Functional Dependency (FD)

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.

$$X \rightarrow Y$$

determinant Dependent

→ It is a tool for normalization

Advantage

→ Through the identification and removal of redundant or unneeded data.

→ By guaranteeing that data is correct and consistent.

* Describe. Data is dependent on FD,

FD is not dependent on Data.

<u>explain:</u>

FD can exist independently, while data cannot exist or be meaningful without FD.

imagin FD is a rule or Function and Data is input.

→ FD: A Function $y=2x+3$

→ Date: the value of x or input

FD has meaning the relation between x an y but the data (1, 2, 3) are meaningless.

FD is independent.

Data is dependent.

FD

Trivial

$A \to A$

$AB \to BA$

$B \to B$

if B is subset of A

Non-trivial

$AB \to ABZ$

$A \to A$

$B \to B$

$A \to B$

$B \to C$

B is not subset of A

x, y is a set of attributes

$x \to y$

determinate   dependent

R

x	y
1	1
2	1
3	2
4	3
2	5
2	5 1

$t_1 \to$ (row 2), $t_2 \to$ (last row)

$x = 2$ $y = 1$

must be 1

y

$1 \neq 5$

if $t_1 x = t_2 x$

then $t_1 y = t_2 y$

Example

R.No	Name	marks	Dept	course
1	a	78	CS	c_1
2	b	60	ECE	c_1
3	a	78	CS	c_2
4	b	60	ECE	c_3

FD: $x \rightarrow y$

if $t_1 x = t_2 x$

then $t_1 y = t_2 y$

R.NO $\rightarrow$ Name ✓ ($x \rightarrow y$)

Name $\rightarrow$ R.No ✗

R.NO $\rightarrow$ marks ✓

Dept $\rightarrow$ course ✗

course $\rightarrow$ Dept ✗

$\boxed{\text{marks} \rightarrow \text{Dept}}$ ✓

(R.No, name) $\rightarrow$ marks ✓

name $\rightarrow$ marks ✓

(name, mark) $\rightarrow$ dept ✓

(Name, marks) $\rightarrow$ (dept, course) ✗

Closure of a Set of Functional Dependencies

It means the complete set of all possible attributes that can be functionally derived from given functional dependency using the Inference rules known as Armstrong's Rule.

→ It denoted by F^+

InFerence Rule (IR)

① Reflexive Rule

if $y \subseteq x$ then $x \rightarrow y$

$x = \{a, b, c, d, e\}$

$y = \{a, b, c\}$

② Augmentation Rule

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

For R(ABCD)

if $A \rightarrow B$ then $AC \rightarrow BC$

(iii) Transitive Rule

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

(iv) Union Rule

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

(v) Decomposition Rule

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

closure set

$X^+ \rightarrow$ contains set of attributes determined by a

Exercise

R(ABCDEFG)

$A \rightarrow B$

$BC \rightarrow DE$

$AEG \rightarrow G$

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

R(ABCDE)

- $A \to BC$
- $CD \to E$
- $B \to D$
- $E \to A$

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

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

B

R(ABCDEFGH)

- $A \to BC$
- $CD \to E$
- $E \to C$
- $D \to AEH$
- $ABH \to BD$
- $DH \to BC$

$(BCD)^+ = \{B, C, D, E, A, H\}$

Irreducable set of FD (canonical cover)

F^c if F^c don't have
(i) redundant attribute
(ii) redundent FD.

Steps :-

$F : \{AB \to C, C \to AB, B \to C, ABC \to AC, A \to C, AC \to B\}$

$S-1 = \{AB \to C, C \to A, C \to B, B \to C, ABC \to A, ABC \to C, A \to C, AC \to B\}$

redundant

$S-2 = \{\cancel{B \to C}, C \to A, C \to B, B \to C, A \to C, A \to B\}$

$S-3 = \{C \to A, \cancel{A \to C}, A \to B\}, B \to C\}$

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

$B = \{B$

$C = \{C, B\}$

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

redundant

$B = \{B\}$ $A = \{A, B\}$

$C = \{C, B\}$ $A \{AC$

Find Key

super keys : set of attributes whose closure contain of all attributes of given Relation

candidate key

candidate keys are super keys with the least number od attributes.

