Normalization

Decompose our table (relation)

R(ABCDE)

R_1(ABC) R_2(CDE)

By decomposing, maintainance cost ↑ces, space ↑ces, ...

By Normalization is very imp. because in our original table, there may be data redundancy which also leading to data inconsistency.

Data Redundancy leads to (1) Data inconsistency.

(2) Modification Anamolies.

↳ (i) Insertion anamoly. (not able to insert)

Student

SID	SName	CID	CName
1	S1	C1	DBMS
2	S2	C2	O.S
3	S3	C1	D.
4	S4	C2	O
5	S5	C1	D
1	S1	C3	DSA

6 S6 cannot be inserted, because CID and CName not known & if CID is primary key in course table, then it can't happen

(2) Deletion anamoly.

if suppose S1 is passed out and wants to be deleted, it can't happen, ∵ here C3 will be deleted entirely, & there will be only 2 courses seen. So if decomposed, then we can see all 3 courses.

(3) Updation anamoly.

Suppose, C2 updated to Adv. O.S. (AOS), then it needs to be updated in each tuple.

Process of decomposing a complex relation into less complex relations.

→ Lossless decomposition :- (must be)

if R → R_1 R_2 & when R_1 & R_2 are merged, it must be same as R, then known as lossless, if doesn't then lossy.

(getting extra tuples)

→ Dependency preserving :- (may or may not be)

→ Need
→ To reduce/eliminate redundancy. (it causes Modification anamolies).

→ To maintain data consistency.

Sid	Sname	Cid	Cname

Suppose. Sid, Sname
Cid, Cname.
while merging there is no common, so it follows cartesian product then
$3 \times 3 = 9$ tuples. (lossy decomp.

Sid, Sname, Cid.
Cid, Cname (✓). [lossless decomposition)

While decomposing,
Define foreign key in old table.
Refer it new table as primary key.

Sid → Sname Cid
Cid → Cname
} This is dependency preserving.
It may or may not be (ϕ no problem).

If Sname → Cname (then no problem)
No dependency preserving.

Disadv:
Maintainance cost ↑ces.
Query processing cost ↑ces.

Normal Forms-
1NF, 2NF, 3NF, BCNF — reduces Redundancy is due to FD's
BCNF - means free from redundancy due to FD's
4NF, 5NF, DKNF, Higher normal forms. ↘ Reduces redundacy due to multivalued FD.

Normal forms are cummulative

□ → 1NF ← C1

1NF ← C1
2NF ← C2
3NF ← C3
BCNF ← C4

if it is in 3NF, then we say that it follows
C1, C2, and also C3
1NF 2NF 3NF.

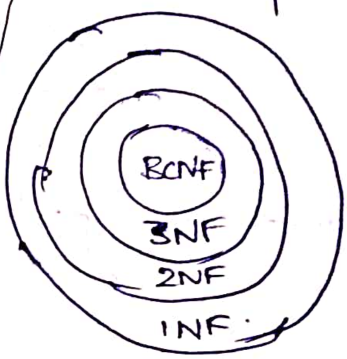
A relation in a given normal form is free from certain set of modification anamolies.

To be in a normal form, a relation must satisfy a condition.

1NF = C1
2NF = 1NF + C2
3NF = 2NF + C3
BCNF = 3NF + C4

C1:- Domain of all attributes must be same.
C2:- No partial FD.
C3:- No transitive FD.
C4:- Every attributes must be dependent on super key only.

(1) No repeating grps.
(2) No multivalued attributes.
(3) No composite attributes.



ABCDF.

Suppose AB is superkey then AB ← C, AB ← D, AB ← E

B ← C

Normalization Process:-

Identify the FD set for relation.
Identify the candidate keys for relation.
Apply the definition

1NF

A relation R is said to be in 1NF, iff.

The domain of each attribute must be atomic.

→ A domain is said to be atomic if the values of the domain are indivisible unit and not set of values.

→ ~~R~~ In other words, repeated groups are not allowed.

2NF :-

No Partial FD's and it is in 1NF

Ex:- R(ABCD) $A \to B$
$C \to D$.

