SUPER KEY | CANDIDATE KEY | PRIMARY KEY

04 October 2023 19:44

SID	Name	Marks	Dept	course
1	a	78	CS	C1
2	b	60	EE	C2
3	a	78	CS	C2
4	b	60	EE	C3
5	c	80	IT	C2

- Get record with name = a
  → 2 records exist

- something to identify records uniquely is required. → key concept
  → S.No is a key

- Is {Dept, course} a key?
  Eg (CS, C1) → no other combo
  (EE, C1) (CS, C2) (EE, C3) (IT, C2)
  → all unique, so {Dept, course} is a key ✓

- Is {Name, marks} a key?
  (a, 78) is repeated twice. ✗

(x) (9, 78) is repeated

Is {name, Marks, Dept, course}

a key? YES (✓)

Taking all attributes will definitely form a key.

DEFN: A key is a single attribute or a set of attributes that uniquely identify a record in a table.

TYPES OF KEYS

(i) superkey ≡ key [same defn]

eg SID → SK (✓)

SID, Sname → SK (✓)

If α is a SK, all combinations with α will definitely be a SK

SID, marks → (✓) SK

{name, marks, dept} → (x) (9, 78, CS) repeated

{marks, course, dept} → (✓) SK

maximum no. of superkeys in a ... form

Q : Maximum no. of superkeys in a relation

$^5C_1 + {}^5C_2 + {}^5C_3 + {}^5C_4 + {}^5C_5$

$= 2^5 - 1 = 32 - 1 = 31$

[remember $^nC_r = \frac{\lfloor n}{\lfloor n-r \lfloor r}$]

If there are 4 attributes

$^4C_1 + {}^4C_2 + {}^4C_3 + {}^4C_4 = 2^4 - 1 = 15$

In general, a relation with n attributes can have a maximum of $2^n - 1$ superkeys

---

R(A, B, C, D)

A	B	C	D
1	1	5	1
2	1	7	1
3	1	7	1
4	2	7	1
5	2	5	1
6	2	5	2

Find the total no. of superkeys relation

Find the total no. of sup... in this relation

A	AB	ABC	ABCD
B	AC	ABD	
C	AD	ACD	
D	BC	BCD	
	BD		
	CD		

superkeys → { A, AB, AC, AD, ABC, ABD, ACD, ABCD }

→ 8 superkeys are present

---

② candidate keys

DEFN :- A candidate key is a superkey whose proper subsets are not superkeys

≡ minimal super-key

R(A, B, C)

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

3 2 1

4 2 2

$\boxed{A}$ AB AC ABC

B

C $\boxed{BC}$

candidate keys = 2

super keys = 5

$\square + \bigcirc$

$\square$ → candidate keys

$\bigcirc$ → non-minimal super keys

---

* Minimal → as minimum as possible

* Minimum → having minimum no. of attributes

* Every candidate key is a super key, but every super key need not be a candidate key.

---

③ PRIMARY KEY

- Any one candidate key is chosen and made as the primary key.
- The chosen primary key must have no NULL values.
- Remaining candidate keys will be considered as secondary keys / alternate keys.
- A relation can have more than one super key & candidate key but must have only one primary key.

R(A, B, C) A is candidate key.

Find the total no. of super keys.

A with all combos of BC

↪ 2^2 = $\boxed{4 \text{ super keys}}$

R(A, B, C) AC is a candidate key.

Find total no. of super keys.

R(...

Find total no. of super keys.

AC with all combos of B

↳ $2^1 = 2$ super keys

R(A, B, C, D) A & D are candidate keys. Find the total no. of super keys

$\boxed{A}$	AB	ABC	ABCD
B	AC	ABD	
$\boxed{C}$	AD	ACD	
D	BC	BCD	
	BD		
	CD		

no. of candidate keys = 2

no. of super keys = 12

FUNCTIONAL DEPENDENCIES

04 October 2023 20:20

FD : $x \to y$

If value of x is known, y can be uniquely determined

x	y
⋮	⋮

relation given

∴ It doesn't mean y can be computed functionally using value of x. nothing like this.

$y = x^2 + 5x$ ✗

$x \longrightarrow y$

determinant — dependant

x	y
1	1
2	1
3	2
4	3

$x = 2$: $y = 1$ uniquely

$x = 3$: $y = 2$ uniquely

$x \to y$ exists ✓

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

Here if $x = 2$:
$y = 1$ (or) 5
cannot be uniquely determined

So $x \to y$ ⊗ does not exist

$y = x^2$

$x = 5 \to y = 25$

$x = -5 \to y = 25$

But $y = 25 \to x = \pm 5$

So $x \to y$ exists

$y \to x$ does not exist

$x \to y$

For functional dependency $x \to y$ to exist, if the same x value exists in 2 or more records, the corresponding y values must also match.

In a class

Rno → name exists ✓

But in a seminar hall

Rno → name ⊗ does not exist

many students

Rno → name ⊗

[CSE 13 , ECE 15, many students with the same rno may exist]

Rno, Dept → name exists ✓

FD : $x \to y$ is true

if $t_1.x = t_2.x$

then $t_1.y = t_2.y$

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

$t_1 = (1,1)$ $t_2 = (2,1)$

$t_1.x \neq t_2.x$ ⇒ no need to check y

$t_1 = (2,1)$ $t_2 = (2,5)$

$t_1.x = t_2.x$ but $t_1.y \neq t_2.y$

so $x \to y$ ⊗ does not exist

x	y
1	1
2	1
3	2
4	3

here $x \to y$ functional dependency exists

R No	Name	Marks	Dept	course
1	a	78	CS	C1
2	b	60	EE	C1
3	a	78	CS	C2
4	b	60	EE	C3
5	c	80	IT	C3
6	d	80	EC	C2

check if the functional dependencies exist :

(1) R NO $\longrightarrow$ Name

as no rno is duplicated

$t_1 . x \neq t_2 . x \Rightarrow$ no need to check y

Dependency exist Rno $\rightarrow$ name ✓

(2) Name $\longrightarrow$ R NO

$t_1 = (a, 1)$ $t_2 = (a, 3)$

$t_1 . x = t_2 . x$ but $t_1 . y \neq t_2 . y$

Dependency does not exist ✗

(3) R no $\longrightarrow$ marks ✓

④ Dept → course

$t_1 = (CS, C_1)$ $t_2 = (CS, C_2)$

$t_1.x = t_2.x$ but $t_1.y \neq t_2.y$

dependency does not exist ⓧ

⑤ course → Dept

$t_1 = (C_1, CS)$ $t_2 = (C_1, EE)$

$t_1.x = t_2.x$ but $t_1.y \neq t_2.y$

dependency does not exist ⓧ

⑥ Marks → Dept

in all cases where $t_1.x = t_2.x$

$t_1.y = t_2.y$ holds true

dependency exist ✓

marks → dept

⑦ $\underbrace{\text{Rno, name}}_{x} \rightarrow \underbrace{\text{marks}}_{y}$

No rno repetition exists ✓

so no x repetition

rno, name → marks

(8) name → marks

FD exists name → marks

(9) name , marks → dept

t_1 = (a 78 , CS) t_2 = (a 78, CS)

only place where $t_1 . x = t_2 . x$

here $t_1 . y = t_2 . y$ => no problem

name, marks → dept

(10) name, marks → dept , course

t_1 = (a 78 , CS C1)

t_2 = (a 78, CS C2)

$t_1 . x = t_2 . x$ but $t_1 . y \neq t_2 . y$

dependency does not exist.