* Any CK can not be proper subset of Any other super keys

Like $A^+ = \{A, B, C, D\} \longrightarrow SK, CK$

$AD^+ = \{A, B, C, D\} \longrightarrow SK$ but not CK

$\{A\}\ \{D\}$

R(ABCD)

FD: $\{A \to B, B \to C, C \to A\}$

$ABCD^+ = \{A, B, C, D\} \to SK$

$ACD^+ = \{A, C, D, B\} \to SK$

$AD^+ = \{A, D, B, C\} \to SK$

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

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

CK

BD, CD

Prime Attribute

$\{A, D\}$

R(ABCDEF)

FD: $\{AB \to C, C \to D, B \to AE\}$

$AB \to D$

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

$ABDEF = \{A, B, D, E, F, C\} \to SK$

$ABEF = \{A, B, F, C, D, E\} \to SK$

$BF = \{B, A, E, C, D, F\} \to SK$

?

$R = (ABCDEF)$ $C \to D$, $C \to E$

$F = \{AB \to C,\ C \to DE,\ E \to F,\ D \to A,\ C \to B\}$

$AB \to DE$

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

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

$AB\cancel{F}^{+} = \{A\ B\ F\ C\ DE\}$

$AB = \{A, B, C, DE, F\} \to CK$

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

$\{B\} = \{B\}$

$AB \to C$
$C \to E$
$E \to F$
$\therefore AB \to F$

must | non must
F

Prime Attribute

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

AB

$DB \to CK$

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

$\{B\} = \{B\}$

$CK \Rightarrow AB\ \ DB\ \ C$

$AC \to$ not candidate key

$C^{+} = \{DE\ C\ A, B\ C\} \to CK$

NORmalization

Anomaly

NORmalization is a technique to reduce data redundancy from a table.

• what is data redundancy ?

→ Repetition of similar data at multiple places.

• Repetition of data increases the size of database so we reduce redundancy.

others problem

- Insertion problem
- Deletion
- Updation

student table.

Roll	name	age	dept	hod
01	A	22	CSE	ME
02	B	23	CSE	ME
03	C	~~24~~ 22	CSE	ME
04	D	23	CSE	ME

TO insert redundant data for every new row is a data insertion ANOMALY

Now delete Row one by one delete student info. also dept. info.

* Loss of related dataset when some other dataset is deleted called deletion Anomaly

* Updation Anomaly

Student table info

Student table

Roll	Age	name	dept
01	22	A	CSE
02	23	B	CSE

dept. table

dept	name
CSE	ME

- Insertion
- updation
- deletion

A) 1st NF

→ Table should not contain any multivalued Attribute

Student table

Roll	Name	course
1	Abir	C++, Java
2	Kabir	DBMS
3	Nibir	C+, DSA

Solution : each attribute of a table must have atomic (single) values.

P1)

Roll	Name	course
1	Abir	C++
1	Abir	Java
2	Kabir	DBMS
3	Nibir	C++
3	Nibir	DSA

OR

P2)

Roll	Name
1	Abir
2	Kabir
3	Nibir

Roll	course
1	C++
1	Java
2	DBMS
3	C++
3	DSA

Roll + course

composit key

8, 256,

2NF

HP elit book 840 G3

→ Table is in 1NF

→ can't partial dependency

250-
10-
10
60
50
20
10
50

partial dependency

Partial dependency is a situation in which a non-key attribute of a table depends on only a part of the primary key.

composit key

A	B	C	D

Std Id	course Id	cours name	Inst.
1	101	Math	X
1	102	Eng	Y
2	101	His	Z

AB → D

B → D

Here (Std Id + cours Id) is composit primary key.

course name → Std. Id

cours name depends on part of the primary key Std. Id.

⊛ How to remove partial dependency

~~Stt~~

Stud-id	course-id	course fee
1	c_1	1000
2	c_2	1500
1	c_4	2000
4	c_3	1000
~~2~~4	~~c3~~ c_1	1000
2	c_5	2000

Stud-id + course-id ⇒ candidate key

course fee ⇒ non-prime attribute.

non-prime attribute is dependent on ~~non-pr~~ candidate key.

so This table conver 2NF, we need split this table in two table.