AC is candidate key.

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

Here, A, C are prime attributes (part of candidate key).
B, D are non-prime attributes (not part of candidate key)

if Nonprime ← Prime. then known as Partial FD.
Nonprime depends on the prime.

$AC \to BD$
$B \to D$

ABCDE A is candidate key.

$A \to B$
$B \to C$
$D \to E$.

Partial FD. only comes in composite keys.

ABCD ABC – composite key
A, B, C – Prime
D → Non-prime

if $D \to A$, $D \to B$, $D \to C$, is present then, it is partial FD & Hence not in 2NF.

Mathematically, $X \to Y$ is PD $\Leftrightarrow$ X is C CK $\wedge$ Y : NPA.

First (supplier_no., status, city, part_no, quantity)

FD set:- (supp no, part no) → quantity
supp.no → status.
supp.no → city.
city → status.

R(ABCDEF).

$ABC \to D$, $ABD \to E$, $CD \to F$, $CDF \to B$, $BF \to D$.

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

$ABF = \{A, B, F, D, E,$

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

$AFC = \{A, F, C,$ (x).

$AEC = \{A, C, E. \ldots$

ABC, ADC

A, B, C, D → are prime attributes, & E, F → Non-prime

Here E depends on ABD (not a key)

F " " CD (not a key).

$\times \subset$ CK.

{ABC, ABDC}..

$ABD \not\subset ABC$

$ABD \not\subset ADC$. So, it is not P.D.

$CD \subset ADC$, So, it is PD.

So, create a table.

Second table (CDF) $R_2(CDF)$ & (ABCDE)

(CD is primary key here)

So, CD is foreign key in original table.

This is preserving the dependency $CD \to F$.

if only CF (or) only DF. then no ~~dat~~ dependency preserving.

A table has fields F_1, F_2, F_3, F_4 and F_5, with following dependies. $R(F_1, F_2, F_3, F_4, F_5)$.

$F_1 \to F_3$, $F_2 \to F_4$, $F_1F_2 \to F_5$.

→ $F_1F_2 = \{F_1, F_2, F_3, F_4, F_5\}$. → candidate key.

F_1F_2 → prime F_3, F_4, F_5 → non prime.

$F_1 \to F_3$ $F_1 \subset F_1F_2$ So it is PD.

$F_2 \to F_4$ $F_2 \subset F_1F_2$ So it is PD

So create other tables (F_1, F_3) (F_2, F_4) & (F_1, F_2, F_5) (F_1 & F_2 foreign keys)

3NF :- No transitivity.

In 2NF, PA ← PA & NPA ← NPA is possible.
but in 3NF, NOT possible for NPA ← NPA.

CK ← NP_2
NP_2 ← NP_1. } This is transitivity.

Non prime →
Not prime →
Both should depend on Prime (directly)

A relation is said to be in 3NF, iff:

→ It is in 2NF
→ No transitive FD's are present.
→ A non-prime attribute must not transitively dependent on key attribute.

$\forall FD_{NT}: X \rightarrow Y \Leftrightarrow (X:SK \vee Y:PA)$

$FD_T : X \rightarrow Y \Leftrightarrow$ X is not CK $\wedge$ Y:NPA.

Second (supp.no, status, city)

FD set :- supp.no → status.
supp.no → city
city → status

Supp.no → CK → prime ; city, status → NP.

Here city → status → ~~No~~ creating problem (It is transitive)
→ So create another table.
→ Here city is primary key
& it is foreign key in original table

(supp.no, city) (city, status).

R(ABCDEFGH)

CH → G, A → BC, B → CFH, E → A, F → EG

AD = {A, B, C, D, F, H, G, E}

BD = {B, D, C, F, H, G, E, A}.

CD = {C, D, · (x)

ED = {A, D, E, B, C, F, H, G}

DF = {D, F, E, G, A, B, C, F, H}.

DG = {D, G, (x)

DH = {D, H, (x)

HCDG = {C, D, G, H,

So candidate keys :- AD, BD, ED, DF.

