# DBMS Notes

25/07/2023

DBMS

## Fundamentals :

1] Data : Raw and isolated data about an entity

2] Information : Information about entity which is meaningful
Processed data is information

eg A, P, P, L, E → data
APPLE → Information

3] Database : Collection of similar / Related data.
Represented in tables.

4] Table : Collection of rows and ~~inform~~ columns.

5] DBMS : - Database Management System
- Collection of related data and set of program to access it is DBMS

## Disadvantage of File processing system / Advantages of DBMS

1] Data Redundancy - duplication of data

2] Data Inconsistency - data redundancy leads to it

3] Difficulty in accessing data - scattering of data in various files leads retrival difficult.

4] Data isolation -

5] Security Problem - Secured from unauthorised access.

6] Atomicity issue - Atomicity of transaction is difficult
Atomicity means either all the operation in transaction are done or not.

7] Integrity problem - No automatic constraints check.

## OLAP VS OLTP :

| 1] OLAP : Online Analytical Processing | 2] OLTP |
|---|---|
| - Historical data storing. | - Online transaction process |
| - Subject Orientation | - Current data |
| - decision making | - Application oriented |
| - Size TB, PB | - Day to day operation |
| | - MB, GB |

# ER diagram

1] Introduction : introduced by peter chen
- Entity Relationship diagram
- High level conceptual data model
- visual representation of data.
- collection entities and Hnship betn them.

1] Entity : Real world Object

Types

Tangible
→ Physically exist in real world

Intangible
→ logically exist

- Symbol: Entity-Name

2] Entity Set : collection of similar entities

3] Attribute : - Properties of entity.
- Attribute has domain (acceptable values)
- represented as (Coval)

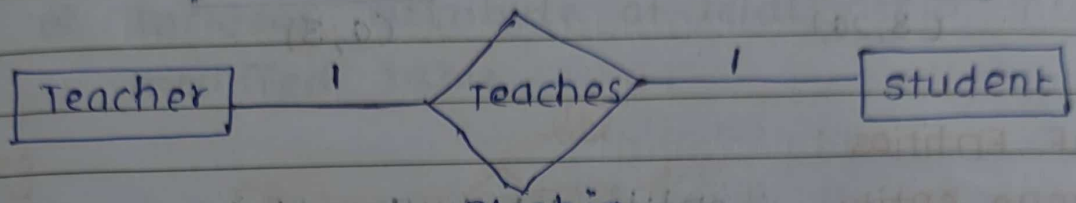
Types :
i) Simple : No Further dividing - RollNo
ii) Composite : further dividing - Name ← First, middle, last
iii) Single Valued : single value - RollNo
iv) Multivalued : Multiple values - Phone_No
v) Stored : Physically stored - DOB
vi) Derived : derived from stored attribute - age

4] Relationship : links betn two or more entities

Types :
1) One-to-One (1:1)
2) One-to-Many (1:M)
3) Many-to-Many (M:N)
4) Many-to-One (M:1)

three components in rlnship
- Name
- Degree
- cardinality ratio / Participation constant

**Degree of relationship**: no. of entity set participated in relationship



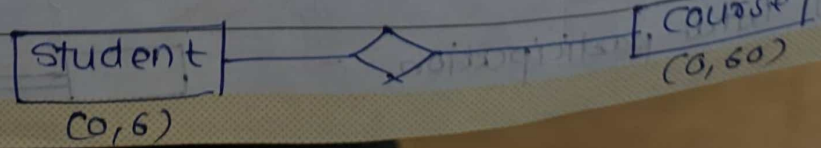
Rlnship

**Mapping cardinality / cardinality ratio**:
→ No. of entities to which other entity can be related via rlnship

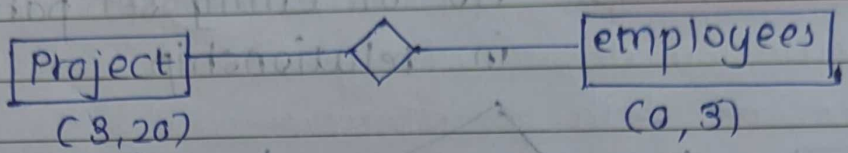
**Participation constraint**:
→ Specifies whether the existence of an entity depends on its being related to another entity via rlnship type

- **max cardinality**: defines max. no times an entity occurrence participating in a rlnship
  If 1 then 1 time PP ~~(partial participation)~~
  If n then ~~n time (Total participation)~~
- **Min cardinality**: If 0 then PP (0, n)
  If 1 then TP (1, n)

Ex: Consider all students need not to register for course, but they can go max upto 6 courses. course can allow max 60 students but all courses need not have



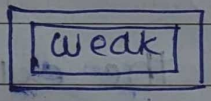
Ex. 2] Project suppose to min 3 employees and max 20, All employees need not be the part of project but they can participate upto 3 at a time.




Types of Entities :

1) Strong entity : entity set with attribute which can be used as primary or candidate key.

2) Weak entity : Strong
↳ entity set without any primary key




ER diagram symbols :

1] Entity -
2] Strong entity -
3] Weak entity -
4] Attribute -
5] Multivalued -
6] Composite -
7] Derived -
8] Rlnship -
9] Total participation. -

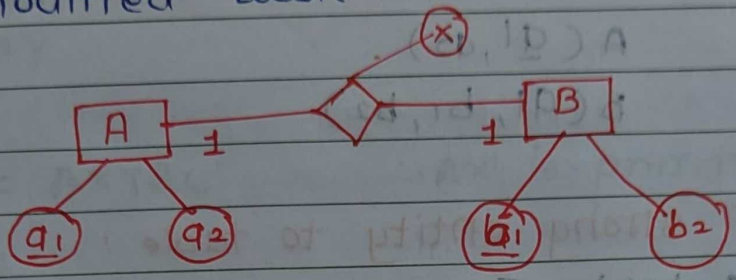
22/07/2023

## ER diagrams to tables:

### 1) Conversion of one-to-one relationship:

Steps: ① either A or B should be modified, include primary key of other side as foreign key.
② A or B is having total participation, then that should be modified table.
③ Includes attribute of relationship into modified table

eg.

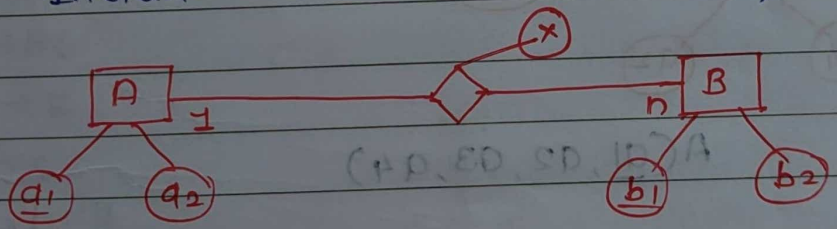



A table = A (a1, a2)
B table = B (b1, b2, a1, x)

### 2) Conversion of one to many relationship:

Steps: ① modify n side table
② Include primary key of 1 side into n side.
③ Include attribute of Rnship into it.

eg

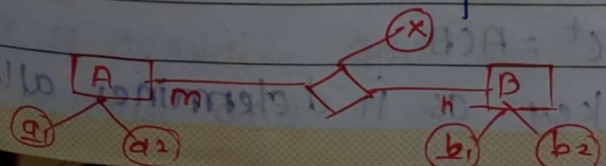



A table = A (a1, a2)
B table = B (b1, b2, a1, x)

### 3) Conversion of many to many relationship:

Steps: ① Create new table including the primary key of m and n side as foreign keys in new table.
② Declare combination of ~~primary~~ foreign keys as primary key for new table.
③ Include Rnship attributes too.

eg.