Stud-id	course-id

course-id	course fee

R(ABCDEF)

→ NOT in table

L.H.S proper subset of CK and

R.H.S non-prime attribute

FD: {proper subset of CK

$C \to F$ → pd

$E \to A$ → Non prime attribute

$EC \to D$ → pd

$A \to B$ ✗

subset

not proper subset

$EC = FADB$

$EC^+ = \{ECFADB\}$

(i) $CK = \{EC\}$

(ii) prime attributes $\{E, C\}$

(iii) NON prime $\{A, B, D F\}$

3NF ⟺ partial ← Dependency → full, → transitive

Transitive dependency :- If the value of a non-primary attribute can be defined using another non-primary attribute then it is called transitive dependency

Rules

- Table must be in 2NF
- Transitive functional dependency of non-pri attribute on any super key should be removed.

Employee table

emp_id	emp_name	emp_zip	emp_state	emp_city
101	A	1200	UK	nodia
102	B	2100	USA	chicago
103	C	8400	UP	Bhopal
104	D	9700	US	Norwich

candidate key = Emp_Id

~~Non~~ prime attribute = Emp-id

Non prime attribue = anothers key

Here emp_state and emp_city dependent on emp_zip.

and emp_zip dependent on emp_id

empid	emp name	emp_zip

emp -zip	emp -state	emp_city

Ex

R(ABCD)

FD: $AB \rightarrow CD,\ D \rightarrow A$ R.S

CK: $AB^+ = ABCD$ $\Rightarrow$ CK: {AB, DB}

$DB^+ = DBAC$

prime Attributes = {A, B, D}

Non prime Attribute {C}

valid for 3NF

For each FD :

LHS must be a ck or SK ~~or~~

RHS is a prime attribute

Here AB → CD

ck

D → A

prime

So this table is in 3NF

~~# BNCF~~ BCNF

α

Non prime attribute ⟷ prime attribute

super key convert

- ~~must have s~~
- Table must be in 3NF
- A table is in BCNF if every functional dependency $x \to y$, x is the super key of the table

• for BCNF, the table should be in 3NF and for every FD, LHS is super key

$\underset{\text{must be super key}}{x} \rightarrow y$

Ex

$R(A, B, C)$

FD: $\{\underset{SK}{A} \rightarrow B,\ \underset{SK}{B} \rightarrow C,\ \underset{SK}{C} \rightarrow A\}$ → it is 3NF

super key
$ABC^+ = \{ABC\}$
$AC^+ = \{ABC\}$
$A^+ = \{ABC\}$
$C^+ = \{ABC\}$
$B^+ = \{ABC\}$

$CK = \{A, C, B\}$

BCNF: ✓

valid for ...

Lec 24 Jennys

Example

Normalize a relation from 3NF to BCNF

R(ABCDEFGH)

FD: {

$A \rightarrow BD$

$B \rightarrow C$

$E \rightarrow FG$

$AE \rightarrow H$

}

Find CK

$\cancel{ABCDEFGH}^+ = \{ABCDEFGH$

SK ← $AE^+ = \{AEHBDCFG\}$

proper sub set

$A^+ = \{BDCA\}$

CK

$E^+ = \{EFG\}$

∴ prime attribute = {A, E}

non prime Attribute {BCDFGH}

Find PD partial dependency

L.H.S proper subset of CK and

RHS Non. prime attribute.