A, B, D, E, F → prime, C, G, H → NP.

CH → G
NP NP.

CH ⊄ {A, B, D, E, F} So, it is not PD

A → BC
NP.

A ⊂ {A, B, D, E, F}, So, it is PD.

So, Not in 2NF.

BCNF. (Boyce-Codd Normal form).

A relation is said to be in BCNF, iff. No dependency preserving.

it is in 3NF.

if and if only if every determinant is a super key.

∀FDNT: X → Y ⟺ X : SK.

Supplier_part (supp.no, supp.name partno., quantity).

supp.no, partno → quan.

supp.no, part no. → suppname.

supp.name, partno. → quant.

Suppname, partno → supno.

Sup.name → supno.

Sup.no → sup.name

Partno = {partno.} (x)

Supno., partno. = {Supno., PNo., quant, SName}. ✓

Supname, partno. = { " " " " ✓

CK :- Sup no., part no ; Supname, partno.

Here, Supname → Supno. , Sup.no → suppname

Not super key so, create another table.

(Sno., Sname)
Primary key
anyone is kept as primary

(Sno., quant, partno.)
foreign key.

R(ABCDEPG).

AB → CD, DE → P, C → E, P → C, B → G.

$AB = \{A, B, C, D, E, P, G\}$

AB → candidate key.

A, B → prime, C, D, E, P, G → Non prime.

AB → CD
Prime . NP

$AB \subset \{A, B\}$. So it is PD.

So not in 2NF.

Relational Algebra (Procedural

It is based on set theory, so it removes duplicates.
It uses certain operations to answer a query.

select (σ) :- to select required tuples.

Project (π)
selects the columns, vertical separation

o/p < Unary
      Binary.

in RA) only Unary & Binary.

$\pi_{A_1, A_2, A_3}(R)$ Here relation is the input (single input)
Projection is unary operation

$\pi_{A_1}(\underbrace{R_1 \bowtie R_2}_{\text{this is binary}})$ for projection, only single table.
Unary.
Join is a binary operator.

Issuer(IID, IName, IM, IA)
Book(BID, BN, BA, BC (code), IC, BP).
IBR(IID, BID).

Issuer issues Book.
many to many.

Issuer

I_1	IAB	91	AP
I_2	IBC	92	MP
I_3	ICD	93	UP
I_4	IDE	94	MH.

Book

B_1	Dbms	Korth.	10	2	100
B_2	OS	Galv.	10	1	200
B_3	DDA	Coreman	10	1	300
B_4	C	Kametkar	10	1	100.

IBR.

I_1	B_1
I_2	B_1
I_3	B_2
I_3	B_3
I_4	B_4.

$\pi_{BA}(Book)$ Korth, Galv, Coreman, Kanmetkar

$\pi_{IName}(Issuer)$ IAB, IBC, ICD, IDE.

Selection:- Horizontal separation

Filter for rows/records.
Only rows which satisfy our condition.
predicate → It is the condition.

σ_P P: predicate.
The tuples which only satisfy our predicate.

π_{BID} (Book). Projects Book ID from Book

$\cancel{\sigma_{BC>5}(\pi_{BID}(Book))}$ $\pi_{BID}(\sigma_{BC>5}(Book))$

$\cancel{\pi_{BID}(Book(\sigma_{BC>5}))}$ Select only those tuples which satisfy condition & projects BID.

Selection is a unary operator.

Rename:- (ρ) :- also unary operator.

$\rho_{newtablename(NA1, NA2)}(R)$.

R with A1, A2 is renamed with Newtablename(NA1, NA2)

$\rho_{(NA1, NA2)}(R)$ attributes are renamed.
for renaming attributes, write within brackets.
$\rho_{NA1}(R)$
· Tablename R is renamed with NA1.

All BID where BC>5.
Rename the new table as BookID.

$\rho_{BookID}(\pi_{BID}(\sigma_{BC>5}(Book)))$

$\rho_{BA='BookAdd'}(Book)$
it will rename R.

Union:- $R_1 \cup R_2$ arity of R_1 and R_2 must be same.