(11) rno → name , marks ✓

(12) dept, course → name ✓

(13) rno, marks → dept ✓

(14) name → course ✗

(15) name, marks, dept → rno ✗

(15) name, marks, dept → rno

ARMSTRONG'S AXIOMS IN DBMS

04 October 2023 22:06

Types of Functional Dependencies

1. Trivial
2. Non-Trivial
3. Multi-valued
4. Transitive

FD : $X \rightarrow Y$

if $t_1.x = t_2.x$

then $t_1.y = t_2.y$

1. TRIVIAL FD

FD : $X \rightarrow Y$

if $Y \subseteq X \Rightarrow X \rightarrow Y$ is a trivial functional dependency

Rno → Rno

Rno, Name → Name

Rno, name → Rno

trivial FD always valid

No use in writing these FD

[understood by default]

2. NON TRIVIAL FDs

$X \cap Y = \phi$

② NON TRIVIAL FDs

$X \rightarrow Y$ and $X \cap Y = \phi$

RNO → Name → may / may not be valid → use table to check

$X \rightarrow Y$ $Y \not\subset X$ ⇒ semi trivial FD

RNO, Name → Name, Marks

↳ check table to check validity

ARMSTRONG'S AXIOMS / INFERENCE RULES

RNO	Name	Marks	Dept	course
1	a	78	CS	C_1
2	b	60	EE	C_1
3	a	78	CS	C_2
4	b	60	EE	C_3
5	c	80	IT	C_2

① REFLEXIVITY

$X \rightarrow X$ $X \rightarrow Y$ $Y \subseteq X$

RNO → RNO Name → Name

RNO → RNO
Marks → marks
course → course

Name → Name
Dept → Dept
RNo, Name → RNo

---

② TRANSITIVITY

If (X → Y and Y → Z) then (X → Z) exists

Ram is sibling of shyam
shyam is sibling of Mohan
⇓
Ram is sibling of Mohan

Name → Marks
Marks → Dept
} Name → Dept

If a new tuple (6, d, 80, Er, c4)

Name → Marks ✓
But Marks → Dept ✗

Nothing can be directly commented about Name → Dept
on checking table ⇒ it exists ✓

---

③ AUGMENTATION

③ AUGMENTATION

If $X \rightarrow Y$ then $XA \rightarrow YA$

If RNO $\rightarrow$ name,

RNO, Marks $\rightarrow$ Name, Marks

④ UNION

If $X \rightarrow Y$ and $X \rightarrow Z$ then $X \rightarrow YZ$

If rno $\rightarrow$ name and rno $\rightarrow$ marks then

rno $\rightarrow$ name, marks

⑤ DECOMPOSITION / SPLITTING :

If $X \rightarrow YZ$ then $X \rightarrow Y$ and $X \rightarrow Z$

If $XY \rightarrow Z$ then $\left.\begin{matrix} X \rightarrow Z \\ Y \rightarrow Z \end{matrix}\right\}$ ✗

(wrong)

The determinant side (LHS) can never be split. only the RHS can be split.

name, marks $\rightarrow$ dept, course

$\Downarrow$

name, marks $\rightarrow$ dept

name, marks → dept
name, marks → course

---

⑥ PSEUDO TRANSITIVITY

If $X \to Y$ and $YZ \to A$
then $XZ \to A$

Rno → name
Name, Marks → Dept
then RNO, marks → Dept

---

⑦ COMPOSITION

If $X \to Y$ and $A \to B$ then
$XA \to YB$

ATTRIBUTE CLOSURE

09 October 2023 19:12

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

FD : { A → B, B → C, C → D, D → E }

A → A (reflexivity)

A → B, B → C ⇒ A → C (transitivity)

A → C, C → D ⇒ A → D

A → D, D → E ⇒ A → E

So A → ABCDE

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

attribute closure of A

Similarly for B :

B → B, B → C, B → D, B → E

B $\not\to$ A not possible

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

Also

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

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

$E^+ = \{E\}$

In simple words :

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

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

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

By augmentation property :

$A \rightarrow B \Rightarrow AD \rightarrow BD \Rightarrow$ $AD \rightarrow B$, $AD \rightarrow D$

. . . . .

$AD \rightarrow \underbrace{ABCDE}$ can be proved

attribute closure of AD

---

super key wrt attribute closure :

super key $\downarrow$ set of attributes whose closure contains all attributes of the relation.

$A^+ = \{A, B, C, D, E\} \Rightarrow$ super key ✓

$AD^+ = \{A, B, C, D, E\} \Rightarrow$ super key ✓

$AB^+ = \{A, B, C, D, E\} \Rightarrow$ super key ✓

$B^+ = \{B, C, D, E\} \Rightarrow$ not a super key ✗

$\boxed{A}$ $\underbrace{\{B, C, D, E\}}_{\text{all combinations}}$ $= 2^4 =$ 16 super keys

super key

$BCDE^+ = \{B, C, D, E\} \Rightarrow$ ✗

no subset of this can be a super key.

candidate key : minimal super key

Here A is the only candidate

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

FD : $\{A \to B, D \to E\}$

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

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

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

all attributes of the relation will obviously form a super key.

$ABDE^+ = \{A, B, D, E\} \Rightarrow$ ✗

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

super key

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

super key.

also ⇒ minimal → candidate key

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

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

$\boxed{\begin{array}{l} ACD \Rightarrow 1 \text{ candidate key} \\ BE \text{ combos} \Rightarrow 2^2 = 4 \text{ super keys} \end{array}}$

CANDIDATE KEYS

09 October 2023 19:35

$R(A, B, C, D, E\}$ $FD = \{A \to B, D \to E\}$

Find all candidate keys and super keys.

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

↳ by reflexivity property (each attribute can determine itself)

Remove dependencies :

$A \cancel{B} C D E$ $A \to B$, so discard B.

$A C D \cancel{E}$ $D \to E$, so discard E.

$(ACD) \Rightarrow$ nothing can be discarded

↳ super key ✓

also it can be verified that no proper subset is a super key ⇒ so ACD is a candidate key.

Prime Attributes

↓

those attributes which are a part of the candidate key.

In the previous qn,
prime attributes = { A, C, D }

> ACD → candidate key.
> Now check if any of the prime attributes appear on the right hand side (RHS) of any functional dependency. If no ⇒ there will be only one candidate key.

A, C, D → none of them appear in the RHS. So ACD is the only candidate key.

R (A, B, C, D)   FD = { A → B, B → C, C → A }

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

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

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

↳ no more attributes can be discarded.

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

So AD ⇒ candidate key

prime attributes = { A, D }

so

Prime attributes = {A, D}

A is present on the RHS of a FD.

So AD is not the only candidate key.