$A \rightarrow BD \implies PD$

$B \rightarrow C \implies \times$

$E \rightarrow FG \implies PD$

$AE \rightarrow H \implies \cancel{PD}\ \times$

subset

so this table now 1NF

Now solve the Pd

$A^+ = ABCD$

$R_1 = \{A, B, C, D\}$

R_3

$A^+ = ABDC$

$A \rightarrow BDC$

$B^+ = BC$

$B \rightarrow C$

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

$\cancel{D^+ = D}$

$E^+ = EFG$

$R_2 = \{E, F, G\}$

R_3

$E^+ = EFG$

$E \rightarrow FG$

$\cancel{F^+ = F}\ \times$

$\cancel{G^+ = G}\ \times$

$\cancel{R_3 = \{H\}}$

$R_3 = \{H, A, E\}$

$\cancel{H^+ = H}\ \times$

$A^+ = ABDC\ \times$

$\cancel{E^+ = EFG}\ \times$

$AE^+ = AE\,BDCFGH$

$AE \rightarrow H$

$AH^+ = AHBDC\ \times$

$FD_1: \{ \overset{CK}{A} \to BDC$

$B \to C \to TD$

$\}$

CK: A

$FD_2: \{$

$E \to FG$

Superkey $\}$

CK: E

BCNF

$FD_3: \{$

$AE \to H$

SK $\}$

CK: AE

BCNF

$B^+ = BC$

$R_{11} = \{B, C\}$

$B^+ = BC$

$B \to C$ BCNF

$C^+ = C$ $\times$

$R_{12} = \{A, D, B\}$

$A^+ = \cancel{A}BD\cancel{C}$ $\times$

$A \to BD$ BCNF

$B^+ = \cancel{B}\cancel{C}$ $\times$

$D^+ = D$ $\times$

$R(ABCDEFGH)$

$FD: \{A \to BD, B \to C, E \to FG, AE \to H\}$

$R_1(A, B, C, D)$ $FD_1: \{A \to BDC, B \to C\}$

$R_{11}(B, C)$ $FD_{11}: \{B \to C\}$ BCNF

$R_{12}(ADB)$ $FD_{12}: \{A \to BD\}$ BCNF

$R_2(E, F, G)$ $FD_2: \{E \to FG\}$ BCNF

$R_3(AEH)$ $FD_3: \{AE \to H\}$ BCNF

'19-20 6(a)

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

P(f, g, h, i)

a → b

b → c

b → e

b → d

f → g

P { f → h

h → i }

accessionno = a

isbn = b

title = c

author = d

publisher = e

user id = f

name = g

deptid = h

deptname = i

R = books

P = users

1st NF: give schema is already 1NF because all attributes contain atomic values.

2NF: To achive 2NF there should bo partial dependency in FD.

in R(a,b,c,d,e) find ck

$a\cancel{b}\cancel{c}\cancel{d}\cancel{e}^{+} = \{abcde\}$

so candidate key is A

prime attribute = {A}

Non prime attribute {B, C, D, E}

and, in P(F, G, H, I) fin CK

$F\cancel{G}\cancel{H}\cancel{I}^{+} = \{FGHI\}$

CK = F

PA = {F}

NPA = {GHI}

find pd in both tables

A → B → ~~E~~d	F → G
B → C	F → H
B → E	H → I
B → D	
no partial dependencies	No partial dependencies

So all relation have 2NF

3NF — TO achive 3NF all transitive dependencies should be removed that means non prime attributes can not determine non prime attribute

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

$A \rightarrow B$ → only valid

$B \rightarrow C$

$B \rightarrow E$

$B \rightarrow D$

} → TD

$R_1(A, B)$

$A^+ = AB$

$A \rightarrow B$

BCNF

$B^+ = B\cancel{C}DEK$

$FD_{R_1}: \{A \rightarrow B\}$

BCNF

valid

$R_2(B, C, D, E)$

$B^+ = BCDE$

$B \rightarrow CDE$

$C^+ \Rightarrow C\alpha$

$D^+ = D\alpha$

$E = E\alpha$

$FD_{R_2}: \{B \rightarrow CDE$

BCNF

$P(F, G, H)$

$F \rightarrow G$

$F \rightarrow H$

$H \rightarrow I$ → TD

$P_1(H, I)$

$H^+ = HI$

$H \rightarrow I$

$I^+ = I\alpha$

$FD_{P_1}: \{H \rightarrow I\}$

BCNF

$P_2(F, G, H)$

$F^+ = FGHI$

$F \rightarrow GH$

$G = G\alpha$

$H = HI\alpha$

$FD_{P_2}: \{F \rightarrow GH\}$

BCNF

Transaction

A Transaction is a logical unit of work. It is the set of operations (read, write) to perform unit to work.

Operations of Transaction

① Read (x): A read operation is used to read the value of x from the DB and store in the buffer in the main memory for further actions.

② write (x): A write operation ~~to~~ is used to write the value of the db from the buffer in the main memory

To withdraw mony from Id first read() operation then write() operation.

Balance

Id	name	Balance
01	x	500
02	y	1000

COMMIT is a transaction control language that is used to permanently save the changes don in the transaction in db.

RollBack: is a transaction control language that is used undo the transaction that have not been saved in the db

employee

Id	name	salary
1	Rahim	20000
2	Karim	30000

start transaction;
update employee
set salary = 50000
where id = 3;

COMMIT; / ROLLBACK → Return before state

→ successfully parmanently save in table

Save point :- This is used to set a point within a transaction to which you can later rollback if needed.