$R_1 \cup R_2$
$= \{t \mid t \in R_1 \text{ OR } t \in R_2\}$

(1) no. of attributes should be equal in R_1 & R_2
(2) Domain of the attributes (corresponding) must be same (Domain compatibility)

Suppose Issuer

IID	IName	IM	IA

Book.

BID	BN	BC	BA

π_{IID} (Issuer) $\cup$ π_{BID} (Book)

Intersection:- $R_1 \cap R_2 = \{t \mid t \in R_1 \wedge t \in R_2\}$.

Arity of R_1 & R_2 must be same.

Set difference:- $R_1 - R_2 = \{t \mid t \in R_1$ and $t \notin R_2\}$.

Arity must be same

Ex:-

A

Col1	Col2
1	1
1	2

B.

Col1	Col2
1	1
1	3

A∪B.

Col1	Col2
1	1
1	2
1	3

A∩B.

Col1	Col2
1	1

A−B.

Col1	Col2
1	2

Cartesian product:- (also called cross product)

$R_1 \times R_2 = \{ \langle r_1, r_2 \rangle \mid r_1 \in R_1 \wedge r_2 \in R_2 \}$

cartesian product:-

A×B -

1	1	1	1
1	1	1	3
1	2	1	1
1	2	1	3

Join:- filter acc. to condition (selection criteria/filter).

first finds cartesian product & applies condition.

So, $t < t_1 \times t_2$ if t_1 – no. of tuples in A.

t_2 – " " " B.

selection criteria/filter over cartesian product.

Inner join
- Theta join. (θ)
- equijoin → in this there is a special case of Natural join

R_1 (C_1 C_2) R_2 (C_3 C_4)

$R_1 \bowtie_\theta R_2$.

Join represented by $\bowtie$

$\bowtie_\theta$ ⇒ theta join

$\bowtie_=$ ⇒ equi join.

$\bowtie_{R_1.C_2 > R_2.C_3}$. ⇒ theta join (other than equal to).

$\bowtie_{R_1.C_2 = R_2.C_3}$ ⇒ equi join (relational o/p is equal to)

Suppose

$\bowtie_{R_1 \cdot C_1} = \bowtie_{R_2 \cdot C_1}$ $R_1(C_1 C_2)$

Here common attribute $R_2(C_1 C_3 C_4)$

So, it is a natural join.

If in ~~are~~ an equi join, the attributes are common then Natural join.

$\bowtie_{R_1 \cdot C_2} = \bowtie_{R_2 \cdot C_3}$ C_2 C_3

SID SID

then both equi & Natural join.

if C_2 - SID C_3 - BID

then only equi join.

R

$R_1 \bowtie R_2$.

R_1, R_2 can be instance of one relation or may be diff. totally.

$R_1 \bowtie_{R_1 \cdot C_1 > R_2 \cdot C_2} R_2 \Rightarrow$ Theta join.

~~Here R~~

~~for inner~~ joi

Outer join:-

left outer Join:- We care about the tuples from left which didn't satisfy join criteria.

Right outer join:- We care about the tuples from right which didn't satisfy join criteria.

Full outer join:-

$A \bowtie_{A \cdot col2 > B \cdot col2} (B)$

col1	col2
1	2

(Theta join)

A×B

11 11

11 13

12 11(✓)

12 13.

$A \bowtie_{A col2 = B \cdot col2} (B)$

col1	col2
1	1

(Equi join)

Natural join.

C

Num	Sq.
2	4
3	9

D

Num	Cube
2	8
3	27

C×D

2	4	2	8
2	4	3	27
3	9	2	8
3	9	3	27

C ⋈ D.

Num	sq	Cube
2	4	4
3	9	27

Outer join:-

A

Num	Sq
2	4
3	9
4	16

B

Num	Cube
2	8
3	18
5	75

A×B

2	4	2	8 ✓
2	4	3	18
2	4	5	75
3	9	2	8 ✗
3	9	3	18 ✓
3	9	5	75
4	16	2	8
4	16	3	18
4	16	5	75

A ⟕ B

Num	Sq	Cube
2	4	8
3	9	18
4	16	NULL

A ⟖ B

Num	Sq	Cube
2	4	8
3	9	18
5	NULL	75

A ⟗ B

Num	Sq	Cube
2	4	8
3	9	18
4	16	NULL
5	NULL	75

Relational Calculus

Relational calculus is not procedural.