$C \to A$ so CD is a SK

$B \to C$ so BD is a SK

now checking proper subsets

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

candidate keys = {AD, BD, CD}

Prime attributes = {A, B, C, D}

R (A, B, C, D)

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

A	$\boxed{AB}$	ABC	
B	AC	ABD	ABCD
C	$\boxed{AD}$	ACD	
D	$\boxed{BC}$	BCD	
	BD		
	$\boxed{CD}$		

4 candidate keys

4 + 5 = 9 super keys

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

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

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

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

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

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

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

so AB is a candidate key.

For now prime attributes = $\{A, B\}$

$D \to A$ so more candidate keys are present

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

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

so BD is a candidate key

Now prime attributes = $\{A, B, D\}$

$C \to B$, so $AB \to DB$, $AB \to AC$

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

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

$A^+ = \{A\}$

super key but not candidate key

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

super key

candidate key

Now prime attributes = $\{A, B, C, D\}$

$DB \rightarrow CB$

$AB \rightarrow DB$

$AB \rightarrow AC$

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

not candidate key

so

$$\boxed{\begin{array}{l} \text{candidate keys} \Rightarrow AB,\ BD,\ C \\ \text{prime attributes} = \{A, B, C, D\} \\ \text{non-prime attributes} = \{E, F\} \end{array}}$$

Alternate path:

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

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

without using transitive rule, we are left with 3 attributes (earlier there were only 2).

there were only 2).

check all proper subsets of BDE -

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

super key + candidate key also

BD, then C, then AC

[same answer irrespective of path taken]

NORMALIZATION

09 October 2023 21:55

student

SID	SName	credits	Dept Name	Building	Room No
1	Rahul	5	CSE	B1	101
2	Jiya	8	CSE	B1	101
3	Jenny	9	FD	B2	201
4	Payal	9	FD	B2	201
5	Ankur	7	civil	B1	110
6	Aakash	7	ECE	B1	115
7	Vanshika	8	civil	B1	101
8	Tanshika	7	CSE	B1	101

Dept Building & Room no → repeated in all tuples

↳ large schema leads to more redundancy.

Problems that occur due to redundancy

1. insertion anomaly
2. updation anomaly
3. deletion anomaly

Anomaly : when a data has multiple copies, and we update the data at one location but forget to update on location.

one location but forget to update that data at another location.

Insertion Anomaly

Eg: only a new Dept MME with Building B1 and Room 120 needs to be Inserted. But it is not possible as there is no student yet.

Updation Anomaly

Updation @ many locations

eg if CSE Room ~~101~~ → 301 change updating it in one location & forgetting to update it in another → leads to data inconsistency.

Delete Anomaly

If only one student is there in a dept & the student leaves, when that tuple is deleted, simultaneously all data of that department is also lost.

SOLN: Decomposition into 2 Tables

SID	Sname	Credits	Dept name

Dept name → Foreign key

key

SID	Sname	Credits	Dname
1	Rahul	5	CSE
2	Jiya	8	CSE
3	Jenny	9	FD
4	Payal	9	FD
5	Ankur	7	civil
6	Aakash	7	ECE
7	Vanshika	8	civil
8	Tanishka	7	CSE

(Primary key) Dept Name	Building	Room No
CSE	B1	101
civil	B1	110
FD	B2	201
ECE	B1	115

[Tables are related to each other using FOREIGN KEY]

(1) Insertion Anomaly → solved

we can insert a new dept without adding any student.

adding any ...
(MME, B1, 120)

② Update Anomaly → solved

only one tuple needs to be updated, eg while changing the room no. of a department.

③ Delete Anomaly → solved

Deleting student data doesn't erase information about the department

Normalization

Process of making table free of insertion, updation and deletion anomalies. It also helps save space by reducing redundant data

But it is not always the best idea to normalize data. Sometimes, denormalisation is better.

...actions → OLTP → normalised data ✓.

transactions ← OLTP → normalised data.

OLAP → denormalised data ✗

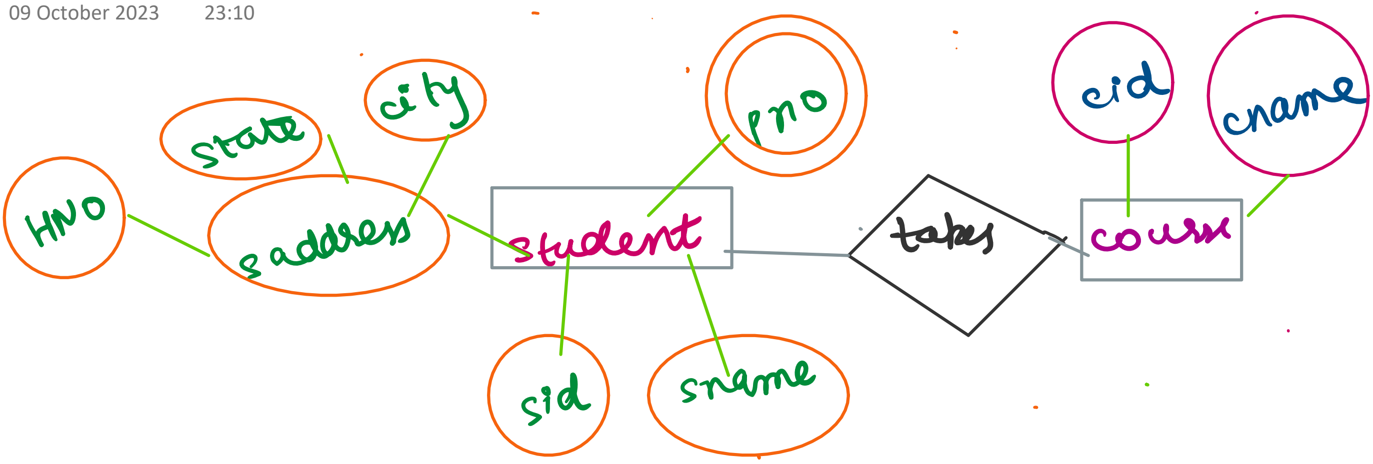
analytics
↓
joining of tables necessary
↳ instead, have original table without normalising it

Advantages of Normalization:

1. removal of anomalies
2. reduction of redundancy & saving of space
3. minimization of NULL values
4. easier understanding of table schema

FIRST NORMAL FORM

09 October 2023 23:10



student

SID	SName	SAddr	PNO
1	Jenny	Haryana India	P1 P2
2	Jiya	Punjab India	P3
3	Payal	Rajasthan India	P4, P5
4	Shanvi	Haryana India	P7

First Normal Form (1NF) :
A table is said to be in 1NF iff all attributes have atomic domains.

$\underbrace{\text{atomic domains}}$ → indivisible units which cannot be further decomposed.

Rule-1

[each cell must contain exactly ONE value only]

For composite attributes: make separate attributes for each, eg state & country as 2 attributes

For multi-valued attributes: make one new tuple for each value in the multi-valued set.

SID	Sname	state	country	phone No
1	Jenny	HR	INDIA	P1
1	Jenny	HR	INDIA	P2
2	Jiya	PUNJAB	INDIA	P3
3	Payal	RAJ	INDIA	P4
3	Payal	RAJ	INDIA	P5

Rule-2

A column must contain values from the same domain.

$\underbrace{\text{same domain}}_{\text{of same type}}$

Rule-3

A column should have a unique name.

Each column should have a unique name.