SAVEPOINT savepoint_name;
ROLLBACK TO savepoint-name;

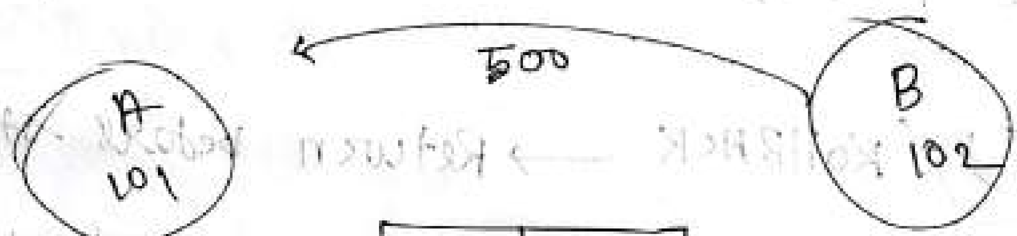
→ ~~Autom~~ Atomicity: The entire transaction takes place at once or doesn't happen at all

→ consistency: The database must be consistent before and after the transaction

ACID

→ Isolation: multiple transaction occure independently without interference

→ Durability: The changes of successfully transaction occurs even if the system failure occure. (commit)



1000+500
=1500

id	balance
101	1000
102	2000

2000 − 500
=1500

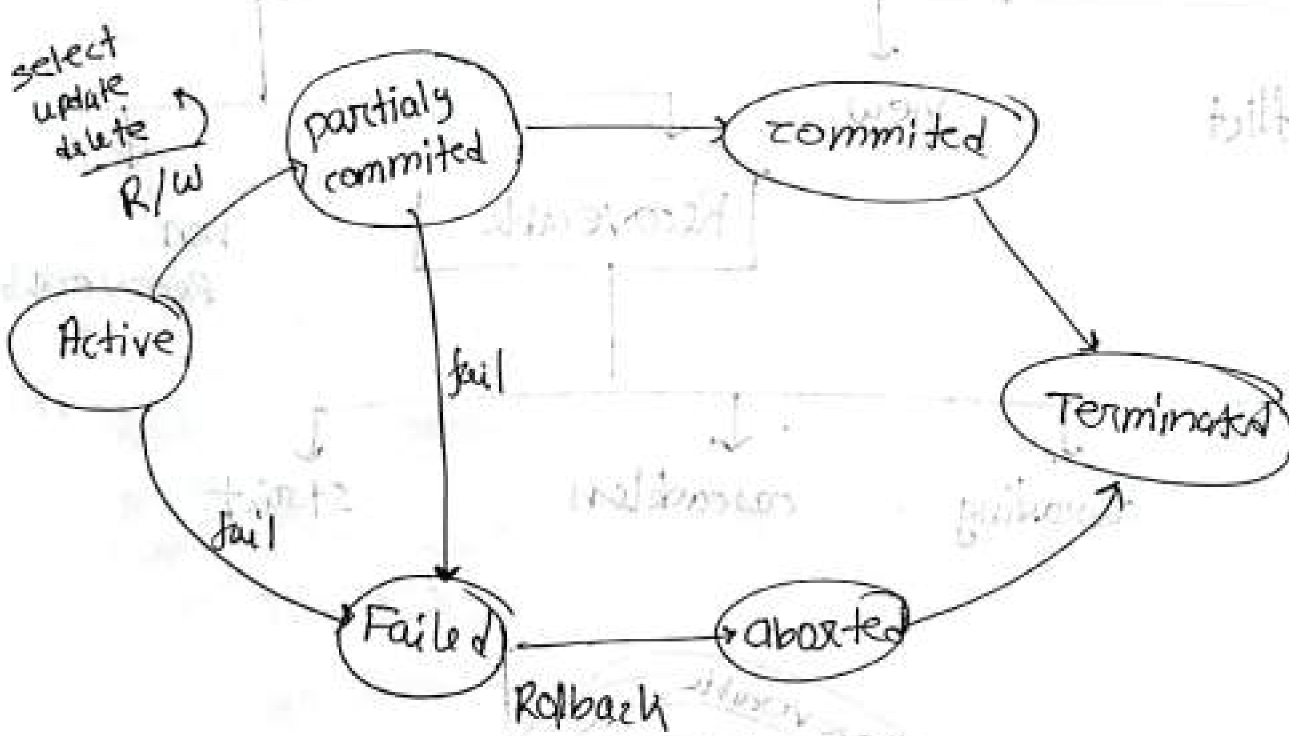
before — Transaction — after

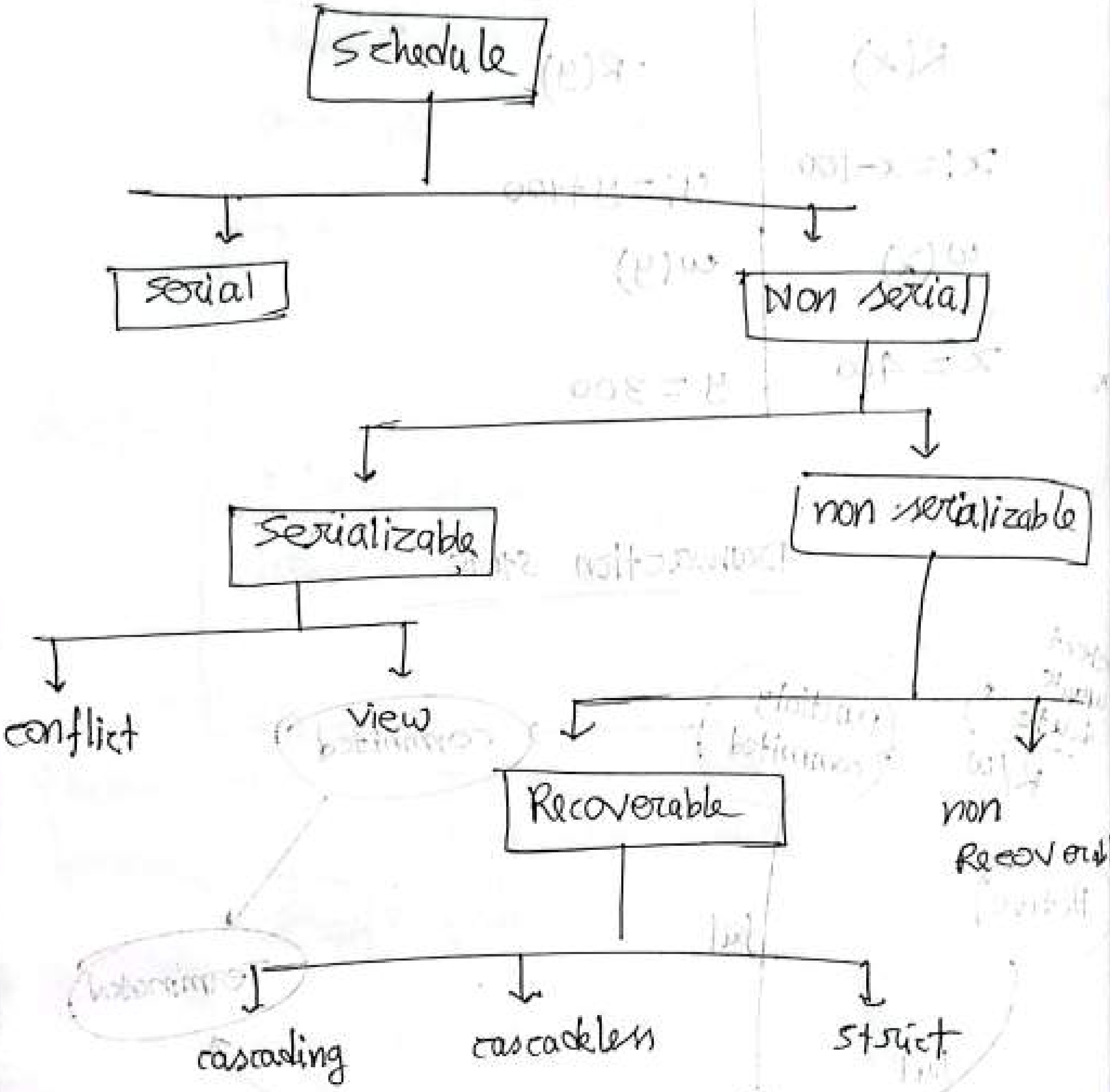
101 → 1000 } = 3000
102 → 2000

101 → 1500 } = 3000
102 → 1500

	T₁	T₂	
before $x=500$			$y=200$
	$R(x)$	$R(y)$	
	$x:=x-100$	$y:=y+100$	
	$\omega(x)$	$\omega(y)$	
After	$x=400$	$y=300$	

Transaction state




Schedule
serial
Non serial
Serializable
non serializable
conflict
view
Recoverable
non Recoverable
cascading
cascadeless
strict


recoverable
cascadeless
strict

multi user support: concurrency

performance and speed: parallel

Advantages of concurrency

- decrease waiting time
- decrease responce time
- Resource utilization
- Increase efficiency

problems with concurrency

parallel + transaction

Lost update problems (w - w conflict)