DRC (Domain) TRC (Tuple)

Query form $\{T \mid F(T)\}$ → formula.

T: Tuple variable.

(1) variables.

(2) Many T.V are allowed in formulas.

has to be a Free variable

↳ which donot include existence quantifiers.

Projection is not allowed. $(\exists\ \forall)$ if $\exists T$ $\forall T$. then its is not a free variable.

F(T) - a formula can be an atomic formula.

(1) $T \in$ Relation

S ∈ sailor.

S.Rating > R.Rating

(2) T.a op S.b (<, >, <=, >=, =, <> ≠).

(3) T.a op constant

if F_1 & F_2 are formula, then. $\neg F_1$, $\neg F_2$, $F_1 \wedge F_2$, $F_1 \vee F_2$, $F_1 \Rightarrow F_2$ are also formulas.

if F(T) is a formula then $\forall T \in R(F(T))$

$\exists T \in R(F(T))$

Sailor (Sid, SName, ~~sa~~ age, rating)

Boat (Bid, BN, Color)

Reserve (Bid, Sid, Day)

Find a sailor with rating >7

$\Rightarrow \{S \mid S \in Sailor \wedge S.rating > 7\}$

Find a sailor whose age < 30.

$\Rightarrow \{S \mid S \in Sailor \wedge S.age < 30\}$

Find the name of the sailor with rating > 7.

$\Rightarrow \{S \mid \exists S_1 \in Sailor(S_1.rating > 7 \wedge S_1.Name = S.Name)\}$

Find age and name of the sailors with rating >7.

$\Rightarrow \{S \mid \exists S_1 \in Sailor(S_1.rating > 7 \wedge S_1.name = S.name \wedge S_1.age = S.age)\}$

$S_1 \Rightarrow$ complete table with $R>7$.

$S \Rightarrow$ only name & age from S_1

selection $\{S \mid S \in Sailor \wedge S.rating > 7\}$. $\rightarrow$ ~~not~~ a free variable

projection $\{S \mid \exists S_1 \in Sailor (S_1.rating > 7 \wedge S_1.name = S.name)\}$ $\rightarrow$ not a free variable.

Find sailors rating >7 and who have reserved boat id $= 103$.

$\{S \mid S \in Sailor \wedge S.rating > 7 \wedge (\exists R \in Reserve (R.Sid = S.Sid \wedge R.Bid = 103))\}$

Find sailor name for above.

$\{S \mid \exists S_1 \in Sailor (S_1.rating > 7 \wedge S_1.name = S.name) \wedge (\exists R \in Reserve (R.Sid = S_1.Sid \wedge R.Bid = 103))\}$

Find sailor name who reserved a red Boat.

$\{S \mid \exists S_1 \in Sailor(S_1.name = S.name \wedge (\exists R \in Reserve (R.Sid = S_1.Sid \wedge (\exists B \in Boat (B.Bid = R.Bid \wedge B.color = Red)))))\}$

S is free variable

S_1, R, B aren't free variable.

or)

$\{S \mid \exists S_1, \exists R, \exists B (S_1.Sid = R.Sid \wedge R.Bid = B.Bid \wedge B.Color = Red \wedge S_1.name = S.name)\}$

Find a sailor who has reserved red or green boat.

~~{S | S ∈ Sailor (∃R ∈ Reserve (~~

Same as above but $\rightarrow B.color = red \vee B.color = green$.

Find a sailor who reserved Red & Green color boat

(B.color = red $\wedge$ B.color = green)x.

which means a sailor has a boat with both green & red colour.

So write separates.