Rule - 4

No ordering to the rows and columns must be applicable.

Rule - 5

No duplicate rows must be present.

Primary key

Here (sid, phone no)

Good ER diagram → Relational schema will be in 1NF by default

If not → bad design of ER diagram.

1NF is an essential property in a relational schema.

Decomposing the above table into 2 :

SID	SName	State	country
1	Jenny	HR	IN

1	Jenny	HR	IN
2	Jiya	PB	IN
3	payal	RJ	IN

foreign key

SNO	PNO
1	P1
1	P2
2	P3
3	P4
3	P1

SNO, PNO
↓
primary key

[many to one relationship exists]

Note : integers are atomic values, but a set of integers isn't.

123 ⟶ atomic
but 1, 2, 3 ⟶ non-atomic (can be split into 3 parts)

CS001 ⟶ non-atomic ⟶ CS, ⟶ 001

Having schema as :
(SNO, Sname, state, country, Phone1, Phone2)
↓
partially solves the problem, but it fails if one person has more than 2 phone numbers, and therefore is not recommended.

SECOND NORMAL FORM

10 October 2023 14:08

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

FD = { A → B , B → C , C → D , D → E }

[in overwhelming majority of situations, it will already be in 1NF]

A relation is said to be in 2NF if : ① It is in 1NF . ② No partial dependency exists in the relation.

Partial Dependency Format :

proper subset of any candidate key → non-prime attribute

For the given relation, A ~~BCDE~~ F

candidate key = AF = 1

super keys = AF(BCDE) combos = 16

$A^+ = ABCDE$
$F^+ = F$

so AF is a CK

prime attributes = { A, F } not in RHS only 1 CK

Non-prime attributes = { B, C, D, E }

Non-prime attributes = { B, C, . . . }

AF : A → B

F ✗

partial dependency

prime → non prime

Partial dependency is present. so the relation is not in 2NF.

R (A, B, C, D)

FD = { AB → CD, C → A, D → B }.

Finding candidate keys:

A

B

C

D

$\boxed{AB}$

AC

$\boxed{AD}$

$\boxed{BC}$

BD

$\boxed{CD}$

ABC

ABD

ACD

BCD

ABCD

4 candidate keys

Prime attributes = { A, B, C, D }

no non-prime attribute

↳ ✓ It is in 2 NF

R (A, B, C, D)

FD : { A → B, B → C, C → D }.

candidate key

A^+ = { A, B, C, D }

not in RHS

only ① into

candidate key A - ? ...

Prime attributes = { A } → not in only 1 candidate key

Non-Prime attributes = { B, C, D }

There is no proper subset of the candidate key A → so, definitely, the given relation is in 2NF.

R (A, B, C, D)

FD = { A → B, B → D }

candidate key : AC^+ = { A, C, B, D }

Prime attributes = { A, C } ← not in the RHS of any FD so it is the only candidate key.

Non-Prime attributes = { B, D }

A is a proper subset of AC.

A → B → non-prime attribute

partial functional dependency exits.

Therefore the relation is not in 2NF

therefore the relation ... not in 2NF

THIRD NORMAL FORM

10 October 2023 19:22

Student

PK SID	sname	DOB	state	country	pin code	Total credits
1	A	-	HR	IN	122001	-
2	B	-	HR	IN	122001	-
3	C	-	HR	IN	122001	-
4	D	-	PB	IN	123456	-

↳ this relation is in 2NF as the candidate key has only one attribute.

Based on the no. of attributes present in the candidate key, there are 2 types of keys :

1. simple candidate key (1 attribute)
2. composite key (≥ 2 attributes)

Partial dependency condition :

proper subset of candidate key → non-prime attribute

Even if a relation is in 2NF, it isn't considered to be good enough.

that can arise :

Pin 4 considered ...

Problems that can arise :

eg pincode → state, country

If pincode is changed → corresponding state and country also need to be changed.

↳ update anomaly exists.

also redundancy exits

soln : form another table

State	country	pincode
HR	IN	122001

cause of problem :

Non - Prime Attribute → Non - prime Attribute

↓

Transitive Dependency

A relation is said to be in 3NF iff

① It is in 2NF.

② no transitive dependency exists for non - prime attributes

A table is in 3NF iff. for each of its non-trivial functional dependencies, at least one of the following conditions holds:

1. LHS is super key.
2. RHS is prime attribute.

R(A, B, C, D)

FD : { A → B, B → C, C → D }

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

prime attributes : { A }

Non-prime attributes : { B, C, D }

candidate key has only a single attribute
↳ relation is in 2NF.

But B → C

non-prime attribute   non-prime attribute

same for C → D

so this relation is not in 3NF

1NF	2NF	3NF
✓	✓	✗

(ii) alternate approach :

(or) alternate approach :

A → B LHS is superkey ✓

× B → C × both conditions failed ⊗

× C → D × both conditions failed ⊗

Relation is not in 3NF

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

FD : { AB → CDEF, BD → F }

candidate key

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

↳ only candidate key

prime attributes = { A, B }

Non-prime attributes = { C, D, E, F }

AB → CDEF

full candidate key

BD → F

not a proper subset of candidate key

both are not partial dependencies.

∴ Relation is in 2NF.

AB → CDEF → no problem

non prime

$\underbrace{AB}_{\text{prime}} \rightarrow \underbrace{CDEF}_{\text{non prime}}$

$\underbrace{BD} \rightarrow \underbrace{F}_{\text{non-prime}}$ $\Rightarrow$ transitive dependency exists

mixture of prime & non-prime attributes ≡ considered as non-prime

Therefore the relation is not in 3NF.

1NF	2NF	3NF
✓	✓	✗

Advantages of 3NF over 2NF:

2NF had some anomalies (insert, update, remove) → all of them are removed in 3NF.

then it is a good database design.

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

FD : { A → B, B → C, C → D, D → A }

candidate keys :

AE, BE, CE, DE

AE , BE , CE , DE

prime attributes = { A, B, C, D, E }

Non - prime attributes = ϕ

↳ As there are no non - prime attributes , the relation is in 2NF and 3NF

BOYCE CODD NORMAL FORM

10 October 2023 20:00

Drawbacks in 3NF :

↓

when multiple overlapping candidate keys are present

AB BC CD

↓

To handle these cases, a stronger form of 3NF known as **BCNF** was developed.

Developed by E.F. Codd and Reymond F Boyce.

father of DBMS, also developed 1, 2, 3 NFs

A relation is said to be in BCNF iff :

1. It is in 3NF.
2. For each non-trivial functional dependency, the LHS is a super key.

R (A, B, C)   FD : { A → B, B → C, C → A }

R (A, B, C)   FD: { A → B, B → C, C → A }

candidate keys : A, B, C

prime attributes = { A, B, C }

Non-prime attributes = ϕ

then 2NF ✓   3NF ✓

A → B   B → C   C → A

SK ✓   SK ✓   SK ✓

↳ therefore the given relation is in BCNF ✓

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

FD: { A → BCDE, BC → ACE, D → E }

candidate keys : A, BC

Prime attributes = { A, B, C }

Non-Prime attributes = { D, E }

A → BCDE   BC → ACE   D → E

SK ✓   SK ✓   × SK

↳ not in BCNF

BC → ACE   D → E

A → BCDE

SK ✓

BC → ACE

SK ✓

D → E

× SK

× not prime

↳ not in 3NF

A → BCDE

not proper subset of CK

BC → ACE

D → E

↳ in 2NF ✓

Highest normal form of this relation : 2NF

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

f : { AB → CDE , D → A }

candidate keys : AB , BD

prime attributes = { A , B , D }

Non-prime attributes = { C , E }

AB → CDE

↳ SK ✓

D → A

↳ not SK ✗

↳ not in BCNF.