T_1: R(A), A = A+200, w(A)

A=1000

T_2: R(A), A = A-700, w(A)

	T_1		T_2	
1200	R(A)	t_1		
1200	A=A+200	t_2		
		t_3	R(A)	1000
		t_4	A=A-700	300
1200	w(A)	t_5		
		t_6	w(A)	300

blind write

buffer

② unrepeatable read problem

R-W ~~conflict~~

T1	T2	same dB
1000 R(A)		A = 1000
	R(A) 1000	
	A = A − 200 800	
	W(A) 800	
800 R(A)		

800 — problem

T1 depends on T2

③ Dirty read problem

W-R conflict

T1	T2	A = ~~1000~~ ~~1500~~ 2000
1000 ← R(A)		
← A = A+500		
1500 ← W(A)	R(A) – 1500	
	A = A + 500	
	W(A) 2000	
Rollback R(B)		

A 1000 but A = 2000 present

failure

Serial schedule

T_1	T_2
R(A)	
A:=A-500	
W(A)	
commit	
	R(A)
	A:=A+1000
	W(A)
	commit

Non serial schedule.

T_1	T_2
R(A)	
A:=A-500	
W(A)	
	R(A)
	A:=A+500
R(A)	
W(A)	

Recoverable → cascading
→ cascadeless
→ Strict

If some transaction T_j is reading or written be some other transaction T_i, then the commit of T_j must occure after the T_i commit.

② means if T_2 is dependent of T_1 then first commit of T_1 then T_2 commit

T_1	T_2	A = 1000
1000 R(A)		
R = A+500		
1500 W(A)		
	R(A) 1500	
	R = A + 500	
	W(A) 2000	
commit		
	commit	

NK cascading schedule

T_1	T_2	T_3
source R(A)		
A := A+500	depend on T_1	
W(A)	R(A)	
Rollback	A := A + 500	depend on T_2
R(B) failure	W(A)	
commit		R(A)
	Rollback	A := A + 500
	commit	W(A)
		Rollback
		commit

When failure in one T and this leads other T, rollback the source T also dependent T.

② cascadeless schedule

T_1	T_2
R(A)	
W(A)	
	W(A)
commit	
	R(A)
	W(A)
	commit

when a transaction is not allowed to read data until the last transaction which ~~to~~ has written is commited

mean: if a T, read another T written data ensure that this written data is commited.

Here T2 read updated data of T1 if only T1 is commited

③ strict schedule

T_1	T_2
R(A)	
	R(A)
W(A)	
commit	
	W(A)
	R(A)
	commit

in strict schedule Read and write updated value of another T must ~~be~~ have commited.

lec:99 Farhan

Serializability schedule

Conflict :-
(i) can't loop/cycle
(ii) consistency of serial schedule

T1	T2	T3
→ R(A)		
	→ R(B) 20	
		→ R(B) 20
	→ W(B) 30	
→ W(A) 20		
		→ W(A) 30
	30 → R(A)	
	→ W(A) 40	

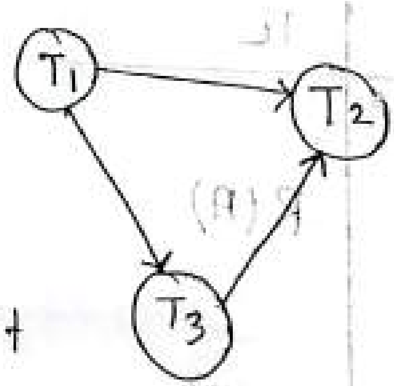
W(A) = 40
W(B) = 30

R — W
W — R
W — W

consistency check (valid)

A = 10, B = 20 } W(A) = A+10, W(B) = B+10

Create a precedency graph

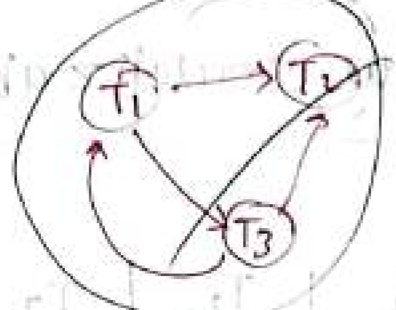


in this graph have not any loop or cycle so it is conflict schedule and it convert serial schedule.