{s | s ∈ Sailor (∃ R ∈ Reserve (R.Sid = S.Sid ∧ (∃ B ∈ Boat (R.Bid = B.Bid ∧ B.color = red)))

{s | s ∈ Sailor, ∃ R ∈ Reserve, ∃ B ∈ Boat (S.Sid = R.Sid ∧ R.Bid = B.Bid ∧ B.color = read) ∧ (S.Sid = R.Sid ∧ R.Bid = B.Bid ∧ B.color = green)

Find a sailors who has reserved two diff. boats

~~{s | s ∈ Sailor, ∃ R ∈ Reserve (R.Sid = S.Sid ∧~~

{s | s ∈ Sailor, ∃ R_1 ∈ Reserve, ∃ R_2 ∈ Reserve (R_1.sid = S.Sid ∧ R_2.Sid = S.Sid ∧ R_1.bid ≠ R_2.bid)}

Domain Relational Calculus (DRC).

Domain of attributes belonging to a relation.

Query format :- $\{\langle x_1 x_2 \ldots x_n \rangle \mid F(\langle x_1, x_2 \ldots x_n \rangle)\}$.

$x_1 x_2 \ldots x_n \rightarrow$ Domain variables.

In TRC, we can't mention individual attribute we ask for whole tuple. $\{\underset{\langle x \rangle}{S.Id}$

But in DRC, it is possible.

$\langle x_1 x_2 \ldots x_n \rangle \in R$. where R is a relation on n attributes and $x_1, x_2, \ldots, x_n$ are domain variables.

$x\ op\ y$ $\quad$ $x\ op$ content.

Find all sailors with a rating above 7.

Sailor (Sid, SN, R, age)

$\{\langle Sid, SN, R, age \rangle \mid \langle Sid, SN, R, age \rangle \in Sailor \wedge R > 7\}$

$\{T \mid T \in Sailor \wedge \cancel{T.rating}\ T.R > 7\}$

only Sid

$\rightarrow$ $\{\langle Sid \rangle \mid \exists SN, R, age\ (\langle SId\ SN\ R\ age \rangle \in Sailor \wedge R > 7)\}$.

Find the names of sailors who have reserved a boat 103.

TRC ~~$\{T \mid \exists T_1 \in Sailor$~~ $\wedge \exists$

$\{T \mid \exists T_1 \in Sailor \wedge (\exists R \in Reserve\ (R.Sid = T_1.Sid \wedge R.Bid = 103 \wedge T_1.name = T.name))\}$

I	N	T	A	
Sid	SN	R	Age	

Reserve (Sid, Bid, Day)
$\quad$ Ir $\quad$ Br $\quad$ D.

$\{N \mid \exists ITA\ (\langle INTA \rangle \in Sailor \wedge \exists Ir\, Br\, D\ (\langle Ir\, Br\, D \rangle \in Reserve \wedge Ir = I \wedge Br = 103))\}$.

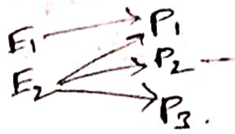
Find the names of sailors who have reserved a red boat.

S(I, N, T, A), B(B, BN, C) R(Ir, Br, D).

$\{N \mid \exists ITA \cdot (\langle INTA \rangle \in Sailor \wedge \exists\ Ir\, Br\, D\ (\langle Ir\, Br\, D \rangle \in R \wedge Ir = I \wedge \exists B, BN, C\ (\langle B\, BN\, C \rangle \in B \wedge B = Br \wedge C = red)))\}$

Division operator.

- Emp who is working on all projects
- Sailor who reserved all boats.
- Student who enrolled himself in all courses



Ans:- E_2.

"all"

Possible with division operator.

~~$R_1 \div R_2$~~ $R_1(X)$ $R_2(Y)$ X, Y are set of attributes

$R(Z) = R_1(X) \div R_2(Y) \Rightarrow$ (1) $Z = X - Y$. (attributes only in R_1 not in R_2). $(X = Z \cup Y)$

(2) T_R is a Tuple from R_1. iff it is associated with all the tuples of R_2.