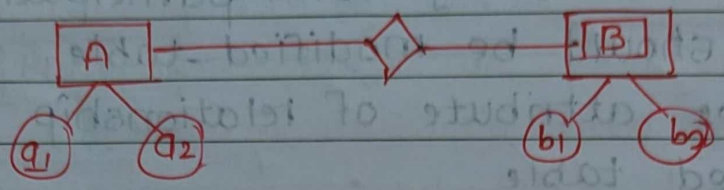



X table = Z (a1, b1, x)

4) Conversion of weak entity to table :

① create table 1 which contains primary key.

② create table 2 include above primary key as foreign key in it




A(a1, a2)

B(a1, b1, b2)

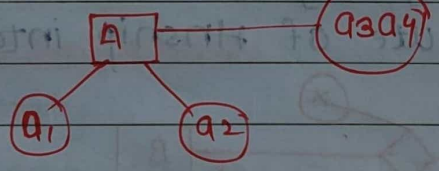
5) Conversion of strong entity to table :

① create table for each strong entity

② Include all attributes, if composite attribute divide it into simple attributes.

③ ignore multivalued attributes at this stage

④ select primary key for table




A(a1, a2, a3, a4)

01/08/2023

## Functional Dependency :

- Defines association among two sets of attributes

1) $A \to B$
$B \to C$
$A \to C$
$AC \to B$
$AB \to C$

} Find primary key : to find calculate Closure ($n^+$) of all

$A^+ = ABCB$ $A^+ = ABC$ $AB^+ = ABC$

$B^+ = BC$ $AC^+ = ACB$

$\therefore A^+$ is primary key as it determines all remaining

attributes. and it is smallest.

2) R(A, B, C, X, Y)

$A \rightarrow B$

$B \rightarrow C$

$A \rightarrow C$

$AB \rightarrow C$

$AC \rightarrow B$

$X \rightarrow Y$

$Ax^+ = AXYBC$ ∴ Ax is primary key here.

3) $A \rightarrow B$

$BC \rightarrow DE$

$AEG \rightarrow G$

$(AC)^+ = ?$ find

→ $AC^+ = ACBDE$

4) R(ABCDEFGH)

$A \rightarrow BC$

$CD \rightarrow E$

$E \rightarrow C$

$D \rightarrow AEH$

$ABH \rightarrow BD$

$DH \rightarrow BC$

$BCD \rightarrow H$ — is this possible

→ Take $BCD^+ = BCDEAH$ —— ∴ possible

**Algorithm to identify closure set of attributes:**

- Equate an attribute to x whose closure to be find.
- Repeatedly take functional dependency one by one and check whether left side attribute is available or not
- If available add right hand side attribute to x.
- Repeat and stop.

→ Applications using closure set of attributes :

(2m) 1) Find out additional functional dependency.

(3m) 2) To find out equivalence.

3) To identify candidate key

(6-7m) 4) To identify irreducible set of functional dependency or canonical team.

Examples :

1) $A \rightarrow BC$
$CD \rightarrow E$
$E \rightarrow C$
$D \rightarrow AEH$
$ABH \rightarrow BD$
$DH \rightarrow BC$

find $BCD \Rightarrow$ ? is possible and primary key

→ Find closure of BCD

$\therefore BCD^+ = BCDEAH$ — ∴ possible

$D^+ = DAEHBC$ — primary key.

2) Equivalence :

i)

| F1 | F2 |
|---|---|
| $A \rightarrow B$ | $A \rightarrow B$ |
| $C \rightarrow HD$ | $D \rightarrow BC$ |
| $F \rightarrow E$ | $E \rightarrow AF$ |

Steps : 1) Take F1 or F2 side
2) Calculate closure of that side using another side
3) Then cross check

→ i) $A^+$
$C^+$
$F^+$
Find closure from F2 side

ii)
$A^+ = AB$

$C^+ = C$

$F^+ = F$

iii) Here

$A^+, C^+, F^+$ closure do not satisfy $F_1$

iv) NO equivalence.

ii)

| $F_1$ | $F_2$ |
|---|---|
| $A \to C$ | |
| $AC \to D$ | $A \to CD$ |
| $E \to AD$ | $E \to AH$ |
| $E \to H$ | |

→ i) $A^+ = A\underline{CD}$

$Ac^+ = \underline{ACD}$

$E^+ = \underline{EAHDC}$

∴ all are covered

∴ equivalence

iii)

| $F_1$ | $F_2$ |
|---|---|
| $B \to CD$ | $B \to CDE$ |
| $AD \to E$ | $B \to ABC$ |
| $B \to A$ | $AD \to E$ |

$B^+ = \underline{BCDEA}$

$AD^+ = \underline{ADE}$

∴ all are covered

∴ equivalence.

03/08/2023

3] Canonical Team / Terms :

R(w, x, y, z)

① $x \to w$

$wz \to x$

$wz \to y$

$y \to wxz$

— 1] $x \to w$ —(1)

$wz \to x$ —(2)

$wz \to y$ —(3)

$y \to w$ —(4)

$y \to x$ —(5)

$y \to z$ —(6)

1) Calculate $x^+$ ignore (1)

$x^+ = x$ —(1) mandatory

2) Calculate $wz^+$ ignore (2)

$wz^+ = wzyx$ —(2) redundant

∴ New set of F.D.

$x \to w$

$wz \to y$

$y \to w$

$y \to x$

$y \to z$

3) Calculate $wz^+$ — ignore (3)

$wz^+ = wz$ —(3) mandatory

4) Calculate $y^+$ — ignore (4)

$y^+ = xzwy$ — redundant

∴ new set

$x \to w$

$wz \to y$

$y \to x$

$y \to z$

***Steps :**

Inorder to remove redundant elements.

1] Decomposition

2] Find closure of each and ignore it if it is redundant

3] Find essential one's.

4] Check for composite attributes and reduce

5] calculate $y+$ —— ignore (5)
$y+ = yz$ —— (5) mandatory

6] calculate $y+$ — ignore (6)
$y+ = yx$ —— (6) mandatory

$\therefore$ $x \to w$
$wz \to y$
$y \to x$
$y \to z$

i.e $x \to w$
$wz \to y$
$y \to xz$
} $\therefore$ F.D set

- Checking composite attribute

case : 01 Hide $w$

| | |
|---|---|
| $x \to w$ | $x \to w$ |
| $wz \to y$ | $wz \to y$ |
| $y \to xz$ | $y \to x$ |
| $y+ = z$ | $z+ = zyxw$ |

$\therefore$ Both closure are not equal thus we can't eliminate $w$.

case : 02 Hide $z$

| | |
|---|---|
| $x \to w$ | $x \to w$ |
| $wz \to y$ | $w \to y$ |
| $y \to xz$ | $y \to x$ |
| $w+ = w$ | $w+ = wyx$ |

$\therefore$ can't eliminate $z$

Final F.D = $x \to w$
$wz \to y$
$y \to xz$

Note : If closure same, recheck F.D for any repetition (repeate whole process from start)
If any composite attribute reduce it to one and again check for F.D. from 1st.

Q. R(ABCD)

A → B
C → B
D → ABC
AC → D

⇒ (1)
A → B — (1)
C → B — (2)
D → A — (3)
D → B — (4) ×
D → C — (5)
AC → D — (6)

(2)
- A+ = ? — ignore (1)
A+ = A — mandatory
- C+ = ? — ignore (2)
C+ = C — mandatory
- D+ = ? — ignore (3)
D+ = DBC — mandatory
- D+ = ? — ignore (4)
D+ = DABC — redundant
- D+ = ? — ignore (5)
D+ = DBA — mandatory
- AC+ = ? — ignore (6)
AC = ACB — mandatory

F.D set
A → B
C → B
D → AC
AC → D

(2) check for composite :

case : 01 ignore 'A'

| | |
|---|---|
| $A \to B$ | $A \to B$ |
| $C \to B$ | $C \to B$ |
| $D \to AC$ | $D \to AC$ |
| $AC \to D$ | $C \to B$ |
| $AC^+ = ABCD$ | $C^+ = BC$ |

can't eliminate 'A'

case: 02 ignore 'B'

| | |
|---|---|
| $A \to B$ | $A \to B$ |
| $C \to B$ | $C \to B$ |
| $D \to AC$ | $D \to AC$ |
| $AC \to D$ | $A \to D$ |
| $A^+ = AB$ | $A^+ = ABD$ |

can't eliminate 'C'

∴ Final F.D.

