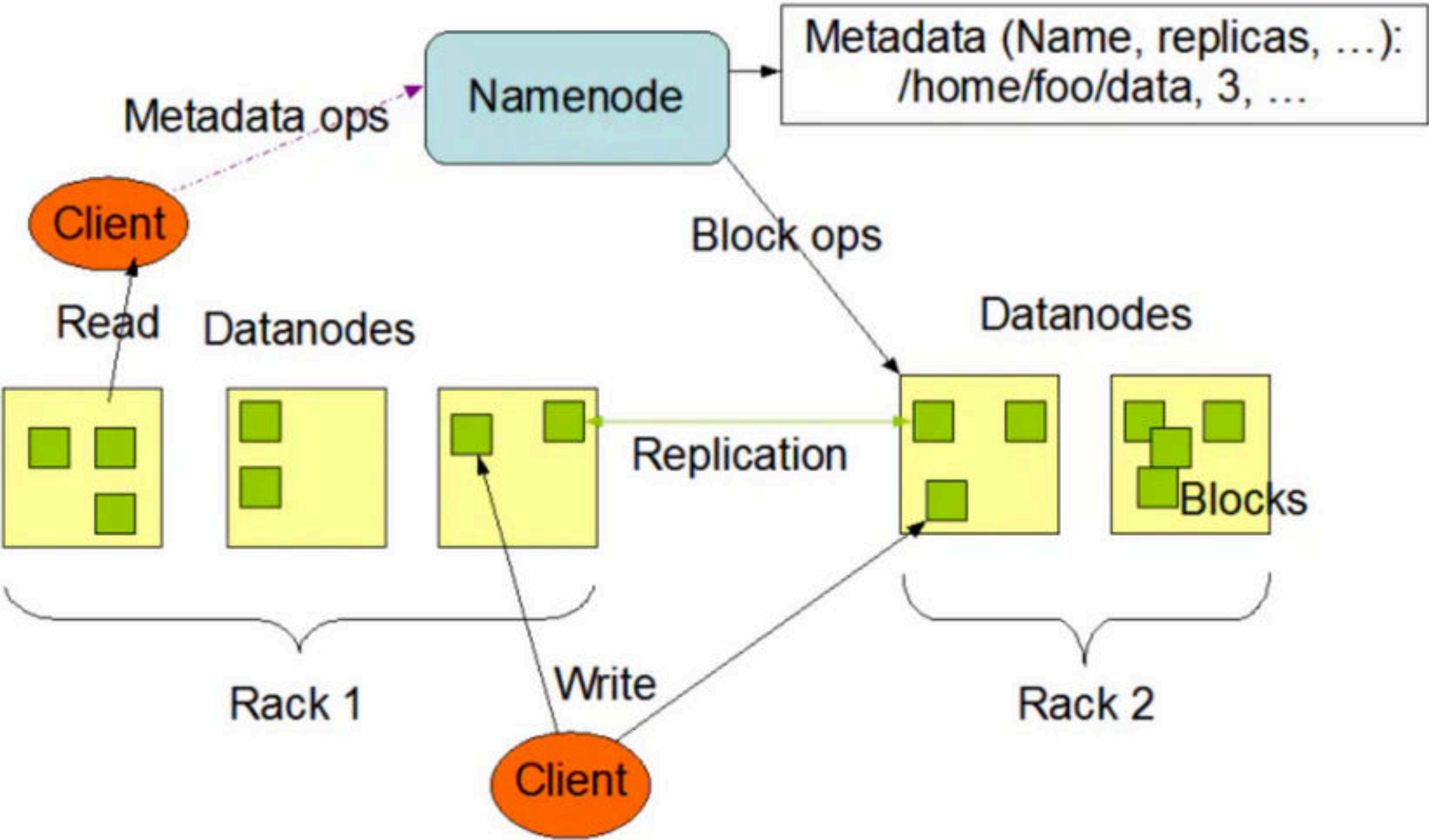
Hive and **HBase** are two different Hadoop based technologies . **Hive** is a SQL-like engine that runs MapReduce jobs, and **HBase** is a NoSQL key/value database on Hadoop. But just as Google can be used for search and Facebook for social networking, **Hive** can be used for analytical queries while **HBase** for real-time querying

The data node and name node in HADOOP are

- (1) Worker Node and Master Node respectively
- (2) Master Node and Worker Node respectively
- (3) Both Worker Nodes
- (4) Both Master Nodes

UGC NET 2020

HDFS Architecture



NameNode works as Master in [Hadoop](#) cluster. Below listed are the main function performed by NameNode:

1. Stores metadata of actual data. E.g. Filename, Path, No. of [Data Blocks](#), Block IDs, Block Location, No. of Replicas, Slave related configuration
2. Manages File system namespace.
3. Regulates client access request for actual file data file.
4. Assign work to Slaves(DataNode).
5. Executes file system name space operation like opening/closing files, renaming files and directories.
6. As Name node keep metadata in memory for fast retrieval, the huge amount of memory is required for its operation. This should be hosted on reliable hardware.

1. NameNode is the centerpiece of [HDFS](#).
2. NameNode is also known as the Master
NameNode only stores the metadata of HDFS – the directory tree of all files in the file system, and tracks the files across the cluster.
3. NameNode does not store the actual data or the dataset. The data itself is actually stored in the DataNodes.
4. NameNode knows the list of the [Blocks](#) and its location for any given file in HDFS. With this information NameNode knows how to construct the file from blocks.
5. NameNode is so critical to HDFS and when the NameNode is down, HDFS/[Hadoop cluster](#) is inaccessible and considered down.
6. NameNode is a single point of failure in Hadoop cluster.
7. NameNode is usually configured with a lot of memory (RAM). Because the block locations are held in main memory

DataNode works as Slave in [Hadoop cluster](#) . Below listed are the main function performed by DataNode:

1. Actually stores Business data.
2. This is actual worker node where Read/Write/Data processing is handled.
3. Upon instruction from Master, it performs creation/replication/deletion of data blocks.
4. As all the Business data is stored on DataNode, the huge amount of storage is required for its operation. Commodity hardware can be used for hosting DataNode.

1. DataNode is responsible for storing the actual data in HDFS.
2. DataNode is also known as the Slave
3. NameNode and DataNode are in constant communication.
4. When a DataNode starts up it announce itself to the NameNode along with the list of blocks it is responsible for.
5. When a DataNode is down, it does not affect the availability of data or the cluster. NameNode will arrange for replication for the blocks managed by the DataNode that is not available.
6. DataNode is usually configured with a lot of hard disk space. Because the actual data is stored in the DataNode.

DataNode periodically send HEARTBEATS to NameNode

The data node and name node in HADOOP are

- (1) ✓ Worker Node and Master Node respectively
- (2) Master Node and Worker Node respectively
- (3) Both Worker Nodes
- (4) Both Master Nodes

UGC NET 2020

Correct Ans: Option 1

What is speculative execution in MapReduce/Hadoop?

- A. Speculatively allocate more nodes to handle future MapReduce tasks
- B. Certain number of duplicate tasks are launched on same slave node to handle the fault-tolerance issue
- C. Speculatively terminate some nodes in MapReduce cluster for conservation of energy
- D. Certain number of duplicate tasks are launched on different slave nodes to handle the fault-tolerance issue

What is Speculative Execution?