D → A

3NF ✓

AB → CDE

prime

D → A

prime

3NF ✓

AB → CDE

not a subset of ck

non prime

D → A

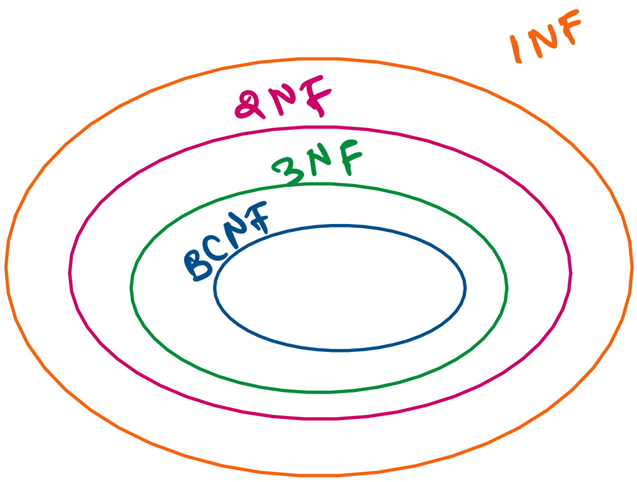
not a subset of ck

2NF ✓

Highest normal form : 3NF

HIGHEST NORMAL FORM OF A RELATION

10 October 2023 20:32



R (A B C D E F G H)

FD : { ABC → DE , E → GH , H → G, G → H ,
ABCD → EF }

candidate key : ABC

Prime attributes = { A , B , C }

non-prime attributes = { D, E, F, G, H }

checking for 2NF :

↳ no LHS has proper subset of ABC → so 2NF ✓

checking for 3NF :

↳ $\underbrace{E}_{\text{non-prime}} \rightarrow \underbrace{GH}_{\text{non-prime}}$ ⇒ transitive functional dependency exists ✗

So, the relation is in 2NF

R(ABCD)

FD: { AB → CD, AC → BD, BC → D }

Candidate keys: AB, AC

Prime attributes = { A, B, C }

Non-prime attributes = { D }

Checking for BCNF:

AB → CD
Sk ✓

AC → BD
Sk ✓

BC → D
✗ Sk

BCNF ✗

Checking for 3NF:

AB → CD
Sk ✓

AC → BD
✓ Sk

BC → D
✗ non-prime ✗

3NF ✗

Checking for 2NF:

AB → CD
not proper subset

AC → BD
not proper subset

BC → D
from AB, from AC
proper subset, non prime
partial dependency

2NF ✗

∴ The given relation is in 1NF only.

(proper subset un pen is also ... non ...)

∴ The given relation is in 1NF only. ("proper" in "proper subset of CK". also proper subset)

Points to Note :

① If all candidate keys are simple (single attributes) then it would be in 2NF.

② If all attributes of a relation are prime attributes, it would be in 3NF.

③ If relation is in 3NF & all candidate keys are simple, then it is in BCNF.

R(ABCDE)

FD : { AB → CDE, D → BE }

candidate keys : AB, AD

2NF :

AB → CDE

↳ 2NF ✓

D → BE

D: subset of AD

E: non-prime attribute

Relation is in 1NF

R(ABCDE)

FD : { AE → BC, AC → D, CD → BE, D → E }

candidate key : AC, AD, AE

AE → BC   AC → D   CD → BE   D → E

CD proper subset, BE non prime; D proper subset

2NF ✓ ✓ ✗ ✓

Not even 2NF.

Relation is in 1NF.

R(ABCD)

FD : { AB → C, ABD → C, ABC → D, AC → D }

candidate key : AB

2NF : AB → C   ABD → C   ABC → D   AC → D

PS ✗   PS ✗   PS ✗   PS ✗

no proper subsets of candidate key

AB → 2NF ✓

3NF : AB → C   ABD → C   ABC → D   AC → D

prime   non prime   non prime   non   non   non non

transitive dependencies

not in 3NF ✗

Relation is

∴ Relation is in 2NF

R(ABCD)

FD : { A → BCD , BC → AD, D → B }

candidate keys : A, BC, CD

all attributes are prime attributes → 3NF.

checking BCNF :

A → BCD   BC → AD   D → B

SK ✓   SK ✓   × not SK

not in BCNF ⊗

∴ the relation is in 3NF.

R(ABC)

FD : { A → B, B → AC }

candidate key : A, B

2NF :

A → B   B → AC

no proper subset of candidate key

↪ 2NF ✓

no proper subset ...

↪ 2NF ✓

3NF :

A → B (prime)

B → AC (prime)

no transitive dependency ✓

3NF ✓

BCNF :

A → B (A: sk ✓)

B → AC (B: sk ✓)

BCNF ✓

∴ Given relation is in BCNF

R(ABCDE)

FD : { A → BC, BC → AD, D → E }

candidate keys : A, BC

2NF :

A → BC

BC → AD

D → E

none are proper subsets of any candidate key, so it is in 2NF ✓

3NF :

A → BC (prime)

BC → AD (prime)

D → E (D: non prime, E: non prime)

prime prime non prime prime

transitive dependency

3NF ⓧ

∴ The given relation is in 2NF.

R(ABCDE)

FD : { A → B, BC → E, DE → A }

candidate keys : ACD, BCD, CDE

↳ As all attributes are prime attributes, the given relation is in 3NF.

BCNF:

A → B — SK ✓

BC → E — ✗ SK

DE → A — ✗ SK

ⓧ not in BCNF

therefore the given relation is in 3NF

R(ABCDE)

FD : { B → A, A → C, BC → D, AC → BE }

candidate keys : A, B

↳ all candidate keys are simple, 2NF.

↳ all candidate keys are simple,
so the relation is in 2NF.