$A \to B$

$C \to B$

$D \to AC$

$AC \to D$

(3) $AB \to C$

$C \to B$

$A \to B$

⇒ $AB \to C$ — (1)

$C \to B$ — (2)

$A \to B$ — (3)

$AB^+ = AB$ — ignoring (1) ∴ mandatory

$C^+ = C$ — ignoring (2) ∴ mandatory

$A^+ = AB$ — ignoring (3) ∴ mandatory

composite key

| | | |
|---|---|---|
| $AB \to C$ | $B \to C$ | $A \to C$ |
| $C \to B$ | $C \to B$ | $C \to B$ |
| $A \to B$ | $A \to B$ | $A \to B$ |
| | Ignore A | ignore B |
| $A^+ = AB$ | $A^+ = ABC$ | $B^+ = B$ |
| $B^+ = B$ | | |

∴ We can eliminate B.

∴ Final F.D

$A \rightarrow C$ — (1)

$C \rightarrow B$ — (2)

$A \rightarrow B$ — (3) ✗

$A^+ = AB$ — ignoring (1) mandatory

$C^+ = C$ — ignoring (2) mandatory

$A^+ = ACB$ — Mad Redundant

∴ Final FD

$A \rightarrow C$

$C \rightarrow B$

04/08/2023

Q. R(X, Y, Z)

$X \rightarrow YZ$

$Y \rightarrow Z$

$X \rightarrow Y$

$XY \rightarrow Z$

→

① $X \rightarrow Y$ — (1) ✗

$X \rightarrow Z$ — (2) ✗

$Y \rightarrow Z$ — (3)

$X \rightarrow Y$ — (4)

$XY \rightarrow Z$ — (5)

② $X^+ = XZY$ — Redundant

$X^+ = XYZ$ — Redundant

$Y^+ = Y$ — mandatory

$X^+ = X$ — madatory

$XY^+ = XYZ$ — Redundant

∴ Final F.D

$Y \rightarrow Z$

$X \rightarrow Y$

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

$A \to BC$

$CD \to E$

$B \to D$

$E \to A$

→ ① $A \to B$ —(1)

$A \to C$ —(2)

$CD \to E$ —(3)

$B \to D$ —(4)

$E \to A$ —(5)

② $A^+ = AC$ — mandatory

$A^+ = ABD$ — mandatory

$CD^+ = CD$ — mandatory

$B^+ = B$ — mandatory

$E^+ = E$ — mandatory

③ Try for composite

$CD^+ = CDE$

$C^+ = C$ } — mandatory

$D^+ = D$ }

④ Find F.D.

$A \to BC$

$CD \to E$

$B \to D$

$E \to A$

Q R(A, B, C)

$A \to BC$

$B \to CA$

$AB \to C$

→

① $A \to B$ $AB \to C$

$A \to C$ ✗

$B \to C$ ✗

$B \to A$

② $A^+ = AC$ — mandatory
$A^+ = ABC$ — ignore / Redundant
$B^+ = BAC$ — Redundant
$B^+ = B$ — mandatory
$AB^+ = AB$ — mandatory.

③ Find composite
~~$AB^+ = ABC$~~
$A^+ = ABC$
$B^+ =$

Find composite

| | | | |
|---|---|---|---|
| $A \to BC$ | $A \to B$ | $A \to B$ | $A \to B$ |
| $B \to A$ | $B \to A$ | $B \to A$ | $B \to A$ |
| $AB \to C$ | $AB \to C$ | $AB \to C$ | $A \to C$ |

ignore A
$\therefore B^+ = BAC = AB^+ = BCA$
$\therefore$ we can eliminate A

ignore B
$\therefore A^+ = ABC = A^+ = ACB$
$\therefore$ we can eliminate B

④ Final F.D.
$A \to B$
$B \to A$
~~$A \to C$~~ ✗
$B \to C$

⑤ Repeate Process :
$A^+ = AC$ — mandatory
$B^+ = BC$ — mandatory
~~$A^+$~~ $= BAC$ — Redundant
$B^+ = BA$ — mandatory.

⑥ Final F.D
$A \to B$
$B \to A$
$B \to C$

## Keys :

- In order to access record uniquely.
- It is attribute or set of attributes for unique identification of row or tuple.
- (key)$^+$ = Relation

① Superkey (S.K) : - attribute or set of attributes that uniquely identify each record in relation.
- It is just theoretical concept.
∴ max super key possible = $2^n - 1$

② Candidate key (C.K) : - Candidate key is superkey whose proper subset is not a superkey.

### Examples :

① R(ABCD)
A → BC
Here $A^+ = ABC$
∴ it is not a key

② R(ABCD)
ABC → D
AB → CD
A → BCD
Here
$ABC^+ = ABCD$
$AB^+ = ABCD$
$A^+ = ABCD$
∴ Here all are keys
* All three are super key
* A is candidate key.

③ R(AB CD)
B → ACB
ACD → B
keys.
$B^+ = BAC$
$ACD^+ = ACDB$
Here ACD is key.
ACD is super key.
ACD is candidate key.

④ R(ABCD)
AB → C
C → BD
D → A
Here $(AB)^+ = ABCD$
$(C)^+ = CBDA$
$(D)^+ = DA$
* $(AB)^+, (C)^+$ = keys / SK
* $(AB)^+, (C)^+$ = CK.

### Q Identify Candidate keys.

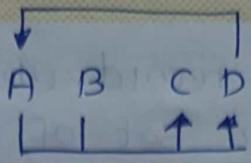
1) R(ABCD)
A → B
B → C
C → A

⇒ R(A B C D)

$AD^+ = ABDC$
$BD^+ = BDCA$
$CD^+ = BDCA$
} C.K

② R(A B C D)

AB → CD
D → A




= No incoming edge towards B, so B is must in C.K.

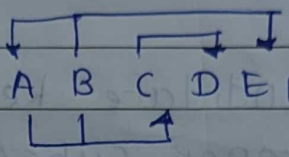
$AB^+ = ABCD$ — C.k

$AC^+ = AC$

$BD^+ = BDAC$ — C.k

③ R(A B C D E F)

AB → C
C → D
B → AE

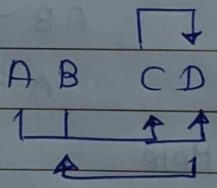



= BF is must $BF^+ = BFAECD$ — C.k

~~BF A+ = ABCDE~~

④ R(A B C D)

AB → CD
C → D
D → B



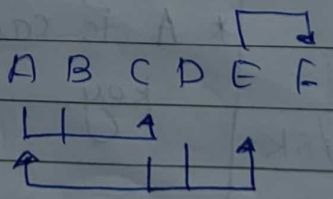

= A is must

$AB^+ = ABCD$ — C.k

$AC^+ = AC$

$AD^+ = ADBC$ — C.k

⑤ R(A B C D E F)




$BD^+ = BD$

BDA = BDAEF — C.k.