Sometimes you will notice that a Job which has 3 input splits executed 4 mappers and killed the 4th mapper. The job would still complete successfully but ever wondered why did it execute the 4th Mapper when there are only 3 input splits and kill it at the end ?

What you see is called Speculative Execution.

When Hadoop framework feels that a certain task (Mapper or Reducer) is taking longer on average compared to the other tasks from the same job, it clones the “long running” task and run it on another node. This is called Speculative Execution.

Meaning Hadoop is speculating that something is wrong with the “long running” task and runs a clone task on the other node. The slowness in the “long running” job could be due to a faulty hardware, network congestion, or the node could be simply busy etc. Most of the the time this is a false alarm and the task which was considered long running or problematic completes successfully. In that case Hadoop will kill the cloned task and proceed with the results from the completed task.

What is speculative execution in MapReduce/Hadoop?

- A. Speculatively allocate more nodes to handle future MapReduce tasks
- B. Certain number of duplicate tasks are launched on same slave node to handle the fault-tolerance issue
- C. Speculatively terminate some nodes in MapReduce cluster for conservation of energy
- D. Certain number of duplicate tasks are launched on different slave nodes to handle the fault-tolerance issue

What is Data Warehousing?

Data warehousing is the process of constructing and using a data warehouse. A data warehouse is constructed by integrating data from multiple heterogeneous sources that support analytical reporting, structured and/or ad hoc queries, and decision making. Data warehousing involves data cleaning, data integration, and data consolidations.

On a technical level, a **data warehouse** periodically pulls data from those apps and systems; then, the data goes through formatting and import processes to match the data already in the warehouse. **The data warehouse stores this processed data so it's ready for decision makers to access.** How frequently data pulls occur, or how data is formatted, etc., will vary depending on the needs of the organization.

benefits of a data warehouse

Organizations that use a **data warehouse to assist their analytics and business intelligence**:

- **Better data** — Adding **data sources** to a data warehouse enables organizations to ensure that they are collecting consistent and relevant data from that source. They don't need to wonder whether the data will be accessible or inconsistent as it comes in to the system. This ensures higher **data quality** and **data integrity** for sound decision making.
- **Faster decisions** — Data in a warehouse is in such consistent formats that it is ready to be analyzed. It also provides the analytical power and a more complete dataset to base decisions on hard facts. Therefore, decision makers no longer need to rely on hunches, incomplete data, or poor quality data and risk delivering slow and inaccurate results.

	Database	Data Warehouse
What it is	Data collected for multiple transactional purposes. Optimized for read/write access.	Aggregated transactional data , transformed and stored for analytical purposes. Optimized for aggregation and retrieval of large data sets.
How it's used	Databases are made to quickly record and retrieve information.	Data warehouses store data from multiple databases, which makes it easier to analyze.
Types	Databases are used in data warehousing. However, the term usually refers to an online, transactional processing database. There are other types as well, including csv, html, and Excel spreadsheets used for database purposes.	A data warehouse is an analytical database that layers on top of transactional databases to allow for analytics.

The future of the data warehouse: move to the cloud

As businesses make the [move to the cloud](#), so too do their databases and data warehousing tools.

The cloud offers many advantages: flexibility, collaboration, and accessibility from anywhere, to name a few. Popular tools like [Amazon Redshift](#), [Microsoft Azure SQL Data Warehouse](#), [Snowflake](#), [Google BigQuery](#), and [Databricks](#) have all offered businesses simple ways to warehouse and analyze their cloud data.

Using Data Warehouse Information

There are decision support technologies that help utilize the data available in a data warehouse. These technologies help executives to use the warehouse quickly and effectively. They can gather data, analyze it, and take decisions based on the information present in the warehouse. The information gathered in a warehouse can be used in any of the following domains –

- **Tuning Production Strategies** – The product strategies can be well tuned by repositioning the products and managing the product portfolios by comparing the sales quarterly or yearly.
- **Customer Analysis** – Customer analysis is done by analyzing the customer's buying preferences, buying time, budget cycles, etc.
- **Operations Analysis** – Data warehousing also helps in customer relationship management, and making environmental corrections. The information also allows us to analyze business operations.

Integrating Heterogeneous Databases

To integrate heterogeneous databases, there are two approaches –

- Query-driven Approach
- Update-driven Approach

Query-Driven Approach

This is the traditional approach to integrate heterogeneous databases. This approach was used to build wrappers and integrators on top of multiple heterogeneous databases. These integrators are also known as mediators.

Process of Query-Driven Approach

- When a query is issued to a client side, a metadata dictionary translates the query into an appropriate form for individual heterogeneous sites involved.
- Now these queries are mapped and sent to the local query processor.
- The results from heterogeneous sites are integrated into a global answer set.
- Disadvantages
- Query-driven approach needs complex integration and filtering processes.
- This approach is very inefficient.
- It is very expensive for frequent queries.
- This approach is also very expensive for queries that require aggregations.

Update-Driven Approach

This is an alternative to the traditional approach. Today's data warehouse systems follow update-driven approach rather than the traditional approach discussed earlier. In update-driven approach, the information from multiple heterogeneous sources are integrated in advance and are stored in a warehouse. This information is available for direct querying and analysis.

Advantages

This approach has the following advantages –

- This approach provide high performance.
- The data is copied, processed, integrated, annotated, summarized and restructured in semantic data store in advance.
- Query processing does not require an interface to process data at local sources.

Functions of Data Warehouse Tools and Utilities

The following are the functions of data warehouse tools and utilities –

- **Data Extraction** – Involves gathering data from multiple heterogeneous sources.
- **Data Cleaning** – Involves finding and correcting the errors in data.
- **Data Transformation** – Involves converting the data from legacy format to warehouse format.
- **Data Loading** – Involves sorting, summarizing, consolidating, checking integrity, and building indices and partitions.
- **Refreshing** – Involves updating from data sources to warehouse.

Commonly used terms in data warehousing

Metadata

Metadata is simply defined as data about data. The data that are used to represent other data is known as metadata. For example, the index of a book serves as a metadata for the contents in the book. In other words, we can say that metadata is the summarized data that leads us to the detailed data.

In terms of data warehouse, we can define metadata as following –

- Metadata is a road-map to data warehouse.
- Metadata in data warehouse defines the warehouse objects.
- Metadata acts as a directory. This directory helps the decision support system to locate the contents of a data warehouse.