$R_1 \div R_2 = \{ t \mid t \in \pi_{R_1 - R_2}(R_1) \wedge \forall u \in R_2 (tu \in R_1) \}$

all tuples (u) in S is related with t coming from R_1

Example:- R

A	B
α	1
α	2
α	3
β	1
γ	1
δ	1
δ	3
δ	4
ϵ	6
ϵ	1
β	2

S

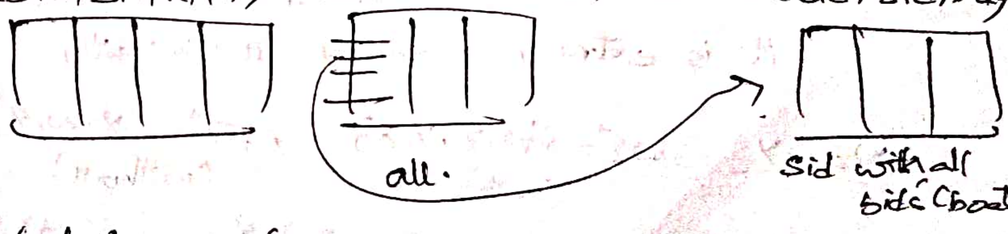
B
1
2

$R \div S$.

A
α
β

Sailor who reserved all boats. (using TRC)

Sailor (Sid, SN, R, A); Boat (Bid, BN, C); Reserve (Bid, Sid, Day)



$\{ S \mid S \in Sailor \wedge (\forall B \in Boat (\exists R \in Reserve \wedge R.Bid = B.Bid \wedge S.Sid = R.Sid)) \}$

unsafe queries $\{S \mid \neg(S \in Sailors)\}$.

(or) $\{S \mid S \notin Sailors\}$ infinite no of solutions.

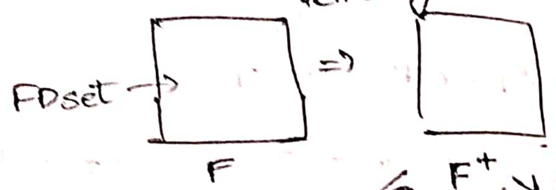
Every query that can be expressed in R.A can be expressed as a safe query in DRC/TRC ; the converse is also true.

R.A → Procedural
R.C → Non-Procedural

→ SQL – Non-Procedural (Implementation)

Canonical cover of FD set / Minimal set of FDs / Irreducible set of FDs.

removing extra or unwanted attributes



So this is minimal set.

we have redundant FD, Extrageneous attributes } removing them.

Suppose $AB \to D$, $B \to D$ — here B is sufficient to uniquely identify D. So, A is extra, Total $\underline{AB \to D}$ is extrageneous.

$R(WXYZ)$ $X \to W$, $WZ \to XY$, $Y \to WXZ$.

Step (1) Apply decomposition rule.

$X \to W$, $WZ \to X$, $WZ \to Y$, $Y \to W$, $Y \to X$, $Y \to Z$.

Step (2) Identify redundancy.

find X^+ using the FD. $(X \to W)$ $X^+ = \{XW\}$ ✓ → So, this is essential.

X^+ without using the FD. $(X \to W)$ $X^+ = \{X\}$

$\underline{WZ \to X}$ $WZ^+ = \{X, W, Z, Y\}$ (with) $WZ^+ = \{W, Z, Y, X\}$. (without)

So closure is same, it is extra, so remove it instantly.

$WZ \to Y$ $WZ^+ = \{W, Z, Y, X\}$ (with), $WZ^+ = \{W, Z\}$. (without)

Not same, So $WZ \to Y$ is essential.

$Y \to W$ $Y^+ = \{Y, W, X, Z\}$ (with), $Y^+ = \{Y, X, Z, W\}$. (without)

Same, so remove it.

$Y \to X \Rightarrow Y^+ = \{Y, X, Z, W\}$, $Y^+ = \{Y, Z\}$.
not same, so it is essential.

$Y \to Z \Rightarrow Y^+ = \{Y, Z, X, W\}$, $Y^+ = \{Y, X, W\}$.
not same, so it is essential.

$X \to W, WZ \to Y, Y \to X, Y \to Z$.

Step(3):- Find redundancy on left hand side.