BDC = BDCAEF —

BDE = BDEF ×

BDF = BDF ×

~~BDE BDF~~ = BDEF ×

⑥ R(A B C D E)

AE → D

D → B

C → A

⇒ $CE^+$ = CEDBA ✓ Ck

⑦ R(A B C D E F G H I J)

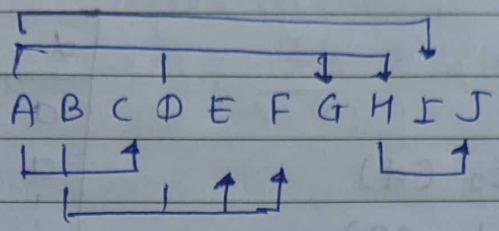
AB → C

AD → GH

BD → EF

A → I

H → J



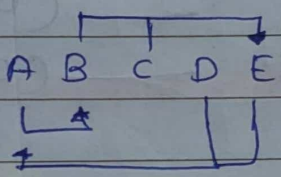

⇒ $ABD^+$ = ABDCGHIJ ✓ Ck.

⑧ R(A B C D E)

A → B

BC → E

DE → A




- CD is essential

CDA = CDABE ✓

CDB = CDBAE ✓ } Ck.

CDE = CDEAB ✓

⑨ Primary key :

- Primary key is ~~foreign~~ is the candidate key which can be never null
- Primary key is choosen from candidate key based on uniqueness and stability.
- In relation there is only one primary key.

⑩ Foreign key :

- It creates rlnship betn tables by referencing primary key of another tables.
- foreign key in one table refers to primary key in another table.

## Functional Dependency:

- R/nship betn two sets of attributes within a rln.

FD: $A \longrightarrow B$ (from value of A we can find value of B)

(determinant) (dependent)

- Values of one set of attributes uniquely determines the values of another set.

$\alpha \subseteq R$, $B \subseteq R$

if $\alpha \longrightarrow \beta$

$t_1(\alpha) = t_2(\alpha)$

then $t_1(B) = t_2(B)$

∴ it is F.D.

R

| $\alpha$ | $\beta$ | |
|---|---|---|
| a | b | ← $t_1$ |
| a | b | ← $t_2$ |
| ⋮ | ⋮ | |

types:

$\alpha \to \beta$ [1] trivial : $AB \to A$ —— ($B \subseteq \alpha$)

[2] Non-trivial : $AB \to C$ —— ($B \not\subseteq \alpha$)

Q] Find valid functional dependency:

R

| A | B | C | D | E |
|---|---|---|---|---|
| a | 2 | 3 | 4 | 5 |
| 2 | a | 3 | 4 | 5 |
| a | 2 | 3 | 6 | 5 |
| a | 2 | 3 | 6 | 6 |

(A) $A \to BC$

(B) $DE \to C$

(C) $C \to D$

(D) $BC \to A$

- How to find dependency is valid or not

→

① check $\alpha$ values, if all $\alpha$ values are di... then valid

② check $\beta$ values, if all $\beta$ values are sa... then valid

③ check manually.

## Armstrong's Axioms:

→ Primary Axioms (RAT)

1] Reflexivity: if $y \subseteq x$ then $x \to y$

2] Augmentation: if $x \to y$ then $xz \to yz$

3] Transitivity: if $x \to y$ && $y \to z$ then $x \to z$

→ Secondary Axioms :

1] Union : if $X \to Y$ && $X \to Z$
$X \to YZ$

2] Decomposition : if $X \to YZ$
$X \to Y$ && $X \to Z$

3] Pseudo transitivity : if $X \to Y$ && $WY \to Z$
then $WX \to Z$

4] Composition : if $X \to Y$ && $Z \to W$
$XZ \to YW$

→ Closure :
→ Equivalence :
→ Canonical : (Irreducible set of functional depency)

## Insertion, Deletion and Updation Anomalies :

1] Insertion anomaly :
- when certain attribute cannot be added to the database without adding unrelated or incorrect data

2] Deletion anomaly :
- This happens when removing data leads to the unexpected loss of data.

3] Update anomaly :
- updating data in one place leads to inconsistencies elsewhere

08/08/2023

## Normalization :

- Let's understand types of functional dependency :
  1. Partial
  2. Total
  3. Transitive

**[1] Total FD :**

- when an attribute is dependent on the entire candidate key.
- Non-prime attributes are determined by a candidate key.
- eg. BCD is primary key / C
  BCD → K ___ Total

**[2] Partial FD :**

- Here attribute depends only on part of candidate key instead of candidate key.
- Non prime attributes are determined by part of candidate key.
- eg. ABCD is primary key / candidate key
  BCD → K ___ partial FD.
  BC → T

**[3] Transitive FD :**

- attribute depends on another non-key attribute.
- Non-prime attributes are determined by other non-prime attribute.
- eg. BCD is PK / CK
  A → E ___ Transitive FD.

**NORMALIZATION :**

- Techniques used to minimize data redundancy and improve data integrity
- It involves breaking down a database table into smaller, related tables to eliminate duplication of data and reduce inconsistencies.
- Process organized into series of normal forms

**• Prime attributes :**
Attributes that are part of candidate keys.

**• Non prime attributes :**
Attributes that are not part of any candidate key

normal forms:

1) 1NF
2) 2NF
3) 3NF
4) BCNF

① 1NF : First Normal Form.
- Ensure each column (attribute) in table contains only atomic (indivisible) values, and there are no repeating groups.

② 2NF : Second Normal Form
- It must follow 1NF
- Ensure that non-prime attributes are Fully / Totally Funtional dependent on entire primary key.
-

③ 3NF :
- It must follow 2NF
- Eliminating transitive depencies, where non-prime attribute determining other non-prime attribute.

④ BCNF
- Boyce-Codd Normal Form extension of 3NF
- It must follow 3NF
- Ensuring for every non-trivial Functional dependency $X \rightarrow Y$, x is a superkey.

* First Normal Form (1NF)

| ID | Name | course |
|---|---|---|
| 1 | khushi | C1, C2 |
| 2 | Saee | C3, C4 |

1NF →

| ID | Name | course |
|---|---|---|
| 1 | khushi | C1 |
| 1 | khushi | C2 |
| 2 | Saee | C3 |
| 2 | Saee | C4 |

*atomic value in each column

## * Second Normal Form :

① Following 1st NF

② check partial dependency it should not be present

steps

i] Find candidate key

ii] Find prime and non prime attribute

iii] if anyonprime attribute is dependent on part of candidate key then Partial dependency.

iv] so, if Partial dependency exsist then not in 2NF.

-so how to translate it to 2NF when PD found

v] Create new relation with it.

Example 1]

R(ABCD)

AB → D

B → C

⇒ 1) It is 1NF.

2) there is PD :- AB → D
B → C

(AB)+ = ABDC — candidate key

(B)+ = BC

3) prime = A,B

Non prime = C, D

4) As C is dependent on B (B → C)

∴ PD exist.

5) Remove :

R1 (AB D)

R2 (B C)

∴ Above is in 2NF.

(2) R( ABCDEFGHIJ)

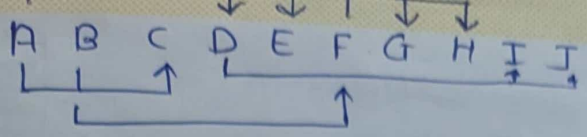
AB → C

A → DE

B → F

F → GH

D → IJ




⇒ 1] $(AB)^+$ = ABCFGHDEIJ —— candidate key.

2] Prime = AB

Non prime = CDEFGHIJ

3] i) A → DE } PD

ii) B → F

4] ∴ R1 (ABC)
R2 (ADEIJ)
R3 (BFGH)

Final tables : R1 (ABC)
R2 (ADEIJ)
R3 (BFGH)

③ R(ABC)

B → C

→ A B C

- $(AB)^+$ = ABC —— candidate key

- Prime = A, B

Non prime = C

- B → C — PD exist

- ∴ R1 (AB)
R2 (BC)

④ R(ABCDE)

AB → C

D → E

— A B C D E

→ $ABD^+$ = ABCDE —— CK.