Metadata Repository

Metadata repository is an integral part of a data warehouse system. It contains the following metadata –

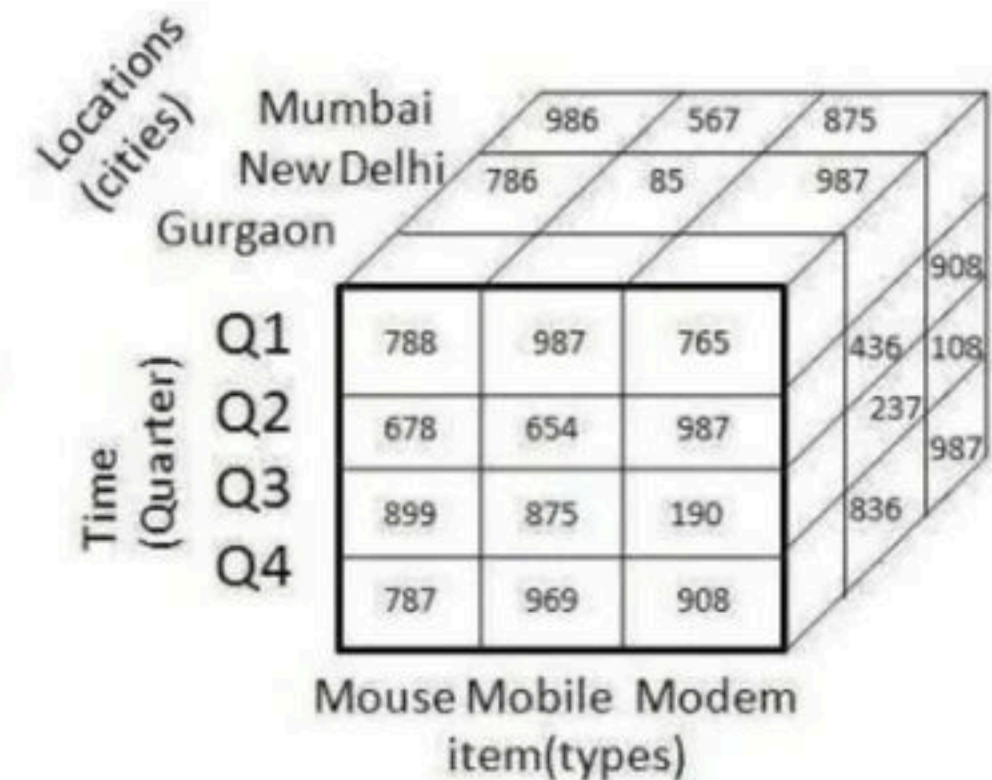
- **Business metadata** – It contains the data ownership information, business definition, and changing policies.
- **Operational metadata** – It includes currency of data and data lineage. Currency of data refers to the data being active, archived, or purged. Lineage of data means history of data migrated and transformation applied on it.
- **Data for mapping from operational environment to data warehouse** – It metadata includes source databases and their contents, data extraction, data partition, cleaning, transformation rules, data refresh and purging rules.
- **The algorithms for summarization** – It includes dimension algorithms, data on granularity, aggregation, summarizing, etc.

Data Cube

A data cube helps us represent data in multiple dimensions. It is defined by dimensions and facts. The dimensions are the entities with respect to which an enterprise preserves the records.

The 3-D view of the sales data with respect to time, item, and location is shown in the table below –

Time	Location="Gurgaon"			Location="New Delhi"			Location="Mumbai"		
	Item			Item			Item		
	Mouse	Mobile	Modem	Mouse	Mobile	Modem	Mouse	Mobile	Modem
Q1	788	987	765	786	85	987	986	567	875
Q2	678	654	987	659	786	436	980	876	908
Q3	899	875	190	983	909	237	987	100	1089
Q4	787	969	908	537	567	836	837	926	987



3-D table can be represented as 3-D data cube

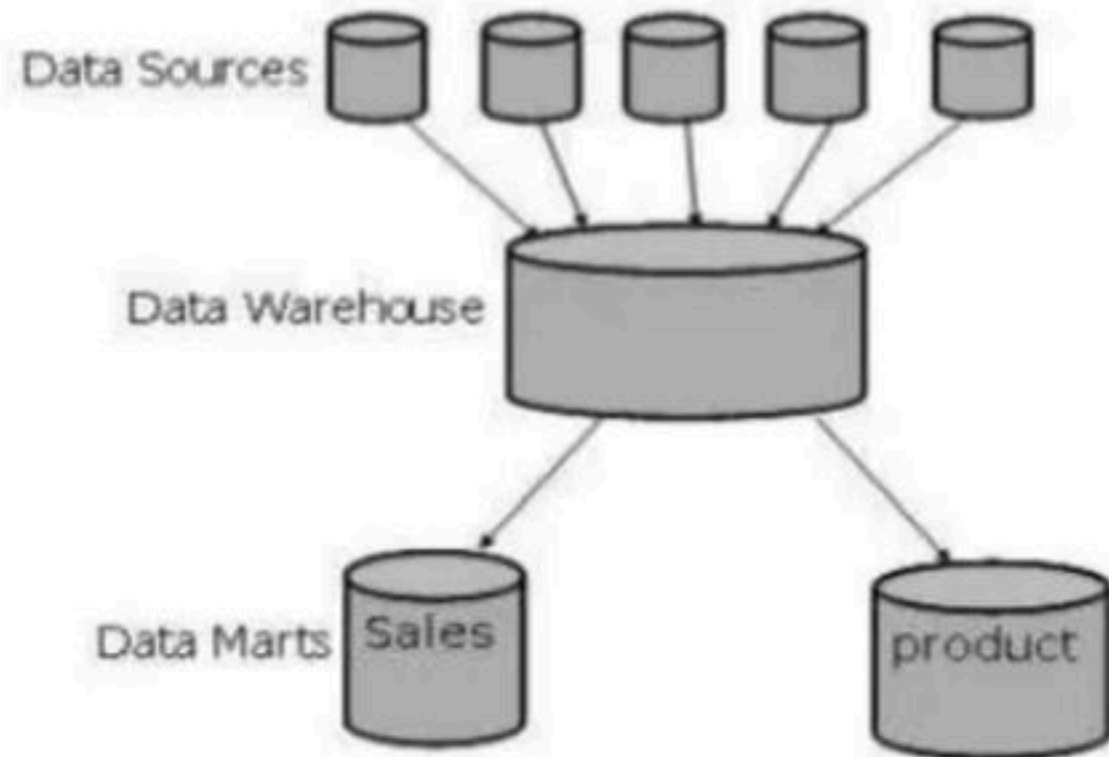
Data Mart

Data marts contain a subset of organization-wide data that is valuable to specific groups of people in an organization. In other words, a data mart contains only those data that is specific to a particular group. For example, the marketing data mart may contain only data related to items, customers, and sales. Data marts are confined to subjects.

Points to Remember About Data Marts

- Windows-based or Unix/Linux-based servers are used to implement data marts. They are implemented on low-cost servers.
- The implementation cycle of a data mart is measured in short periods of time, i.e., in weeks rather than months or years.
- The life cycle of data marts may be complex in the long run, if their planning and design are not organization-wide.
- Data marts are small in size.
- Data marts are customized by department.
- The source of a data mart is departmentally structured data warehouse.
- Data marts are flexible.

Difference between Data Warehouse and Data mart



Data warehouses are also sometimes confused with [data marts](#). But data warehouses are generally much bigger and contain a greater variety of data, while data marts are limited in their application.

Data marts are often subsets of a warehouse, designed to easily deliver specific data to a specific user, for a specific application. In the simplest terms, data marts can be thought of as single-subject, while data warehouses cover multiple subjects.