3NF:

$\underbrace{B}_{\text{prime}} \rightarrow A$ $\qquad$ $\underbrace{A}_{\text{prime}} \rightarrow C$ $\qquad$ $\underbrace{\underbrace{BC}_{\text{non prime}} \rightarrow \underbrace{D}_{\text{non prime}} \qquad \underbrace{AC}_{\text{non prime}} \rightarrow \underbrace{BE}_{\text{non prime}}}_{\text{transitive dependencies exist}}$

3NF ⊗

∴ The given relation is in 2NF

DEPENDENCY PRESERVING DECOMPOSITION

11 October 2023 20:08

R(A, B, C)

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

R(ABCD) → R_1(AB), R_2(CD)

can't just decompose, need to follow some rules

① dependency preserving

② lossless

For given table

FD : { A → B, A → C }

Decomposing the table :

R_1 (A, B)

1	1
2	1
3	2
4	2

R_2 (B, C)

1	1
1	2
2	1
2	2

FD : { A → B } → F_1

no functional dependency → F_2

Logically $F_1 \cup F_2 = F$ must hold

here $F_1 \cup F_2 = \{ A \rightarrow B \}$

But here $F_1 \cup F_2 = \{A \to B\}$

↪ so this decomposition is not a dependency preserving relation

In general, if a relation R is divided into $R_1(F_1)$ $R_2(F_2)$ ... $R_n(F_n)$:

The decomposition is said to be dependency preserving if

$$F_1 \cup F_2 \cup F_3 \cup \ldots \cup F_n = F$$

$R(ABCDE)$

$F : \{A \to B, B \to C, C \to D, D \to A\}$

$R_1(ABC)$

$A^+ = \cancel{A}BC\cancel{D}$

$B^+ = A\cancel{B}C\cancel{D}$

$C^+ = AB\cancel{C}\cancel{D}$

$A \to BC$

$B \to AC$

$C \to AB$

(candidate keys)

$R_2(CDE)$

$C^+ = \cancel{A}B\cancel{C}D$

$D^+ = \cancel{A}\cancel{B}C\cancel{D}$

$E^+ = E$

$C \to BD$

$D \to C$

(candidate keys)

(Candidate keys)

no need to write other functional dependencies

Now performing $F_1 \cup F_2$

$F_1 \cup F_2 = \{ A \to BC,\ B \to AC,\ C \to ABD,\ D \to C \}$

check if every member of F is a member of $F_1 \cup F_2$

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

$\underbrace{F_1 \cup F_2}_{G} \equiv F$

check ← G covers F $G \supseteq F$

F covers G → always true

Note : while decomposing, it always will not lead to a dependency preserving decomposition.

Upto 3NF → dependency preserving

BCNF → exceptional case

Find the highest normal form :

R (ABCDE)

R (A B C D E)

F : { A → B, B → C, C → D, D → A }

candidate keys : AE, BE, CE, DE

↳ As all attributes in the relation are prime attributes, the relation is in 3NF.

A → B
× SK

B → C
× SK

C → D
× SK

D → A
× SK

↳ BCNF ×

R (A B C D E)

F : { A → BCD, B → AE, BC → AED, D → E, C → DE }

R₁ (A B)

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

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

↳ candidate keys

A → B

B → A

R₂ (B C)

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

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

↳ candidate key

B → C

$R_3(CDE)$

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

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

$E^+ = \cancel{E}$

$DE^+ = \cancel{D}\cancel{E}$

$C \rightarrow DE$

$D \rightarrow E$

$F_1 \cup F_2 \cup F_3$

$= \{ A \rightarrow B,\ B \rightarrow A,\ B \rightarrow C,\ C \rightarrow DE,\ D \rightarrow E \}$

G covers F ✓

$A^+ = ABCDE$

$B^+ = BACDE$

$BC^+ = BCADE$

$D^+ = DE$

$C^+ = CDE$

∴ Given decomposition is dependency preserving.

$R(ABCD)$

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

$R_1(AC)$

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

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

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

no functional dependency

$R_2(BD)$

$B^+ = \cancel{B}$

$D^+ = \cancel{D}$

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

no functional dependency

no further dependency

$F_1 = \phi$

dependency

$F_2 = \phi$

$\underbrace{F_1 \cup F_2}_{G} = \phi$ [only trivial FDs are present] But $F : \{A \to B, C \to D\}$

G does not cover F.

so given decomposition is not dependency preserving.

$R(ABCDE)$

$F : \{A \to BCDE,\ BC \to AED,\ D \to E\}$

$R_1(AB)$

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

$B^+ = \cancel{B}$

A B

$R_2(BC)$

$B^+ = \cancel{B}$

$C^+ = \cancel{C}$

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

ϕ

$R_3(CDE)$

$C^+ = \cancel{C}$

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

$E^+ = \cancel{E}$

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

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

$DE^+ = \cancel{D}\cancel{E}$

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

$D \to E$

performing union

$G = \{A \to B,\ D \to E\}$

$F = \{A \to BCDE,\ BC \to ADE,\ D \to E\}$

$F = \{ A \to BCD\ldots, \; BC \ldots$

$D \to E \}$

$A^+ = AB \qquad BC^+ = BC \qquad \underline{\underline{D}}^+ = D\underline{\underline{E}}$

clearly G does not cover F

$\therefore$ The given decomposition is not dependency preserving.

R (A B C D)

$F : \{ A \to B, \; B \to C, \; C \to D, \; D \to A \}$

$R_1 (A B)$

$A^+ = \cancel{A} B \cancel{C} \cancel{D}$

$B^+ = A \cancel{B} \cancel{C} \cancel{D}$

$A \to B$

$B \to A$

$R_2 (B C)$

$B^+ = \cancel{A} \cancel{B} C \cancel{D}$

$C^+ = \cancel{A} B \cancel{C} \cancel{D}$

$B \to C$

$C \to B$

$R_3 (C D)$

$C^+ = \cancel{A} \cancel{B} \cancel{C} D$

$D^+ = \cancel{A} \cancel{B} C \cancel{D}$

$C \to D$

$D \to C$

$$G = \{ A \to B, \; B \to A, \; B \to C, \; C \to B, \; C \to D, \; D \to C \}$$

$\underline{A}^+ = A \underline{BC} D \qquad \underline{B}^+ = B \underline{C} DA$

$\underline{C}^+ = C \underline{D} AB \qquad \underline{D}^+ = D \underline{A} BC$

G covers F.

$\ldots$ is

G covers F.

so, the decomposition is dependency preserving.

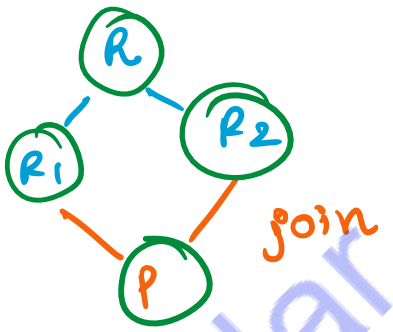
LOSSLESS JOIN DECOMPOSITION | ADDITIVE JOIN DECOMPOSITION PROPERTY

11 October 2023 22:16

lossless join → mandatory property for decomposition.

R(ABC)

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



Is R = P ?

If yes → lossless join decomposition

R_1(AB)

A	B
1	1
2	1
3	2
4	3

R_2(BC)

B	C
1	1
1	2
2	1
3	2

R_1(AB)

A	B
1	1
2	1
3	2

R_2(BC)

B	C
1	1
1	2
2	1

cartesian product $R_1 \times R_2$

3×3 = 9 rows are formed

A	B	B	C
1	1	1	1

A	B	D	C
1	1	1	1
1	1	1	2
⋮			

3×3 = 9 rows are formed

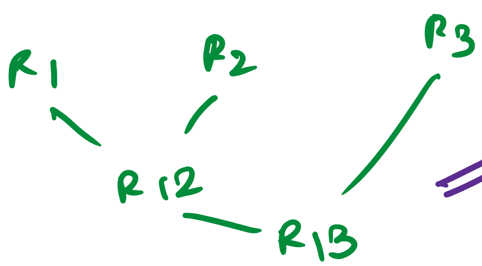
Natural join $R_1 \bowtie R_2$

→ consider only those rows where $R_1.B = R_2.B$ while joining

A	B	C
1	1	1
1	1	2
2	1	1
2	1	2
3	2	1

→ only 5 rows in natural join

Note: If no common attributes are present between the relations, the result of natural join will be the same as that of the cartesian product.