→ Prime = A B D

- Non prime = C E

→ Here PD = '

∴ Is not in 2NF as AB → C, D → E } PD

$\therefore$ $R_1(ABD)$
$R_2(ABC)$
$R_3(DE)$ In 2NF

⑤ R(ABCDE)

$A \to B$

$B \to E$

$C \to D$

→ ABCDE

- $(AC)^+ = ACDBE$ — CK
- prime = A, C
  Non prime = B, D, E
- Here $A \to B$, $C \to D$ } P.D
- Remove PD.
  → $R_1(ACE)$
  → $R_2(ABF)$
  → $R_3(CD)$
  Now in 2NF.

⑥ R(ABCDEFGHIJ)

$AB \to C$

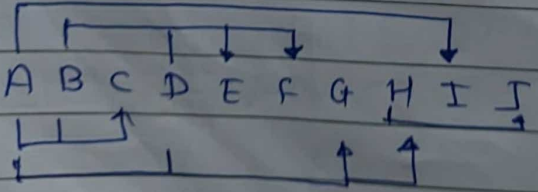
$AD \to GH$

$BD \to EF$

$A \to I$

$H \to J$

→




- $(ABD)^+ = ABDCGHJIEF$ — CK
- Prime = A, B, D
  Non prime = C, E, F, G, H, I, J

- Here

AB → C
AD → GH
BD → EF
A → I

} PD exist.

- Remove

→ R1 (ABCI) —(if I is included here will lead to PD)
→ R2 (ADGHJ)
→ R3 (BDEF)
→ R4 (AI)
→ R5 (ABD)

09/08/2023

## * 3NF (Third Normal Form):

- Must be in 2NF
- check transitive dependency, it should not be present.

steps:

1. check it is in 2NF or not
2. if not convert to 2NF then
3. check if non-prime determine non prime then transitive dependency exist thus not in 3NF
4. Remove it to convert into 3NF.
   go for decomposition.

Example 1:

R(ABCD)

AB → C

C → D

A B C D

(1) (AB)+ = ABCD —C.K

(2) Prime = A, B

non prime = C, D

(3) No P.D here. ∴ It is in 2NF.

4) $C \to D$ i.e non-prime determines non-prime.

∴ TD exist

5) Remove it by decomposition

∴ $R_1$ (ABC)
$R_2$ (CD)

Now it is in 3NF.

② R (ABC)

$A \to B$

$B \to C$

- ABC

→ Candidate key ⇒ $A^+ = ABC$

→ Prime : A

non-prime : BC

→ No PD ∴ In 2NF

- $B \to C$ — (TD exist)

→ Remove it

$R_1$ (AB)
$R_2$ (BC)

• Short trick for 3NF identification
→ $\alpha \to \beta$
$\alpha$ : super key
$\beta$ : prime
if any above exist then it is in 3NF
(All dependencies we have to check, if all follows above then only in 3NF)

③ R (ABCDE)

$AB \to C$

$B \to D$

$D \to E$

→ ABC DE

- $(AB)^+ = ABCDE$ — CK
- Prime = A, B
non prime = C, D, E
- $B \to D$ ∃ PD. Not in 2NF
- Convert to 2NF.

→ R1 (BDE)
→ R2 (AB)
} Now in 2NF.

as B → D
   D → E
} t.D exist not in 3NF

→ R1 (BD)
   R2 (DE)
   R3 (AB)

Now in 3NF.

④ R(ABCDE)
A → B
B → E
C → D

Method ① A B C D E

- (AC)+ = ABCDE — ck
- Prime = A, C
  non prime = B, D, E
- B → E — (∵ transitivity exist) ∴ Not in 3NF
- Removal:

R1 (ABE) → R1 (A, B), R2 (B, E)
R2 (CD)
R3 (AC)

∴ R1 (A, B)
R2 (B E)
R3 (C D)
R4 (A C)

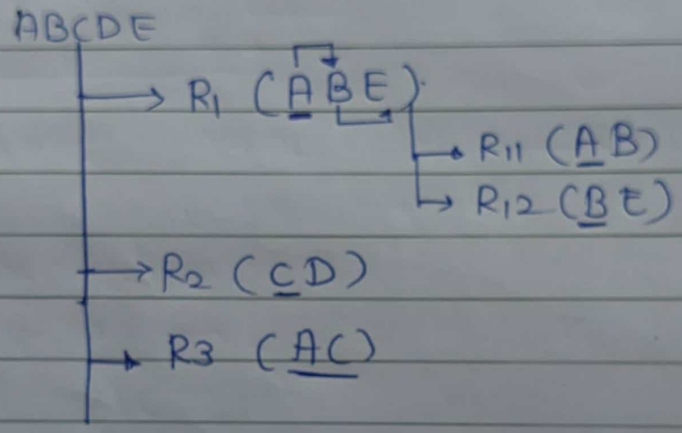
Now in 3NF.

Method (2) $\alpha \rightarrow \beta$ — where $\alpha$ superkey or $\beta$ prime

- $(AC)^+ = ABCDE$ — CK.
- prime = A, C
  Non-prime = B, D, E
- $A \rightarrow B$ ×
  $B \rightarrow E$ ×
  $C \rightarrow D$ ×
  } checked according to $\alpha \rightarrow \beta$

∴ Not in 3NF

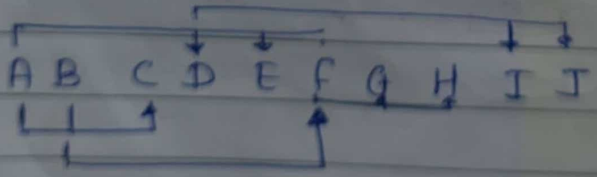
- Removal :




⑤ R(ABCDEFGHIJ)

— AB → C
A → DG
B → F
F → GH
D → IJ

→




- $(AB)^+ = ABCDEFGHIJ$ — CK.
- Prime = A, B
  Non prime = C, D, E, F, G, H, I, J

- $A \rightarrow DE$
  $B \rightarrow F$
  } P.D exist Not in 2NF

- R1 (ADEIJ)
  R2 (BFGH)
  R3 (ABC)

Now in 2NF.

- as

A → DE } TD.
D → IJ }

B → F } FD
F → GH }

- Removal Final Tables

R1 (ADE)
R2 (DIJ
R3 (BF)
R4 (FGH)
R5 (ABC)

R (ABCDEFGHIJ)

AB → C ~
AD → GH ✓
BD → EF
A → I ~
H → J ✓

direct method for 3NF and stepwise solution

Method 1] direct method for 3NF :

1] (ABD)+ = ABCDEFGHIJ
As ABD is candidate key.

2] check by trick of α → β

AB → C —(1) ⊗ AB not superkey, C not prime

thus not in 3NF

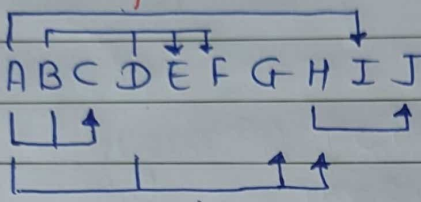
3] → R1 (ABCI)
→ (ABC)
→ (AI)
→ R2 (ADGHJ) → (ADGH)
→ (HJ)
→ R3 (BDEF)
→ R4 (ABD)

∴ Final tables:

$R_1$ (ABCI)
$R_2$ (ABC)
$R_3$ (AJ)
$R_4$ (ADGHI)
$R_5$ (HI)
$R_6$ (BDEF)

Method 2] Stepwise solution for 7 marks:

1) A B C D E F G H I J



$(ABD)^+$ = ABDCEFGHIJ —— Ck.