Virtual Warehouse

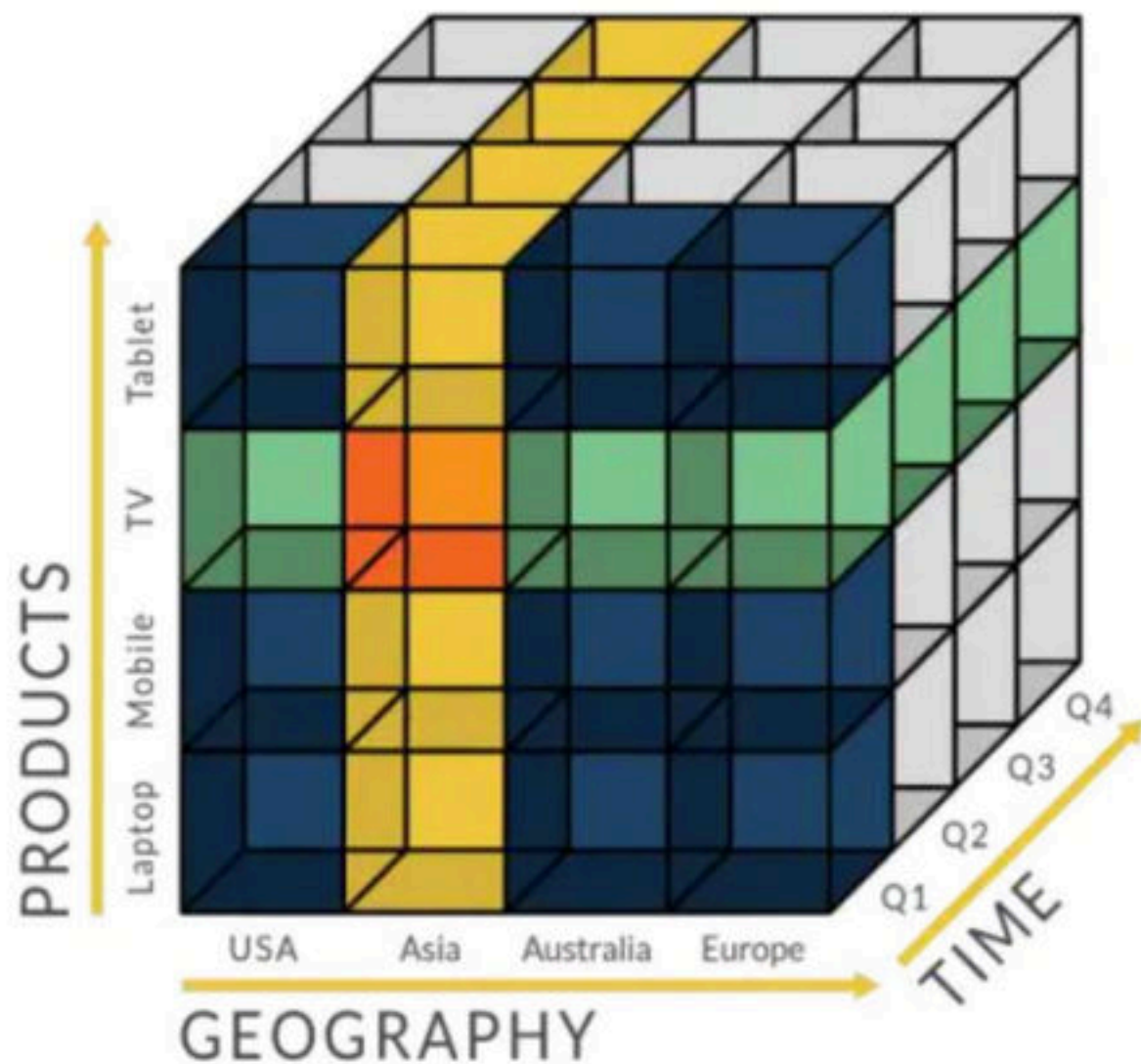
The view over an operational data warehouse is known as virtual warehouse. It is easy to build a virtual warehouse. Building a virtual warehouse requires excess capacity on operational database servers.

OLAP

Online Analytical Processing Server (OLAP) is based on the multidimensional data model. It allows managers, and analysts to get an insight of the information through fast, consistent, and interactive access to information.

OLAP (Online Analytical Processing) is the technology behind many Business Intelligence (BI) applications. OLAP is a powerful technology for data discovery, including capabilities for limitless report viewing, complex analytical calculations, and predictive “what if” scenario (budget, forecast) planning

OLAP technology has been defined as the ability to achieve “fast access to shared multidimensional information.” Given OLAP technology’s ability to create very fast aggregations and calculations of underlying data sets, one can understand its usefulness in helping business leaders make better, quicker “informed” decisions.



OLAP is an acronym for **Online Analytical Processing**. OLAP performs multidimensional analysis of business data and provides the capability for complex calculations, trend analysis, and sophisticated data modeling. It is the foundation for many kinds of business applications for Business Performance Management, Planning, Budgeting, Forecasting, Financial Reporting, Analysis, Simulation Models, Knowledge Discovery, and Data Warehouse Reporting. OLAP enables end-users to perform ad hoc analysis of data in multiple dimensions, thereby providing the insight and understanding they need for better decision making.

What is OLTP?

OLTP (Online Transactional Processing) is a category of data processing that is focused on transaction-oriented tasks. OLTP typically involves inserting, updating, and/or deleting small amounts of data in a database.

OLTP mainly deals with large numbers of transactions by a large number of users.

Examples of OLTP Transactions

Examples of OLTP transactions include:

- Online banking
- Purchasing a book online
- Booking an airline ticket
- Sending a text message
- Order entry
- Telemarketers entering telephone survey results
- Call center staff viewing and updating customers' details

Characteristics of OLTP

OLTP transactions are usually very specific in the task that they perform, and they usually involve a single record or a small selection of records.

For example, an online banking customer might send money from his account to his wife's account. In this case, the transaction only involves two accounts – his account and his wife's. It does not involve the other bank customers.

This is in contrast to [Online Analytical Processing](#) (OLAP), which usually involves querying many records (even all records) in a database for analytical purposes. An OLAP banking example could be a bank manager performing a query across all customer accounts, so that he can see which suburbs had the most active online banking customers during a certain period. OLAP is often used to provide analytics on data that was captured via an OLTP application. So, while OLTP and OLAP often work with the same data sets, they have different characteristics.

OLTP applications typically possess the following characteristics:

- Transactions that involve small amounts of data
- Indexed access to data
- A large number of users
- Frequent queries and updates
- Fast response times

OLAP vs OLTP

	Data Warehouse (OLAP)	Operational Database (OLTP)
1	Involves historical processing of information.	Involves day-to-day processing.
2	OLAP systems are used by knowledge workers such as executives, managers and analysts.	OLTP systems are used by clerks, DBAs, or database professionals.
3	Useful in analyzing the business.	Useful in running the business.
4	It focuses on Information out.	It focuses on Data in.
5	Based on Star Schema, Snowflake, Schema and Fact Constellation Schema.	Based on Entity Relationship Model.
6	Contains historical data.	Contains current data.
7	Provides summarized and consolidated data.	Provides primitive and highly detailed data.
8	Provides summarized and multidimensional view of data.	Provides detailed and flat relational view of data.
9	Number of users is in hundreds.	Number of users is in thousands.
10	Number of records accessed is in millions.	Number of records accessed is in tens.
11	Database size is from 100 GB to 1 TB	Database size is from 100 MB to 1 GB.
12	Highly flexible.	Provides high performance.

Data scrubbing is

A. A process to reject data from the data warehouse and to create the necessary indexes.

B. A process to load the data in the data warehouse and to create the necessary indexes.

C. A process to upgrade the quality of data after it is moved into a data warehouse

D. A process to upgrade the quality of data before it is moved into a data warehouse

Data scrubbing is an **error correction** technique that uses a background task to periodically inspect **main memory** or **storage** for errors, then corrects detected errors using **redundant data** in the form of different **checksums** or copies of data. Data scrubbing reduces the likelihood that single correctable errors will accumulate, leading to reduced risks of uncorrectable errors.

