

DBMS & Warehousing
→ Transaction Management → Not in Gate DA

8-10 Marks

Database Management and Warehousing: ER-model, relational model: relational algebra, tuple calculus, SQL, integrity constraints, normal form, file organization, indexing, data types, data transformation such as normalization, discretization, sampling, compression; data warehouse modelling: schema for multidimensional data models, concept hierarchies, measures: categorization and computations.

→ Books — ① Navathe
② Korth
③ Raghurama Krishnan

Database — Collection of Data & their Relations

DBMS — Software package to manage database
Interface btw user & database

History of DBMS —
→ 1960's — used model → Network model
→ 1970's — IBM DBMS → Hierarchical model
→ E. Codd → Relational model was introduced in his research paper
1980's to till now

Relational model is one way to model, store & modify data in form of tables

→ Object oriented model has come, but relational model is still used in mainstream industry.

Problem with using ordinary file system

- Data retrieval for huge data into main memory
- Multivuser environment, access rights, consistent changes to data
- Quick access; Concurrent access
- Tends to duplicate data
- Version control or verify or review data transaction
- Tends to have inconsistent data
- Data integrity constraints & data validation
- Crash recovery

DBS → Database + DBMS

→ When people say DBMS, more than often they mean DBS

→ DBS Design

→ ① Requirement analysis
↳ Gather information

② Conceptual design phase
↳ Uses friendly model
eg ER model

③ Logical design phase
↳ Conceptual design gets converted to logical design

④ Implementation

Conceptual design

- User friendly
- eg — ER model — most popular
- Optional for simple design
- Temporarily used for design

Logical design

- Implementation friendly
- eg: — Relational model, Object oriented model — most popular
- Based on DBMS software

Data Model

- Conventions, rules, terminology etc, used to represent the data & relationship
- Model makes it easy to analyse, answer questions about data
- Analysing from real world is costlier than model
- Analyse data restrictions

RDBMS → Relational DBMS → based on Relational database model

Relational model

- proposed by E.F Codd in his research paper — Received Turing award.
- How to store data, terminology, definition
- 2 Query languages for relational model
 - Relational algebra
 - Relational calculus
- Theoretical Query Languages } IBM created commercial/practical Query language called SQL.
- Query language used for retrieval
- Concept of Normalisation, keys, Normal forms all proposed in this model

Table/Relation (Implementation friendly)

column name → Attribute

Name	Price	Category	Manufacturer	Storage Loc

→ each row represent one entry } Record or tuple

SQL to create table

Students (
 roll : string,
 name : string,
 age : integer,
 gpa : Real
)

Table name/Relation name

domain of attribute
 → Set of values that attribute can take

name of attribute

Terminology

- 1 Attribute
- 2 Record / Tuple
- 3 Relation
- 4 Domain
- 5 Schema vs instance
- 6 degree vs cardinality

Domain

→ A set of atomic values that a attribute can take

Indivisible

eg:-

Student name	DOB
Ravi Kumar	20-March-1998

INDIVISIBLE

Here Ravi Kumar is an atomic value ie cannot be further subdivided
 if we wanted to retrieve & analyse surname, we should have created separate attribute for Fname & Surname

eg:- Student Ph #
 Arun 334;555;222

In Relational model, Cannot put 2 elements of the domain in a cell } ER model, this is possible
 This character is not part of domain.

Schema vs instance → Snapshot

→ structure / outline

Student name	DOB
Ravi Kumar	20-March-1998

} instance / snapshot

→ Schema changes rarely
→ instance changes frequently

- ① Table name
- ② Attribute name
- ③ Attribute domain

Degree vs Cardinality → # of records in an instance

→ # of attributes in a schema

Why we call table relation?