2) Prime = A, B, D
Non prime = C, E, F, G, H, I, J

3)
AB → C
AD → GH
BD → EF
A → I
} P.D ∴ Not in 2NF

4) Now 2NF: $R_1$ (AB(C))
$R_2$ (ADGHJ)
$R_3$ (BDEF)
$R_4$ (AI)
$R_5$ (ABD)

5)
AD → GH
H → J
} T.D ∴ Not in 3NF

6) decomposition to get 3NF

$R_1$ (AB(C))
$R_2$ (ADGH)
$R_3$ (HJ)
$R_4$ (BDEF)
$R_5$ (AI)
$R_6$ (ABD) in 3NF.

(i) R(ABCD)

$A \to B$

$B \to C$

$C \to D$

ABCD

$A^+ = ABCD$ — [k.

Prime = A

non prime = B, C, D

No P.D ∴ In 2NF

$A \to B$
$B \to C$
$C \to D$
} Are TD., No 3NF

∴ After removal

$R_1$ (A B)

$R_2$ (B C)

$R_3$ (C D) in 3NF.

(ii) $ABD \to AC$

$C \to BE$

$AD \to BF$

$B \to E$

Here,

Candidate key is $AD^+ = ABCDEF$

But ABD exist so, go for canonical form

∴ Hide B

$AD^+ = ABCDEF$

without Hide B

$AD^+ = ABCDEF$

∴ B can be diminate

$AB \to AC$
$C \to BE$
$AD \to BF$
$B \to E$
} again go for canonical

AD → A ✗

AD → C ✓

C → B ✓

C → E ✗

AD → B ✗

AD → F ✓

B → E ✓

∴ $AD^+ = ADCBEF$ ignore

$AD^+ = ADBFE$ — ✓

$C^+ = CE$ — ✓

$C^+ = CBE$ — ignore

$AD^+ = ADFCBE$ — ignore

$AD^+ = ADCBE$ — ✓

$B^+ = B$ — ✓

∴ Final form

AD → C

C → BE

AD → F

B → E

Go for normalization

- $AD^+ = ADCBEF$ — ck.

- prime = A, D  non-prime = B, C, E, F

- No P.D ∴ 2NF

- AD → C
  C → BE
  B → E
  } TD., No 3NF

- Removal :

$R_1$ (≤ B)

$R_2$ (B E)

$R_3$ (AD CF)

Now in 3NF

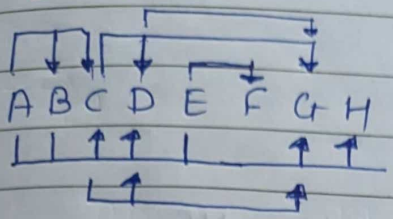
A → BC

(g) ABE → CDGH

C → GD

D → G

E → F




$AE^+$ = ABCDEGH ——— candidate key

ABE is also there so go^ for canonical hide 'B'

$AE^+$ = ABCDEGH

without hide B :-

$AE^+$ = ABCDEGH

∴ B can be eliminate

A → BC

AE → CDGH

C → GD

D → G

E → F

A → B = $A^+$ = A ✓

A → C = $A^+$ = AB ✓

AE → C = $AE^+$ = AEDGHBCF ✗

AE → D = $AE^+$ = ABEGHFCD ✗

AE → G = $AE^+$ = AEBCHGDF ✗

AE → H = $AE^+$ = AEBFCGD ✓

C → G = $C^+$ = CDG ✗

C → D = $C^+$ = C ✓

D → G = $D^+$ = D ✓

E → F = $E^+$ = E ✓

∴ final : A → BC

AE → H

C → D

D → G

– Go for normalization

AE+ — Candidate key

Prime = A, E

Non prime = B, C, D, G, H

Here

$A \to BG$ } P.D , Not in 2NF

$E \to F$ }

∴ After 2NF is applied =

$R_1$ (ABGCD)

$R_2$ (CEF)

$R_3$ (AEH) ∴ in 2NF

But Here

$A \to BC$ }

$C \to D$ } T.D , Not in 3NF

$D \to G$ }

Apply 3NF :

$R_1$ (ABC)

$R_2$ (CD)

$R_3$ (DG)

$R_4$ (EF)

$R_5$ (AEH)

(10) $BCD \to A$

$BC \to E$

$A \to F$

$F \to G$

$C \to D$

$A \to G$

A B C D E F G

$BC^+$ = BCADEFG —— ok

BCD exist so ignore D by canonical

∴ BC → A
BC → E
A → F
F → G
C → D
A → G

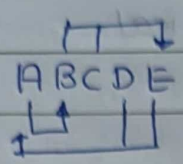
} Go for canonical

| | With | Without | |
|---|---|---|---|
| $BC^+$ | BCAFGED | BCED | |
| $BC^+$ | BCAFGDE | BCAFGD | |
| $A^+$ | AFG | AG | |
| $F^+$ | FG | F | |
| $C^+$ | CD | C | |
| $A^+$ | AGF | AFG | ignored. |

(11) $A \to B$

$BC \to E$

$ED \to A$




$ACD^+ = ACDBE$ — C.k

$BCD^+ = BCDEA$ — C.k

you can go with any C.k

Prime = B, C, D

Non prime = A, E

$BC \to E$ — P.D Not in 2NF.

- Conversion

  $R_1$ ( $\underline{BC}E$ )

  $R_2$ ( $\underline{BCD}EA$ )

  Now in 2NF

- $A \to B$, $ED \to A$ } T.D. Not in 3NF

· Conversion

$R_1$ ( $\underline{A}B$ )

$R_2$ ( $\underline{BC}E$ )

(12) AB is Key

$AB \to CDE$

$A \to C$

$D \to E$

— AB is Ck

Prime = A, B

Non prime = C, D, E

$A \rightarrow C$ — P.D — not in 2NF

∴ conversion

$R_1$ (A C)

$R_2$ (AB DE)

Now in 2NF

$D \rightarrow E$ — T.D not in 2NF

$R_1$ (DE)

$R_2$ (ABD)

$R_3$ (AC)

Now in 3NF.

(B) $AB \rightarrow CDE$

$A \rightarrow C$

$C \rightarrow D$ AB key

→ AB is key

A, B = Prime

C, D, E = Non prime

$A \rightarrow C$ — P.D — Not in 2NF

∴ conversion

$R_1$ (ACD)

$R_2$ (ABE)

Now in 2NF

$A \rightarrow C$

$C \rightarrow D$ — T.D — Not in 3NF.

∴ Conversion

$R_1$ (CD)

$R_2$ (AC)

$R_3$ (ABE)

Now in 3NF.

(14) $A \to FC$
$C \to D$
$B \to E$

— AB — C.k.

Prime = AB

Non prime = C, D, E, f

$A \to fc$ } P.D ∴ Not in 2NF
$B \to E$ }

- conversion

$R_1$ (AFCD)

$R_2$ (BE)

$R_3$ (AB)

Now in 2NF

$C \to D$ — T.D ∴ Not in 3NF

- conversion

$R_1$ (AFC)

$R_2$ (CD)

$R_3$ (BE)

$R_4$ (AB)

(15) $AB \to C$
$BD \to EF$
$AD \to GH$
$A \to I$
same as (6) $H \to J$

17/08/2023

* BCNF : Boyce-Codd Normal form:

(1) Must in 3NF

(2) All determinats are keys

Note: IF any dependency do not follow BCNF RULe then place RHS attributes of that dependency in

separate table along with copy of LHS attribute then remove their attribute from 3NF

① $A \to BC$

$ABE \to CDGH$

$C \to GD$

$D \to G$

$E \to F$

upto 3NF

$R_1$ (AB C)

$R_2$ (D G)

$R_3$ (C D)

$R_4$ (E F)

$R_5$ (A E H)

All determinants are keys ∴ It is in BCNF.