Data integrity is a high-priority concern in writing, reading, storage, transmission, or processing of the **computer data** in computer **operating systems** and in computer storage and **data transmission** systems. However, only a few of the currently existing and used **file systems** provide sufficient protection against **data corruption**.

To address this issue, data scrubbing provides routine checks of all **inconsistencies** in data and, in general, prevention of hardware or software failure. This "scrubbing" feature occurs commonly in memory, disk arrays, **file systems**, or **FPGAs** as a mechanism of error detection and correction.

Data scrubbing is

A. A process to reject data from the data warehouse and to create the necessary indexes.

B. A process to load the data in the data warehouse and to create the necessary indexes.

C. A process to upgrade the quality of data after it is moved into a data warehouse

D. A process to upgrade the quality of data before it is moved into a data warehouse

What Is Data Mining?

Data mining is a process used by companies to turn raw data into useful information. By using software to look for patterns in large batches of data, businesses can learn more about their customers to develop more effective marketing strategies, increase sales and decrease costs. Data mining depends on [effective data collection](#), [warehousing](#), and computer processing.

How Data Mining Works

[Data mining](#) involves exploring and analyzing large blocks of information to glean meaningful patterns and trends. It can be used in a variety of ways, such as database marketing, credit risk management, [fraud detection](#), spam Email filtering, or even to discern the sentiment or opinion of users.

The data mining process breaks down into five steps. First, organizations collect data and load it into their data warehouses. Next, they store and manage the data, either on in-house servers or the cloud. Business analysts, management teams and information technology professionals access the data and determine how they want to organize it. Then, application software sorts the data based on the user's results, and finally, the end-user presents the data in an easy-to-share format, such as a graph or table.

- Data mining is the process of analyzing a large batch of information to discern trends and patterns.
- Data mining can be used by corporations for everything from learning about what customers are interested in or want to buy to fraud detection and spam filtering.
- Data mining programs break down patterns and connections in data based on what information users request or provide.

Example of Data Mining

Grocery stores are well-known users of data mining techniques. Many supermarkets offer free [loyalty cards](#) to customers that give them access to reduced prices not available to non-members. The cards make it easy for stores to track who is buying what, when they are buying it and at what price. After analyzing the data, stores can then use this data to offer customers coupons targeted to their buying habits and decide when to put items on sale or when to sell them at full price.

Data mining can be a cause for concern when a company uses only selected information, which is not representative of the overall sample group, to prove a certain hypothesis.

IBM and _____ have announced a major initiative to use Hadoop to support university courses in distributed computer programming.

- a) Google Latitude
- b) Android (operating system)
- c) Google Variations
- d) Google

IBM and _____ have announced a major initiative to use Hadoop to support university courses in distributed computer programming.

- a) Google Latitude
- b) Android (operating system)
- c) Google Variations
- d) Google

Point out the correct statement.

- a) Hadoop is an ideal environment for extracting and transforming small volumes of data
- b) Hadoop stores data in HDFS and supports data compression/decompression
- c) The Giraph framework is less useful than a MapReduce job to solve graph and machine learning
- d) None of the mentioned

Point out the correct statement.

a) Hadoop is an ideal environment for extracting and transforming small volumes of data

b) Hadoop stores data in HDFS and supports data compression/decompression

c) The Giraph framework is less useful than a MapReduce job to solve graph and machine learning

d) None of the mentioned

Data compression can be achieved using compression algorithms like bzip2, gzip, LZO, etc. Different algorithms can be used in different scenarios based on their capabilities.

What license is Hadoop distributed under?

- a) Apache License 2.0
- b) Mozilla Public License
- c) Shareware
- d) Commercial

What license is Hadoop distributed under?

- a) Apache License 2.0
- b) Mozilla Public License
- c) Shareware
- d) Commercial

Sun also has the Hadoop Live CD _____ project, which allows running a fully functional Hadoop cluster using a live CD.

- a) OpenOffice.org
- b) OpenSolaris
- c) GNU
- d) Linux

Sun also has the Hadoop Live CD _____ project, which allows running a fully functional Hadoop cluster using a live CD.

- a) OpenOffice.org
- b) OpenSolaris
- c) GNU
- d) Linux

The OpenSolaris Hadoop LiveCD project built a bootable CD-ROM image.

What was Hadoop written in?

- a) Java (software platform)
- b) Perl
- c) Java (programming language)
- d) Lua (programming language)

What was Hadoop written in?

- a) Java (software platform)
- b) Perl
- c) Java (programming language)
- d) Lua (programming language)

The Hadoop framework itself is mostly written in the Java programming language, with some native code in C and command-line utilities written as shell-scripts.

Which of the following platforms does Hadoop run on?

- a) Bare metal
- b) Debian
- c) Cross-platform
- d) Unix-like

Which of the following platforms does Hadoop run on?

a) Bare metal

b) Debian

c) Cross-platform

d) Unix-like

Hadoop has support for cross-platform operating system.

Hadoop achieves reliability by replicating the data across multiple hosts and hence does not require _____ storage on hosts.

- a) RAID
- b) Standard RAID levels
- c) ZFS
- d) Operating system

Hadoop achieves reliability by replicating the data across multiple hosts and hence does not require _____ storage on hosts.

- a) RAID
- b) Standard RAID levels
- c) ZFS
- d) Operating system

RAID is a data storage virtualization technology that combines multiple physical disk drive components into one or more logical units for the purposes of data

As companies move past the experimental phase with Hadoop, many cite the need for additional capabilities, including _____

- a) Improved data storage and information retrieval
- b) Improved extract, transform and load features for data integration
- c) Improved data warehousing functionality
- d) Improved security, workload management, and SQL support

As companies move past the experimental phase with Hadoop, many cite the need for additional capabilities, including _____

- a) Improved data storage and information retrieval
- b) Improved extract, transform and load features for data integration
- c) Improved data warehousing functionality
- d) Improved security, workload management, and SQL support

Adding security to Hadoop is challenging because all the interactions do not follow the classic client-server pattern.

Point out the correct statement.

- a) Hadoop do need specialized hardware to process the data
- b) Hadoop 2.0 allows live stream processing of real-time data
- c) In Hadoop programming framework output files are divided into lines or records
- d) None of the mentioned

Point out the correct statement.

- a) Hadoop do need specialized hardware to process the data
- b) Hadoop 2.0 allows live stream processing of real-time data
- c) In Hadoop programming framework output files are divided into lines or records
- d) None of the mentioned

Hadoop batch processes data distributed over a number of computers ranging in 100s and 1000s.

According to analysts, for what can traditional IT systems provide a foundation when they're integrated with big data technologies like Hadoop?

- a) Big data management and data mining
- b) Data warehousing and business intelligence
- c) Management of Hadoop clusters
- d) Collecting and storing unstructured data

According to analysts, for what can traditional IT systems provide a foundation when they're integrated with big data technologies like Hadoop?

- a) Big data management and data mining
- b) Data warehousing and business intelligence
- c) Management of Hadoop clusters
- d) Collecting and storing unstructured data

Data warehousing integrated with Hadoop would give a better understanding of data.