$\underbrace{X \to W, Y \to X, Y \to Z}_{\text{one attribute}}$

$WZ \to Y$.
$WZ^+ = \{W, Z, Y, X\}$.
$W^+ = \{W\}$
$Z^+ = \{Z\}$ } So both are essential.

So final answer:- $X \to W$, $Y \to X$, $Y \to Z$.
(Canonical cover)

Ex-2:- $A \to BC$, $B \to C$, $A \to B$, $AB \to C$ R(ABC)

(i) Apply decomposition rule.
$A \to B$, $A \to C$, $B \to C$, $A \to B$, $AB \to C$.

$A \to B$, $A \to C$, $B \to C$, $AB \to C$.

~~$A \to C$, $A \to C$, $AB \to C$.~~
~~$A \to C$.~~

So $A \to C$ no need.

$A \to B$, $B \to C$, $AB \to C$.
B is alone sufficient.

So, $A \to B$, $B \to C$.
(canonical cover)

Ex-3:- ~~$A \to B$~~, $A \to C$, $B \to C$, $A \to B$, $AB \to C$.

$A \to C$ $A^+ = \{A, C, B\}$, $A^+ = \{A, B, C\}$.
with without
So, remove it.

$B \to C$ - $B^+ = \{B, C\}$, $B^+ = \{B\}$.
with without.
So, essential.

$A \to B$, $A^+ = \{A, B, C\}$, $A^+ = \{A\}$.
essential.

$AB \to C$, $AB^+ = \{A, B, C\}$, $AB^+ = \{A, B, C\}$.
So remove it.

So, $A \to B$, $B \to C$

Canonical ~~set~~ cover is not unique.

1, 2, 3, 4, 5

$\longrightarrow$ depends on the order, we go. so not unique.

Ex.

$C \to B$, $CB \to AC$, $CAE \to FB$, $D \to E$, $CA \to B$.

(1) $C \to B$, $CB \to A$, $CB \to C$, $CAE \to F$, $CAE \to B$, $D \to E$, $CA \to B$.

(2) $C \to B$; $C^+ = \{C, B, A\}$ $\quad C^+ = \{C\}$.

So, essential.

$C \to B$ is sufficient then $CA \to B$ is extrageneous, so, remove.
$CAE \to B$ " " " So, remove.

(3) $CB \to A$; $CB^+ = \{C, B, A\}$, $CB^+ = \{C, B, A\}$.

same.

So, remove it. without $C^+ = \{CB\}$. $B^+ = \{B\}$
So, B no need.

$CB \to C$; $CB^+ = \{C, B, A\}$, $CB^+ = \{C, B, A\}$.
Redundant
So, remove it.

$CAE \to F$; $CAE^+ = \{C, A, E, B, F\}$, $CAE^+ = \{C, A, E, B\}$.
So, essential. , A is no need

without this $C^+ =$ $A^+ =$ $E^+ =$
then 'F' won't come, So, it is essential.

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

So, essential.

($C \to B$, $CE \to F$, $D \to E$, $C \to A$.)

Ex

$F = \{A \to B, AB \to C, D \to AC, D \to E\}$ $G = \{A \to BC, D \to AB\}$.

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

$G = A \to B, A \to C, D \to A, D \to B$.

$F^+ =$ (Canonical cover of F)

$A \to B$. $AB \to C$

A identifies B. , So $A \to B$, $A \to C$.

$AB \to C$ $\quad$ $A \to C$. is essential
no need. without this 'C' won't come.

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

So, essential.

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

~~So essential~~

$D \to C$, $A \to C$.
$D \to A$, $\quad$ So, $D \to C$ is no need.

So, $A \to B, A \to C, D \to A, D \to E$.

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

$\overline{G^+}$

$A \to B, A \to C, D \to A, D \to B$.

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

So, essential.

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

So, essential.

$D \to A$ → essential.

$D \to B$. $D \to A$, $A \to B$ so, $D \to B$ no need.

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

F covers G, but G doesn't cover F.

if suppose G covers H & H covers G.

then H & G are equivalent.