② $AB \to CEFGH$

$A \to D$

$F \to G$

$FB \to H$

$HBC \to ADEFG$

$FBC \to ADE$

$AB^+ = ABCEFGHD$ —— C.K

Prime = A, B

Non prime = C, D, E, F, G, H

$A \to D$ —— P.D Not in 2NF

Convert

∴ $R_1$ (A D)

$R_2$ (AB CEFGH)

$F \to G$ — T.D — Not in 3NF

Page No.: Date: YOUVA

· Convert

$R_1$ (EG)
$R_2$ (AD)
$R_3$ (ABCEFHB)
Now in 3NF.

$FB \to H$
$FBC \to ADEFG$ } check are they keys or not
$HBC \to ADE$

$FB^+ = FBH$
$HBC^+ = ADEHBCFG$ — } Both are keys.
$FBC^+ = FBCADEG$ —

But $FB^+$ is not key ∴ not in BCNF

· Convert

$R_1$ (FBH)
$R_2$ (FG)
$R_3$ (AD)
$R_4$ (ABCE)
Now in BCNF.

(3) $BCD \to A$
$BC \to E$
$A \to F$
$F \to G$
$C \to D$
$A \to G$

— $(BC)^+ = BCEDAFG$ — CK
Prime = B, C)
Non prime = A, D, E, F, G

• As BCD exist go for canonical

Hide D

$BC^+ = BCDEFGA$ ✓

without Hide

$BC^+ = BCADEF$ ✓

∴ D can be removed

$BC \to A$
$BC \to E$
$A \to F$
$F \to G$
$C \to D$
$A \to G$

go again for canonical

| | | with | without | |
|---|---|---|---|---|
| ∴ | $BC^+$ = | BCEDAFGH | BCED | ✓ |
| | $BC^+$ = | BCEDAFGH | BCAFGD | ✓ |
| | $A^+$ = | AGF | AG | ✓ |
| | $A^+$ = | FG | F | ✓ |
| | $C^+$ = | CD | C | ✓ |
| ignore | $A^+$ = | AGF | AFG | ✗ |

$BC \to A$
$BC \to E$
$A \to F$
$F \to G$
$C \to D$

Key = BC

$C \to D$ ∴ P.D — go for removal

• convert

$R_1$ (CD)

$R_2$ (BCAEFG)

Now in 2NF

$A \to F$
$F \to G$
} T.D ∴ Not in 3NF

Convert:

$R_1$ (AF)

$R_2$ (EG)

$R_3$ (CD)

$R_4$ (BCAEF)

Now in 3NF

Here all determinants are keys ∴ It is in BCNF.

18/08/2023:

(4) Student, Teacher → Subject

Student, Subject → Teacher

Teacher → Subject

Student = A

Teacher = B

Subject = C

AB → C

AC → B

B → C

already in 3NF

AB or AC are CK.

$R_1$ (AB)

$R_2$ (BC) } BCNF conversion.

$R_1$ ( Student, Teacher )

$R_2$ ( Teacher Subject )

(5) BC → D

AC → BE

B → E

F → G

AC — is key.

Prime = A, C

Non prime = B, D, E

→ Already in 1NC

→ B → E Transitivity

remove

$R_1$ (B E)

$R_2$ (A C B D)

→ BC+ = BCDE } Not a key

B+ = BE

∴ $R_1$ (B C D)

$R_2$ (B E)

$R_3$ (A C)

(5) AB → C $R_1$ (ABC)

C → B

AB → B

D → C

→ AB = ABC — key

prime = AB

Non prime = C

Already in 3NF

C → B Not a transitivity

Remove $R_1$ (C B) } already BCNF

$R_2$ (A B C)

(7) R (ABCD)

A = ABCD — P.k

Transitive Dependency:

It exist only when the P.k is not a candidate key for the Rn

- A table is in 3NF if and only if

$x \rightarrow A$

where x is key

A is prime

Prime = D, C

Non prime = B, A

g] Roll_no → Name

Roll_no → Fee_street

Name → Fee

$\left.\begin{array}{l} A \rightarrow B \\ A \rightarrow C \\ B \rightarrow D \end{array}\right\}$

A+ = ABCD

prime = A

non prime = B, C, D

R1 (ABCD) —— 2NF

3NF

R1 (AB(C) } 3NF

R2 (BD)

10] stud_NO → Name

stud_NO → state

state → country

stud_NO → age

A → B

A → C

C → D

A → E

→ A+ = ABCDE

prime = A

Non prime = B C D E

2NF

R1 (ABCDE)

25/08

M T W T F S S
Page No.:
Date:
YOUVA

## SQL :

- Structured Query language
- Used for data retrival

between, In → are inclusive

## Examples :

1] select * from student where year ≠ 4;

2] select * from student where year <= 3;

3] select * from student where year NOT IN(4);

4] select * from student where year IN(1,2,3);

## Query based on Pattern :

1) Select * from student where name LIKE '%e%';
→ Before & After e there can be any thing.

2) ——— " ——— LIKE 'S_a%u';
→ one char b/w S and a, anything in between a and u, ending with u

## Aggregate Functions :

- max (marks)
- avg (marks)

select avg(name) from student;
→ Invalid

select avg(mark1, mark2) from student;
⇒ gives avg of two columns

select roll_no, max(marks) from student;
⇒ don't use aggregate with non aggregate
∴ Invalid query.

select * from student where marks = max(marks)
⇒ Invalid query.

select count(names) from student
⇒ All name — but not null

select sum(marks)/count(marks) from student;
⇒ returns avg.

select roll_no, count(marks) from student
group by roll_no.

⇒

| roll_no | count(marks) |
|---|---|
| 101 | 3 |
| 102 | 4 |
| 103 | 5 |

29|08|2023

Aggregate Function:
- works on single column and produce single column o/p
- can't be used in where clause & Group by clause and order by clause
- Frequently used in having & Select clause.

① select br_name, count(roll_no) from student
Group by br_name;

– 

| br_name | count(roll_no) |
|---|---|
| CSE | 10 |
| IT | 20 |

* If count(*) → It will count all rows including NULL and duplications.

GROUP BY : Returns values by grouping gender, Year

② Select br_name, count(roll_no) from student
GROUP BY br_name, gender, year

| br_name | gender | year | count(roll_no) |
|---|---|---|---|
| CSE | M | first | 2 |
| CSE | F | first | 1 |
| CSE | M | second | 1 |
| CSE | F | second | 2 |
| : | : | : | : |

③ SELECT br_name, gender, count(roll_no)
from student
where gender = 'f'
GROUP by br_name, year
having count(roll_no) ≥ 20;

– rows for each combination of branch and year where count of female student ≥ 20.

Where clause -
- Filters rows
- Aggregation not supported
- No alternative for where clause.

GROUP BY CLAUSE:
- Filters groups
- Aggregation is supported
- Alternatives are available.

ORDER BY:
① Select * from students order by roll_no;
- ascending order.
② select roll_no, count(roll_no) from student order by roll_no where br_name = "IT";
- Error!

Sub query:
▶ uncorrelated
Select * from student where marks < (select avg(marks) from student)
outer query (runs n times)
Inner query (runs only once)

▶ Correlated query:
Inner query uses outer query variable and Inner query runs as many as times Values in outer query.

find operators works for inner query
don't work for inner query.

11/08/2025

M T W T F S S

Page No.:

Date:

YOUVA

→ The following operators can't be use betn inner and outer query

① BETWEEN & NOT BETWEEN
② LIKE & NOT LIKE

→ The following operators alone must be used

① IN & NOT IN
② ANY, All, SOME, etc.
③ EXSITS & NOT EXISTS.

i] Select * from student s1 where 3 = (select count(s2.marks) from student s2 where s1.marks ≤ s2.marks)

→ 3rd Highest marks.