Hadoop is a framework that works with a variety of related tools. Common cohorts include _____

- a) MapReduce, Hive and HBase
- b) MapReduce, MySQL and Google Apps
- c) MapReduce, Hummer and Iguana
- d) MapReduce, Heron and Trumpet

Hadoop is a framework that works with a variety of related tools. Common cohorts include _____

a) MapReduce, Hive and HBase

b) MapReduce, MySQL and Google Apps

c) MapReduce, Hummer and Iguana

d) MapReduce, Heron and Trumpet

To use Hive with HBase you'll typically want to launch two clusters, one to run HBase and the other to run Hive.

Point out the wrong statement.

- a) Hardtop processing capabilities are huge and its real advantage lies in the ability to process terabytes & petabytes of data
- b) Hadoop uses a programming model called "MapReduce", all the programs should confirm to this model in order to work on Hadoop platform
- c) The programming model, MapReduce, used by Hadoop is difficult to write and test
- d) All of the mentioned

Point out the wrong statement.

- a) Hardtop processing capabilities are huge and its real advantage lies in the ability to process terabytes & petabytes of data
 - b) Hadoop uses a programming model called "MapReduce", all the programs should confirm to this model in order to work on Hadoop platform
 - c) The programming model, MapReduce, used by Hadoop is difficult to write and test
 - d) All of the mentioned
- The programming model, MapReduce, used by Hadoop is simple to write and test.

All of the following accurately describe Hadoop, EXCEPT

- a) Open-source
- b) Real-time
- c) Java-based
- d) Distributed computing approach

All of the following accurately describe Hadoop, EXCEPT

- a) Open-source
- b) Real-time
- c) Java-based
- d) Distributed computing approach

Apache Hadoop is an open-source software framework for distributed storage and distributed processing of Big Data on clusters of commodity hardware.

_____ has the world's largest Hadoop cluster.

- a) Apple
- b) Datamatics
- c) Facebook
- d) None of the mentioned

_____ has the world's largest Hadoop cluster.

a) Apple

b) Datamatics

c) Facebook

d) None of the mentioned

Facebook Tackles Big Data With _____ based on Hadoop.

- a) 'Project Prism'
- b) 'Prism'
- c) 'Project Big'
- d) 'Project Data'

Facebook Tackles Big Data With _____ based on Hadoop.

a) 'Project Prism'

b) 'Prism'

c) 'Project Big'

d) 'Project Data'

Prism automatically replicates and moves data wherever it's needed across a vast network of computing facilities.

1. **Data scrubbing is which of the following?**

- A. **A process to reject data from the data warehouse and to create the necessary indexes**
- B. **A process to load the data in the data warehouse and to create the necessary indexes**
- C. **A process to upgrade the quality of data after it is moved into a data warehouse**
- D. **A process to upgrade the quality of data before it is moved into a data warehouse**

1. Data scrubbing is which of the following?

- A. A process to reject data from the data warehouse and to create the necessary indexes
- B. A process to load the data in the data warehouse and to create the necessary indexes
- C. A process to upgrade the quality of data after it is moved into a data warehouse
- D. A process to upgrade the quality of data before it is moved into a data warehouse ✓

2. The @active data warehouse architecture includes which of the following?

A. At least one data mart

B. Data that can extracted from numerous internal and external sources

C. Near real-time updates

D. All of the above

2. The @active data warehouse architecture includes which of the following?

A. At least one data mart

B. Data that can extracted from numerous internal and external sources

C. Near real-time updates

D. All of the above ✓

3. A goal of data mining includes which of the following?

- A. To explain some observed event or condition**
- B. To confirm that data exists**
- C. To analyze data for expected relationships**
- D. To create a new data warehouse**

3. A goal of data mining includes which of the following?

- A. To explain some observed event or condition ✓**
- B. To confirm that data exists**
- C. To analyze data for expected relationships**
- D. To create a new data warehouse**

4. An operational system is which of the following?

- A. A system that is used to run the business in real time and is based on historical data.**
- B. A system that is used to run the business in real time and is based on current data.**
- C. A system that is used to support decision making and is based on current data.**
- D. A system that is used to support decision making and is based on historical data.**

4. An operational system is which of the following?

A. A system that is used to run the business in real time and is based on historical data.

B. A system that is used to run the business in real time and is based on current data. ✓

C. A system that is used to support decision making and is based on current data.

D. A system that is used to support decision making and is based on historical data.

5. A data warehouse is which of the following?

- A. Can be updated by end users.**
- B. Contains numerous naming conventions and formats.**
- C. Organized around important subject areas.**
- D. Contains only current data.**

5. A data warehouse is which of the following?

- A. Can be updated by end users.**
- B. Contains numerous naming conventions and formats.**
- C. Organized around important subject areas. ✓**
- D. Contains only current data.**

Pipelining improves performance by exploiting instruction level parallelism.

Data that can be modeled as dimension attributes and measure attributes are called _____ data.

- a) Multidimensional
- b) Singledimensional
- c) Measured
- d) Dimensional

Data that can be modeled as dimension attributes and measure attributes are called _____ data.

- a) Multidimensional
- b) Singledimensional
- c) Measured
- d) Dimensional

Given a relation used for data analysis, we can identify some of its attributes as measure attributes, since they measure some value, and can be aggregated upon. Dimension attribute define the dimensions on which measure attributes, and summaries of measure attributes, are viewed.

The generalization of cross-tab which is represented visually is _____ which is also called as data cube.

- a) Two dimensional cube
- b) Multidimensional cube
- c) N-dimensional cube
- d) Cuboid

The generalization of cross-tab which is represented visually is _____ which is also called as data cube.

- a) Two dimensional cube
- b) Multidimensional cube
- c) N-dimensional cube
- d) Cuboid

Each cell in the cube is identified for the values for the three dimensional attributes.

The process of viewing the cross-tab (Single dimensional) with a fixed value of one attribute is

- a) Slicing
- b) Dicing
- c) Pivoting
- d) Both Slicing and Dicing

The process of viewing the cross-tab (Single dimensional) with a fixed value of one attribute is

- a) Slicing
- b) Dicing
- c) Pivoting
- d) Both Slicing and Dicing

The slice operation selects one particular dimension from a given cube and provides a new sub-cube. Dice selects two or more dimensions from a given cube and provides a new sub-cube.

The operation of moving from finer-granularity data to a coarser granularity (by means of aggregation) is called a

- a) Rollup
- b) Drill down
- c) Dicing
- d) Pivoting

The operation of moving from finer-granularity data to a coarser granularity (by means of aggregation) is called a

a) Rollup

b) Drill down

c) Dicing

d) Pivoting

The opposite operation—that of moving from coarser-granularity data to finer-granularity data—is called a drill down.

In SQL the cross-tabs are created using

- a) Slice
- b) Dice
- c) Pivot
- d) All of the mentioned

In SQL the cross-tabs are created using

- a) Slice
- b) Dice
- c) Pivot
- d) All of the mentioned

Pivot (sum(quantity) for color in ('dark','pastel','white')).

What do data warehouses support?

a) OLAP

b) OLTP

c) OLAP and OLTP

d) Operational databases

What do data warehouses support?

a) OLAP

b) OLTP

c) OLAP and OLTP

d) Operational databases

DBMS

UGC NET CS 2020 Compete Solution

Consider a relational schema $S = (U, V, W, X, Y, Z)$ on which the following functional dependencies hold:

$\{U \rightarrow V, VW \rightarrow X, Y \rightarrow W, X \rightarrow U\}$


Which are the candidate keys among following options?