$T_1 \rightarrow T_3 \rightarrow T_2$

serial schedule

T_1	T_3	T_2
10 R(A)		
W(A)		
20	R(B) 20	
	W(A)	
	30	R(B)
		W(B)
		R(A)
		W(A)



non- conflict

- R(A) = 40
- R(B)

W(A) = 40
W(B) = 70

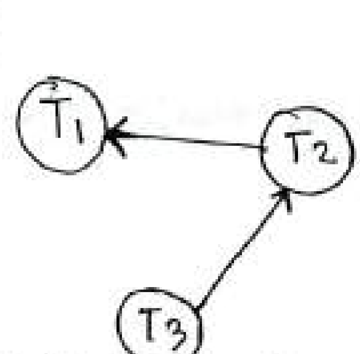
19-20
7c

S: $R_1(A)$; $R_2(A)$; $R_3(B)$; $W_1(A)$; $R_2(C)$; $R_2(B)$; $W_2(B)$; $W_1(C)$;

T_1	T_2	T_3
→ R(A)		
	→ R(A)	
		→ R(B)
→ W(A)		
	→ R(C)	
	→ R(B)	
	→ W(B)	
→ W(C)		

R - W
W - R
W - W

conflict serializability schedule

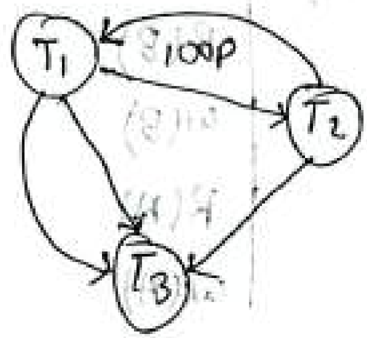


~~in precea~~ in the precendence graph has no cycle so it is conflict

VIEW. serializability

R – W
W – R
W – W

T_1	T_2	T_3
→ R(A)		
	→ W(A)	
→ W(A)		
		W(A)



difficult

18-19
8c

(i) Transaction 1 (Transfer 50 from A to B)

```
START TRANSACTION;
UPDATE account set balance = balance - 50 where
        account_name = 'A';

UPDATE account SET balance = balance + 50 where
        account_name = 'B';

COMMIT;
```

(ii) Transaction 2 (Transfer 10% of balance A to B)

```
START TRANSACTION;
UPDATE account SET balance = balance - (balance * 0.10)
        where account_name = 'A';

UPDATE account SET balance = balance + (balance * .10)
        where account_name = 'B';

COMMIT;
```

18-19
4

ⓐ —

ⓑ ① CK is AD

② ⓔ ⑪ AD⁺ = {AD BC}

ⓒ

BOOK_seller(ISBN_NO, salsman, sold date, comission, discound-amound)

consider that,

Book-seller = R

ISBN-NO = A

salsman = B.

sold-date = C

comission = D

discount = E

F. Dependencies

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

$A \rightarrow C$

$A \rightarrow E$

$C \rightarrow E$

$B \rightarrow D$

Find ck

$AB\cancel{CDE}^{+} = \{ABCDE\}$

$AB^{+} = \{ABCED\}$

CK

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

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

$CK = AB$

$PA = \{A, B\}$

$NPA = \{C, D, E\}$

1NF: all attributes are atomic so this relation is 1NF

2NF: • Relation is in 1NF

• ~~There are no partial dependencies in there~~ Relation

~~so now this relation also 2NF~~

~~2NF~~: check partial dependencies:—

L.H.S is proper subset of ck and

R.H.S is non-prime Attribute.

so partial dependencies relations are

$A \rightarrow C$

$A \rightarrow E$

$B \rightarrow D$

$A^+ = ACE$

$R_1(ACE)$

$A^+ = ACE$

$A \rightarrow CE$ — 2NF

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

$C \rightarrow E \longrightarrow$ Td

$CK = A$

$B^+ = BD$

$R_2(BD)$

$B^+ = BD$

$B \rightarrow D$ BCNF

$C^+ = CE$

$R_{11}(CE)$

$C^+ = CE$

$C \rightarrow E$

BCNF

$R_{12}(AC)$

$A^+ = AC\cancel{E}$

$A \rightarrow C$

BCNF

~~$R_P(A$~~

$R_1(B,D)$

$R_2(C,E)$

$R_3(A,C)$