2
3 = 3

→

ii] select * From student where roll_no IN (Select distinct roll_no from library)

→

iii] select * from student where city IN (Select city where city = "Delhi" or city = "Sangli")

→

iv] Select cid from order
where (cid) IN
(select cid from agent where city IN
('sangli', 'Delhi'))

→ will return cid with city as sangli and Delhi

Inner = Sangli, Delhi ⟶ CID from agent

outer = ↳ CID from order.

v] select * from agent where percentage (>) Any
(select percentage from agent)

→

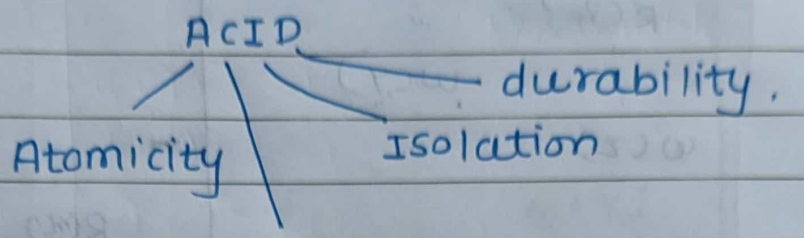
vi] select * from agent where percentage = 6;

→ {

vii] select * from agent where percentage =
(select min(percentage) from agent)

05/09/2023
15 marks

▷ Transaction :




Conflicts :

1. Read Write
2. write Read
3. ~~Read~~ Write Write

Need of transaction –

if transaction control mechanism not implemented it results into inconsistent state, interference from various user & ambiguity for making the permanent change.

* write - Read problem (dirty Read)

| $T_1$ | $T_2$ |
|---|---|
| R(A) | |
| W(A) | |
| | R(A) |
| | W(A) |
| fail() | |

i.e transaction reads the data written by other transaction before commiting.

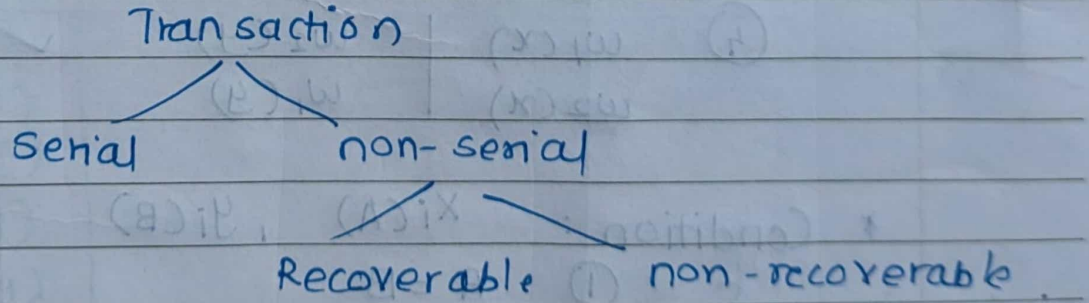
* Read - Write problem

| $T_1$ | $T_2$ | |
|---|---|---|
| R(A) | | Transaction T2 is writing data which is previously read by T1. |
| | R(A) | |
| | W(A) | |
| | Commit | |
| R(A) | | |
| W(A) | | |
| ↳ fail() | | |

* Write - Write problem (Blind write)

| $T_1$ | $T_2$ | |
|---|---|---|
| W(A) | | T2 writes data which is already written by T1. ∴ T2 overwrites the data written by T1. |
| | W(B) | |
| W(B) | | |
| Comm. | | |
| | W(A) | |
| | Commit | |

12/09




(S1)

| $T_1$ | $T_2$ | |
|---|---|---|
| R(x) | R(x) | non-recoverable. |
| W(x) | | |
| | W(x) | |
| | R(x) | |
| | C2 | |

X

(S2)

| $T_1$ | $T_2$ |
|---|---|
| R(x) | |
| | R(y) |
| W(x) | |
| | W(y) |
| | R(x) |
| $a_1$ | |
| | C2 |

Recoverable

Roll Back

- A schedule S with n transaction is said to be serializable if it is equivalent to schedule S1 with a same n transaction

  where S is non serial
  S1 is serial

| | | |
|---|---|---|
| ① | $R_1(x)$ | $W_2(x)$ |
| | $W_2(x)$ | $R_1(x)$ |
| ② | $W_1(x)$ | $W_2(x)$ |
| | $W_2(x)$ | $W_1(x)$ |
| ③ | $R_1(x)$ | $R_2(x)$ ✓ |
| | $R_2(x)$ | $R_1(x)$ |
| ④ | $W_1(x)$ | $W_2(y)$ ✓ |
| | $W_2(x)$ | $W_1(y)$ |

* Condition : $X_i(A)$ , $Y_i(B)$

then it is conflict pair

① A = B
② $i \neq j$
③ x or y must be write operation

① 

| $T_1$ | $T_2$ |
|---|---|
| R(x) | |
| W(x) | → conflict |
| | R(x) |
| | W(x) |
| R(y) | → non-conflict |
| W(y) | → conflict |
| | R(y) |
| | W(y) |

⇒ ① ↓

| $T_1$ | $T_2$ |
|---|---|
| R(x) | |
| W(x) | |
| | R(x) → non conflict |
| R(y) | |
| | W(x) → non-conflict |
| W(y) | |
| | R(y) |
| | W(y) |

② ↓

| $T_1$ | $T_2$ |
|---|---|
| R(x) | |
| W(x) | |
| R(y) | |
| W(y) | |
| | R(x) |
| | W(x) |
| | R(y) |
| | W(y) |

∴ Serializable.

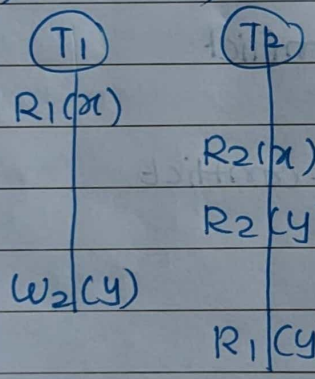
(2) S1 : $R_1(x)$ $R_1(y)$ $R_2(x)$ $R_2(y)$ $W_2(y)$ $W_1(x)$

| $T_1$ | $T_2$ |
|---|---|
| $R_1(x)$ | |
| $R_1(y)$ | |
| | $R_2(x)$ |
| | $R_2(y)$ |
| | $W_2(y)$ |
| $W_1(x)$ | |

can't be serializable

15/09

(1) $R_1(x)$ $R_2(x)$ $R_2(y)$ $W_2(y)$ $R_1(y)$ $W_1(x)$




→

| $T_1$ | $T_2$ |
|---|---|
| $R_1(x)$ | |
| | $R_2(x)$ |
| | $R_2(y)$ |
| | $W_2(y)$ |
| $R_1(y)$ | |
| $W_1(x)$ | |

→

| $T_1$ | $T_2$ |
|---|---|
| | $R_2(x)$ |
| | $R_2(y)$ |
| | $W_2(y)$ |
| $R_1(x)$ | |
| $R_1(y)$ | |
| $W_1(x)$ | |

∴ serializable.

(2)

| T1 | T2 |
|---|---|
| | $R_2(A)$ |
| | $W_2(A)$ |
| $R_1(A)$ | conflict |
| $W_1(A)$ | |
| $R_1(B)$ | |
| $W_1(B)$ | |
| conflic | $R_2(B)$ |
| | $W_2(B)$ |

$\therefore$ not possible

(3)

| $T_1$ | $T_2$ | $T_3$ |
|---|---|---|
| $R(x)$ | | |
| | | $R(y)$ |
| | | $R(x)$ |
| | $R(y)$ | |
| | $R(z)$ | |
| | | $W(y)$ |
| | $W(z)$ | |
| $R(z)$ | | |
| $W(x)$ | | |
| $W(z)$ | | |