⇒ [binary operation, only take 2 at a time]

R(ABC)　　R_1(A B)　　R_2(B C)

R(ABC)

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

1	1
2	1
3	2
4	3

1	1
1	2
2	1
3	2

Now $R_1 \bowtie R_2$:

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

6 tuples

4 + 2 extra

not allowed

Lossless decomposition must be leformed.

↳ EXACTLY SAME

∴ Given decomposition is not lossless. [lossy]

R(A,B,C)

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

R_1(A B) R_2(B)

this decomposition is lossless (C missing)

C is lost while joining

For a lossless join decomposition :

① att(R_1) ∪ att(R_2) = att(R).

② att(R_1) ∩ att(R_2) ≠ φ

③ att(R_1) ∩ att(R_2) must be a ... one ...

③ att(R1) ∩ att(R2) must be candidate key of at least one sub-relation

att(R1) ∩ att(R2) → att(R1)

(OR)

att(R1) ∩ att(R2) → att(R2)

R(ABC)

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

R_1(AB)

A	B
1	1
2	1
3	2
4	2

super key A → B

R_2(AC)

A	C
1	1
2	2
3	1
4	2

super key A → C

$R_1 \bowtie R_2$:

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

∴ This decomposition is lossless

R(ABC)

A	B	C
1	1	3
2	1	3
3	2	6
4	3	7

R_1(AB)

A	B
1	1
2	1
3	2
4	3

R_2(BC)

B	C
1	3
~~1~~	~~3~~ duplicate
2	6
3	7

① AB ∪ BC = ABC ✓

② AB ∩ BC = B ✓

① ID

② AB ∩ BC = B ✓

③ checking if B is ck of R_1 : NO (or)
checking if B is ck of R_2 : yes ✓

↳ all properties are satisfied.

So the decomposition is ~~loss~~ lossless.

performing ⋈ and checking

$R_1 \bowtie R_2$

A	B	C
1	1	3
2	1	3
3	2	6
4	3	7

exactly same as R ✓

R (ABCDE)

A	B	C	D	E
1	1	2	1	3
2	2	2	1	3
3	1	6	3	6
4	2	8	5	7
5	3	9	5	7

① R_1(ABC) R_2(DE)
↳ no common attribute ✗ lossy

↳ no common ...

② $R_1(ABC)$ $R_2(CDE)$

↳ common attribute : C

For R_1 $C \to B$ ⊗ $C \to A$ ⊗ (OR) For R_2 $C \to D$ ✓ $C \to E$ ✓ ✓

✓ lossless

③ $R_1(ABC)$ $R_2(ABDE)$

↳ common attributes : A B

For R_1 $AB \to C$ ✓ For R_2 $AB \to DE$ ✓

✓ lossless

④ $R_1(AB)$ $R_2(BCDE)$

↳ common attributes : B

For R_1 $B \to A$ ⊗ R_2 : $B \to C$ ⊗ $B \to D$ ⊗ $B \to E$ ⊗

⊗ lossy

⑤ $R_1(AB)$ $R_2(CD)$ $R_3(DE)$

common attribute D

R_2 : $D \to C$ ✗ (or) R_3 : $D \to E$ ✓ ✓

decomposition into R_2 and R_3 is lossless

new $R_1(AB)$ $R_{23}(CDE)$

no common attribute

✗ lossy

$R(ABCD)$

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

$R_1(ABC)$ $R_2(CD)$

common attribute : C

$C^+ = ABC \to$ candidate key of R_1

✓ lossless

$R(ABCDEF)$

$F : \{ AB \to C,\ C \to D,\ D \to EF,\ F \to A,\ D \to B \}$

$D : \{ \underbrace{ABC}_{R_1},\ \underbrace{CDE}_{R_2},\ \underbrace{EF}_{R_3} \}$ [another form of representation]

R1 R2 R3 representation]

union of all attributes
= ABCDE ✓

Taking $R_1 \bowtie R_3$ → no common attribute

Taking $R_1 \bowtie R_2$ → C is common attribute

C^+ = CDEFAB → candidate key for both R_1 & R_2

R_1 R_2 is lossless, now join them

R_{12}(ABCDE) R_3(EF)

→ common attribute : E

E^+ = E
→ not a candidate key of R_{12} (or) R_3 ✗

∴ This decomposition is lossy.

Updating previous Qn
adding another FD : E → F

E^+ = EF → E is a candidate key of R_3

key of R_3

↳ lossless

↓

this decomposition is lossless

---

check if it is dependency preserving:

$R(ABCDE)$

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

$D : \{ ABC,\ CDE,\ EF \}$

$R_1(ABC)$

$A^+ = \cancel{A}$

$B^+ = \cancel{B}$

$C^+ = \cancel{C}\cancel{D}\cancel{E}\cancel{F}AB$

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

$C \rightarrow AB$

$AB \rightarrow C$

$R_2(CDE)$

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

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

$E^+ = \cancel{E}$

$C \rightarrow DE$

$D \rightarrow CE$

$R_3(EF)$

$E^+ = \cancel{E}$

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

ϕ

$G = F_1 \cup F_2 \cup F_3$

$G = \{ C \rightarrow AB,\ AB \rightarrow C,\ C \rightarrow DE,\ D \rightarrow CE \}$

ϕ

$D \to F$
$F \to A$ } are lost

$\underline{\underline{AB}}^+ = ABCDE$

$\underline{\underline{C}}^+ = CDEAB$

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

$F^+ = F$

G does not cover F.

$\therefore$ The decomposition is not dependency preserving.

CANONICAL COVER | MINIMAL COVER | IRREDUCIBLE SET OF FUNCTIONAL DEPENDENCIES

12 October 2023 07:16

F' is said to be a canonical cover of F if F' does not have
(i) extraneous / redundant attributes
(ii) redundant functional dependencies

Steps :

1. splitting rule so that in every functional dependency, right hand side has single attribute.
2. Remove extraneous attribute.
3. Remove redundant functional dependencies.

---

eg $AB \rightarrow C$, $A \rightarrow C$

B ↳ extraneous attribute

$AB \rightarrow C$ $A \rightarrow B$

↳ here also B is extraneous

extraneous ← $AB \rightarrow C$ $B \rightarrow C$

extraneous $AB \to C$

$AB \to C \qquad B \to A$

Redundant functional dependency

even after removing the functional dependency, it indirectly still exists

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

① split

$F: \{AB \to C,\ C \to A,\ C \to B,\ B \to C,\ \underbrace{ABC \to A}_{\text{trivial FD}},\ \underbrace{ABC \to C}_{\text{trivial FD}},\ A \to C,\ AC \to B\}$

②

extraneous

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

Note: also $AB \to C \qquad A \to C$

extraneous

canonical cover is not necessarily unique. more than one canonical cover may exist.

cover may exist.