- (1) UY, VY
- (2) UY, VY, XY
- (3) UYZ, VYZ, VWZ
- (4) UYZ, VYZ, XYZ

Consider a relational schema $S = (U, V, W, X, Y, Z)$ on which the following functional dependencies hold:

$\{U \rightarrow V, VW \rightarrow X, Y \rightarrow W, X \rightarrow U\}$

Which are the candidate keys among following options?

- (1) UY, VY
- (2) UY, VY, XY
- (3) UYZ, VYZ, VWZ
- (4)  UYZ, VYZ, XYZ

Correct Ans: Option 4

EMPLOYEE

NAME	VARCHAR (30)	NOT NULL,
EID	VARCHAR (10)	NOT NULL,
DEPTNO	INT (5)	NOT NULL,
HODEID	VARCHAR (10),	
SALARY	INT (10),	

PRIMARY KEY (EID),

FOREIGN KEY (HODEID) REFERENCES EMPLOYEE (EID),

FOREIGN KEY (DEPTNO) REFERENCES DEPARTMENT (DID);

DEPARTMENT

DID	INT (5)	NOT NULL,
DNAME	VARCHAR (30)	NOT NULL,
HODID	VARCHAR (10)	NOT NULL,
HODNAME	VARCHAR (30),	

PRIMARY KEY (DID),

UNIQUE (DNAME),

FOREIGN KEY (HODID) REFERENCES EMPLOYEE (EID);

PROJECT WORK:

EMPID	VARCHAR (10)	NOT NULL,
PROJNO	INT (5)	NOT NULL,
PROJECTLOC	VARCHAR (30)	NOT NULL,

PRIMARY KEY (EMPID, PROJNO),

FOREIGN KEY (EMPID) REFERENCES EMPLOYEE (EID),

Given below are two statements to find the sum of salaries of all employees of the English department as well as the maximum, minimum and average salary in English department.

STATEMENT I: SELECT SUM (SALARY), MAX (SALARY), MIN (SALARY),
AVG (SALARY) FROM EMPLOYEE, DEPARTMENT
WHERE DEPTNO=DID
AND DNAME='ENGLISH';

STATEMENT II: SELECT SUM (SALARY), MAX (SALARY), MIN (SALARY),
AVG (SALARY) FROM EMPLOYEE, DEPARTMENT
WHERE DNAME='ENGLISH';

In the light of the above statements, choose the correct answer from the options given below

- (1) Both Statement I and Statement II are true
- (2) Both Statement I and Statement II are false
- (3) Statement I is correct but Statement II is false
- (4) Statement I is incorrect but Statement II is true

Q.96

Given below are two statements to find the sum of salaries of all employees of the English department as well as the maximum, minimum and average salary in English department.

STATEMENT I: SELECT SUM (SALARY), MAX (SALARY), MIN (SALARY),
AVG (SALARY) FROM EMPLOYEE, DEPARTMENT
WHERE DEPTNO=DID
AND DNAME='ENGLISH';

STATEMENT II: SELECT SUM (SALARY), MAX (SALARY), MIN (SALARY),
AVG (SALARY) FROM EMPLOYEE, DEPARTMENT
WHERE DNAME='ENGLISH';

In the light of the above statements, choose the correct answer from the options given below

- (1) Both Statement I and Statement II are true
- (2) Both Statement I and Statement II are false
- (3) Statement I is correct but Statement II is false ✓
- (4) Statement I is incorrect but Statement II is true

Q.97

In reference to the above given table structures, which of the following query/queries will drop the 'SALARY' column from 'EMPLOYEE' table?

- (A) ALTER TABLE EMPLOYEE DROP SALARY CASCADE;
- (B) ALTER TABLE EMPLOYEE DROP SALARY RESTRICT;
- (C) ALTER EMPLOYEE DROP SALARY;

Choose the correct answer from the options given below:

- (1) (A) and (B) only
- (2) (A) and (C) only
- (3) (B) and (C) only
- (4) (A) only

When an UPDATE or DELETE operation affects a key value in the parent table that has matching rows in the child table, the result depends on the referential action specified by ON UPDATE and ON DELETE subclauses of the FOREIGN KEY clause.

Referential actions include:

CASCADE: Delete or update the row from the parent table and automatically delete or update the matching rows in the child table. Both ON DELETE CASCADE and ON UPDATE CASCADE are supported. Between two tables, do not define several ON UPDATE CASCADE clauses that act on the same column in the parent table or in the child table.

If a FOREIGN KEY clause is defined on both tables in a foreign key relationship, making both tables a parent and child, an ON UPDATE CASCADE or ON DELETE CASCADE subclause defined for one FOREIGN KEY clause must be defined for the other in order for cascading operations to succeed. If an ON UPDATE CASCADE or ON DELETE CASCADE subclause is only defined for one FOREIGN KEY clause, cascading operations fail with an error.

SET NULL: Delete or update the row from the parent table and set the foreign key column or columns in the child table to NULL. Both ON DELETE SET NULL and ON UPDATE SET NULL clauses are supported.

If you specify a SET NULL action, make sure that you have not declared the columns in the child table as NOT NULL.

RESTRICT: Rejects the delete or update operation for the parent table. Specifying RESTRICT (or NO ACTION) is the same as omitting the ON DELETE or ON UPDATE clause.

NO ACTION: A keyword from standard SQL. In MySQL, equivalent to RESTRICT. The MySQL Server rejects the delete or update operation for the parent table if there is a related foreign key value in the referenced table. Some database systems have deferred checks, and NO ACTION is a deferred check. In MySQL, foreign key constraints are checked immediately, so NO ACTION is the same as RESTRICT.

SET DEFAULT: This action is recognized by the MySQL parser, but both InnoDB and NDB reject table definitions containing ON DELETE SET DEFAULT or ON UPDATE SET DEFAULT clauses.

CASCADE : Delete or update the row from the parent table and automatically delete or update the matching rows in the child table. ...

RESTRICT : Rejects the delete or update operation for the parent table. Specifying **RESTRICT** (or **NO ACTION**) is the same as omitting the **ON DELETE** or **ON UPDATE** clause.

Q.97

In reference to the above given table structures, which of the following query/queries will drop the 'SALARY' column from 'EMPLOYEE' table?

- (A) ALTER TABLE EMPLOYEE DROP SALARY CASCADE;
- (B) ALTER TABLE EMPLOYEE DROP SALARY RESTRICT;
- (C) ALTER EMPLOYEE DROP SALARY;

Choose the correct answer from the options given below:

- (1) (A) and (B) only ✓
- (2) (A) and (C) only
- (3) (B) and (C) only
- (4) (A) only

Q.98

Which of the following query/queries return the employee ID and name of employees whose salary is greater than the salary of all employees in department number 20 of university. Order result by employee ID (refer table structures given above).

- (A) SELECT EID, NAME
FROM EMPLOYEE
WHERE SALARY > (SELECT SALARY FROM EMPLOYEE WHERE DEPTNO=20)
ORDER BY EID;
- (B) SELECT EID, NAME
FROM EMPLOYEE WHERE SALARY > (SELECT SALARY FROM EMPLOYEE
WHERE DEPTNO=20);
- (C) SELECT EID, NAME
FROM EMPLOYEE
WHERE SALARY > ALL (SELECT SALARY FROM EMPLOYEE WHERE
DEPTNO=20)
ORDER BY EID

Choose the correct answer from the options given below:

- (1) (A) and (B) only
(2) (A) and (C) only
(3) (B) only
(4) (C) only

The SQL ANY and ALL Operators

The ANY and ALL operators are used with a WHERE or HAVING clause.

The ANY operator returns true if **any of the subquery** values meet the condition.

The ALL operator returns true if **all of the subquery** values meet the condition.

```
SELECT ProductName
FROM Products
WHERE ProductID
= ANY (SELECT ProductID FROM OrderDetails WHERE
Quantity > 99);
```


```
SELECT ProductName
FROM Products
WHERE ProductID
= ANY (SELECT ProductID FROM OrderDetails WHERE
Quantity = 10);
```

Q.98

Which of the following query/queries return the employee ID and name of employees whose salary is greater than the salary of all employees in department number 20 of university. Order result by employee ID (refer table structures given above).

- (A) SELECT EID, NAME
FROM EMPLOYEE
WHERE SALARY > (SELECT SALARY FROM EMPLOYEE WHERE DEPTNO=20)
ORDER BY EID;
- (B) SELECT EID, NAME
FROM EMPLOYEE WHERE SALARY > (SELECT SALARY FROM EMPLOYEE
WHERE DEPTNO=20);
- (C) SELECT EID, NAME
FROM EMPLOYEE
WHERE SALARY > ALL (SELECT SALARY FROM EMPLOYEE WHERE
DEPTNO=20)
ORDER BY EID

Choose the correct answer from the options given below:

- (1) (A) and (B) only
(2) (A) and (C) only
(3) (B) only
(4) (C) only 

Q.99

On the basis of above given table structures, retrieve the distinct employee ID (EMPID) of all employees of university who are working on project No.20, 30 and 40.

- (1)

```
SELECT EMPID
FROM PROJECTWORK
WHERE PROJNO=(20,30,40);
```
- (2)

```
SELECT EMPID
FROM PROJECTWORK
WHERE PROJNO IN (20,30,40);
```
- (3)

```
SELECT DISTINCT EMPID
FROM PROJECTWORK
WHERE PROJNO IN (20,30,40);
```
- (4)

```
SELECT DISTINCT EMPID
FROM PROJECTWORK
WHERE PROJNO=20,30,40;
```

The SQL IN Operator

The IN operator allows you to specify multiple values in a WHERE clause. The IN operator is a shorthand for multiple OR conditions.

IN Syntax

```
SELECT  column_name(s)
FROM    table_name
WHERE   column_name IN (value1, value2, ...);
```

```
SELECT * FROM Customers
WHERE Country IN ('Germany', 'France', 'UK');
```

Q.99

On the basis of above given table structures, retrieve the distinct employee ID (EMPID) of all employees of university who are working on project No.20, 30 and 40.

- (1) SELECT EMPID
FROM PROJECTWORK
WHERE PROJNO=(20,30,40);
- (2) SELECT EMPID
FROM PROJECTWORK
WHERE PROJNO IN (20,30,40);
- (3) SELECT DISTINCT EMPID
FROM PROJECTWORK
WHERE PROJNO IN (20,30,40);
- (4) SELECT DISTINCT EMPID
FROM PROJECTWORK
WHERE PROJNO=20,30,40;



Q.100 Refer table, structures given above. University decided to give all employees in the 'SCIENCE' department a 20% rise in salary. Which of the following query / queries will compute the above results?

- (A) UPDATE EMPLOYEE
SET SALARY = SALARY*1.20
WHERE DEPT NO. IN (SELECT DID FROM DEPARTMENT WHERE
DNAME='SCIENCE');
- (B) UPDATE TABLE EMPLOYEE
SET SALARY = SALARY*1.20
WHERE DNAME='SCIENCE';
- (C) ALTER TABLE EMPLOYEE
SET SALARY=SALARY*1.20
WHERE DEPTNO. IN (SELECT DNAME FROM DEPARTMENT WHERE DNAME
='SCIENCE')

Choose the correct answer from the options given below:

- (1) (A) and (B) only
(2) (A) only
(3) (B) and (C) only
(4) (C) only
-

Q.100 Refer table, structures given above. University decided to give all employees in the 'SCIENCE' department a 20% rise in salary. Which of the following query / queries will compute the above results?

- (A) UPDATE EMPLOYEE
SET SALARY = SALARY*1.20
WHERE DEPT NO. IN (SELECT DID FROM DEPARTMENT WHERE
DNAME='SCIENCE');
- (B) UPDATE TABLE EMPLOYEE
SET SALARY = SALARY*1.20
WHERE DNAME='SCIENCE';
- (C) ALTER TABLE EMPLOYEE
SET SALARY=SALARY*1.20
WHERE DEPTNO. IN (SELECT DNAME FROM DEPARTMENT WHERE DNAME
='SCIENCE')

Choose the correct answer from the options given below:

- (1) (A) and (B) only
- (2) (A) only ✓
- (3) (B) and (C) only
- (4) (C) only
-

GATE 2020 PYQs

Given the relations

employee (*name, salary, deptno*) and
department (*deptno, deptname, address*)

Which of the following queries cannot be expressed using the basic relational algebra

operations ($\cup, -, \times, \pi, \sigma, \rho$)?
(GATE CS 2000)

- (a) Department address of every employee
- (b) Employees whose name is the same as their department name
- (c) The sum of all employees' salaries
- (d) All employees of a given department

Given the relations

employee (*name, salary, deptno*) and
department (*deptno, deptname, address*)

Which of the following queries cannot be expressed using the basic relational algebra

operations ($\cup, -, \times, \pi, \sigma, \rho$)?
(GATE CS 2000)

- (a) Department address of every employee
- (b) Employees whose name is the same as their department name
- (c) The sum of all employees' salaries**
- (d) All employees of a given department

Given the following relation instance.

x	y	z
1	4	2
1	5	3
1	6	3
3	2	2

(GATE CS 2000)

Which of the following functional dependencies are satisfied by the instance?

- (a) $XY \rightarrow Z$ and $Z \rightarrow Y$
- (b) $YZ \rightarrow X$ and $Y \rightarrow Z$
- (c) $YZ \rightarrow X$ and $X \rightarrow Z$
- (d) $XZ \rightarrow Y$ and $Y \rightarrow X$

A functional dependency (FD) is a constraint between two sets of attributes in a relation from a database. A FD $X \rightarrow Y$ requires that the value of X uniquely determines the value of Y where X and Y are sets of attributes. FD is a generalization of the notion of a key.

Given that X , Y , and Z are sets of attributes in a relation R , one can derive several properties of functional dependencies. Among the most important are Armstrong's axioms, which are used in database normalization:

- * Subset Property (Axiom of Reflexivity): If Y is a subset of X , then $X \rightarrow Y$
- * Augmentation (Axiom of Augmentation): If $X \rightarrow Y$, then $XZ \rightarrow YZ$
- * Transitivity (Axiom of Transitivity): If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

Given the following relation instance.

x	y	z
1	4	2
1	5	3
1	6	3
3	2	2

(GATE CS 2000)

Which of the following functional dependencies are satisfied by the instance?

- (a) $XY \rightarrow Z$ and $Z \rightarrow Y$
- (b) $YZ \rightarrow X$ and $Y \rightarrow Z$**
- (c) $YZ \rightarrow X$ and $X \rightarrow Z$
- (d) $XZ \rightarrow Y$ and $Y \rightarrow X$