$F = \{ B \to C,\ C \to A,\ C \to B,\ \underline{A \to C},\ \underline{AC \to B} \}$

extraneous

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

③ redundant FDs

To check if a FD is redundant or not, temporarily remove it and find $(LHS)^+$. If RHS is still formed, it means that the FD is redundant.

checking $C \to A$:

$C^+ = CB$ $\quad C \to A$ is not redundant

checking $C \to B$:

$\underline{C^+} = CA\underline{B}$ exists $\quad C \to B$ is redundant

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

checking $A \to C$:

$\underline{A^+} = AB\underline{C}$ $\quad A \to C$ is redundant

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

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

checking $A \to B$:

$A^+ = A$ ✗

$A \to B$ is essential

Finally

$F' = \{ A \to B, B \to C, C \to A \}$

→ minimal cover / canonical cover / irreducible set of FDs

2NF DECOMPOSITION

12 October 2023 07:34

R (A B C D)

F : { A → B , B → C }

candidate key : AD

2NF : $\underbrace{A}_{\text{proper subset of AD}} \rightarrow \underbrace{B}_{\text{non prime attribute}}$ ⇒ partial dependency

⊗ not in 2NF

[B → C is not a partial dependency]

A → B is a partial dependency.

LHS : A

Finding $A^+ = \underbrace{ABC}$ put these in one relation

In the other relation put remaining attributes + LHS

R_1 (ABC) R_2 (AD)

This is a lossless join decomposition.

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

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

$C^+ = \cancel{C}$

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

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

$D^+ = \cancel{D}$

ϕ

⇓

$BC^{+} = BC$

$A \to BC$
$B \to C$

As there are only 2 attributes, R_2 is in BCNF

Now $R_1(ABC)$ with $A \to BC$, $B \to C$

candidate key : A

As the candidate key is single valued, it is in 2NF.

checking for dependency preserving:

$G = F_1 \cup F_2 = \{ A \to BC,\ B \to C \}$

$A^+ = ABC$ $B^+ = BC$

G covers F

$R(ABCD) \to R_1(ABC)$, $R_2(AD)$ — lossless & dependency preserving

---

$R(ABCD)$

$F : \{ A \to B,\ C \to D \}$

candidate key : AC → only candidate key

2NF :

$A \to B$ (non / non)

$C \to D$ (PS / NPA)

$A \rightarrow D$

proper subset of AC — non prime — PS — NPA

Both are partial dependencies

Lossless ✓

Remaining + LHS into R_3

$A^+ = AB$ — $C^+ = CD$ — $R_3(AC)$

$R_1(AB)$ — $R_2(CD)$ — ϕ

$A \rightarrow B$ — $C \rightarrow D$

$G = F_1 \cup F_2 \cup F_3 = \{A \rightarrow B, C \rightarrow D\} = F$

↳ dependency preserving relation ✓

→ all subrelations have only two attributes → BCNF ✓

(or) checking rule by rule

$R_1(AB)$

CK : A

$\underbrace{A}_{\text{single}} \rightarrow B$ → 2NF ✓

$\underbrace{A}_{\text{prime}} \rightarrow B$ → not transitive 3NF ✓

$\underbrace{A}_{\text{super key}} \rightarrow B$ → BCNF ✓

$R_2(CD)$

CK : C

$\underbrace{C}_{\text{single}} \rightarrow D$ → 2NF ✓

CK : C

single

C → D
prime → not transitive
3NF ✓

C → D
super key → BCNF ✓

R3 (AC)
→ Functional dependencies ⇒ ϕ
→ BCNF ✓

∴ Decomposition is
R1 (AB) R2 (CD) R3 (AC)
It is in BCNF. It is lossless and dependency preserving.

2NF TO 3NF

15 October 2023 09:28

R(ABCD)

F : { A → B, B → C, C → D }

candidate key : A

B → C (B: non prime, C: non prime)

C → D (C: non prime, D: non prime)

] 2 transitive dependencies

(to ensure lossless)

Remaining + LHS

$C^+ = CD$

$R_3(ABC)$

$B^+ = BCD$

$R_2(CD)$

CK $R_1(BCD)$

CK $C^+ = \cancel{C}D$

CK $A^+ = \cancel{A}BC$

CK $B^+ = \cancel{B}CD$

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

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

$C^+ = \cancel{C}$

$D^+ = \cancel{D}$

C → D

BCNF

A → BC

B → C

$\underbrace{B \to CD, \; C \to D}_{\text{transitive dependency}}$

converted into 2NF. But it is not necessary that they should be in 3NF.

$R_1 \to 2NF$ $\quad$ $R_2, R_3 \to BCNF$

again apply the same steps for decomposition

$R_1(BCD)$ [C → D] causes problem

$R_1(BCD)$ $B \to CD$

$[C \to D]$ causes problem

$C^+ = CD$ remaining

$R_{11}(CD)$ $R_{12}(BC)$

$C \to D$ $B \to C$

union $\{B \to C, C \to D\}$ $\left.\begin{array}{l} B \to CD \\ C \to D \end{array}\right\}$ dependency preserving

clearly G covers F

Now $R_{11}(CD) \to$ BCNF

$R_{12}(BC) \to$ BCNF

$R_{11}(CD)$ $R_{12}(BC)$

duplicate

$R_2(CD)$ $R_3(ABC)$

Renaming :

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

relation gets converted to BCNF

Here it is lossless and dependency preserving. But sometimes while converting it into BCNF it may not be dependency preserving.

NORMALIZING A TABLE

15 October 2023 09:46

Normalize a relation from : 1NF → BCNF

R(ABCDEFGH)

F : { A → BD, B → C, E → FG, AE → H }

candidate key : AE → only candidate key

$\underbrace{A \to BD}_{\text{partial dependency}}$

$\underbrace{E \to FG}_{\text{partial dependency}}$

$A^+ = ABCD$

$E^+ = EFG$

D : { ABCD, EFG, AEH }

ABCD

ck $A^+ = \cancel{A}BCD$

$B^+ = \cancel{BC}$

$C^+ = \cancel{C}$

$D^+ = \cancel{D}$

A → BCD

$\underbrace{B \to C}_{\text{partial dependency}}$

EFG

ck $E^+ = \cancel{E}FG$

$F^+ = \cancel{F}$

$G^+ = \cancel{G}$

$\underbrace{E \to FG}_{\text{BCNF}}$

AEH

$A^+ = \cancel{A}\cancel{B}\cancel{C}\cancel{D}$

$E^+ = \cancel{E}\cancel{F}\cancel{G}$

$F^+ = \cancel{F}$

ck $AE^+ = \cancel{AEBD}\cancel{CFG}H$

$\underbrace{AE \to H}_{\text{BCNF}}$

partial dependency

2NF

D till now : { ABCD, EFG, AEH }

requires further decomposition

ABCD = F : { A → BCD, B → C }

candidate key : A

A → BCD

B → C

transitive dependency

$B^+ = BC$

D = { BC, ABD }

BC :

ck

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

$C^+ = \cancel{C}$

B → C

BCNF

ABD :

ck

$A^+ = \cancel{A}B\cancel{C}D$

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

$D^+ = \cancel{D}$

A → BD

BCNF

... decomposition for BCNF :

∴ Final decomposition for BCNF :

{ BC, ABD, EFG, AEH }