Relation in Mathematics

binary relation → $R : A \rightarrow B$ A, B are sets

such that $R = \{ (a, b) \mid a \in A; b \in B \}$

⇒ $R \subseteq A \times B$ → All possible ordered pair
Relation is subset of $A \times B$

ternary relation

$R \subseteq A \times B \times C$

→ A, B, C are sets

$A \times B \times C$ → All possible 3-tuple

$R = \{ (a, b, c) \mid a \in A; b \in B; c \in C \}$

→ Cross product have every possible relation, it is like Superset
→ Relation is subset of cross product.

n-ary Relation

$R \subseteq A_1 \times A_2 \times A_3 \times \dots \times A_n$

$R = \{ (a_1, a_2, \dots, a_n) \mid a_i \in A_i \}$

eg:-

Student	Roll no.	Marks

$R \subseteq D_1 \times D_2 \times D_3$

- Table is subset of cartesian product of attribute domain.
- This is a relation, hence called relation

→ Observation

→ Every record belongs to
→ Every relation subset of
→ Every instance subset of

} Cartesian product of attribute domains } Also called cross product

Q) Given relation R & schema $|D_1| = 2, |D_2| = 3, |D_3| = 3$

a) Cardinality of largest instance possible = $|D_1| \cdot |D_2| \cdot |D_3| = 2 \cdot 3 \cdot 3 = 18$

b) Largest instance = $D_1 \times D_2 \times D_3$

c) #instances possible → each element of cross product can be selected or not selected
→ $2^{|D_1| \cdot |D_2| \cdot |D_3|} = 2^{18}$

Null Values

☆ schema $\neq$ empty relation = empty instance

NULL → not applicable
→ not known
→ absent (not recorded)

Student	Passport	Gender
abc	123	M
efg	NULL	F

→ information not available

Student	Husband	Gender
John	NULL	F

↓
not applicable

S4L

CREATE TABLE STUDENT

```

2 student_name string NOT NULL;
3

```

Keys

→ To retrieve data we need unique identification to find any record uniquely.

SUPERKEY

→ Superkey is the set of attribute that uniquely identifies any record
↳ can be singleton ⇒ Single attribute

eg:- ① Student roll no.
② Batch, year, name, DOB, email

→ Superset of Superkey is a superkey

Super key can have redundant attributes removing which also makes a super key.

CANDIDATE KEY → Must identify non-null keys uniquely.

→ A superkey from which if you remove any attribute, it will not be a superkey. (Minimal superkey). There can be multiple candidate keys.

eg:- ① Student_roll_no, Student_name $\rightarrow$ S key $\star\star$ CK is called Composite Key if it has more than one attribute.
 $\Rightarrow$ Student_roll_no $\rightarrow$ C key.

PRIMARY KEY

Unique at one time $\rightarrow$ Candidate key with minimum no. of attributes
 $\rightarrow$ One of the candidate key is selected as primary key $\} \text{By user or designer}$
 $\rightarrow$ There cannot be more than one primary key
 $\star \rightarrow$ We cannot have null value in primary key, however we can have repeated null values in candidate key & super key

$\star \rightarrow$ if multiple attributes makes a primary key, then no attribute of primary key should have null value.

$\rightarrow$ Remaining keys in the set of all candidate keys are called **Alternate keys**.

$\rightarrow$ Given a relation, there always exist a unique primary key.

ϕ In a relational model, can a complete tuple be null?
 $\rightarrow$ No, because for a relation, there exists always a primary key & primary key cannot have null for any attribute, so $\rightarrow$ No.

ϕ Which of the following is a possible set of all CK in relation $R = (a, b, c, d, e, f)$

$\checkmark$ a) $\{a, b, c\}$	$\checkmark$ d) $\{a, b, c, d, e\}$
$\times$ b) $\{f\}$	$\times$ e) $\{a, ab, bc\}$
$\checkmark$ c) $\{a, b, c, d, e\}$	$\checkmark$ f) $\{ab, bc\}$

ϕ In $R = (a, b, c, d, e)$ if abcde is known to be CK. Then what is the max, no. of possible CKs & Superkeys?
 $\star$ a) Superkey $\rightarrow 1$ b) Candidate key $\rightarrow 1$ } Since we cannot remove any attribute.

In $R = (a, b, c, d, e)$ what is max no. of CKs & SKs possible

SKs $\rightarrow 2^n - 1$ Any subset of (a, b, c, d, e) can be superkey (exclude empty set)

CKs $\rightarrow nC_{n/2}$ $\rightarrow$ n even
 $nC_{n-1/2} = nC_{n+1/2} \rightarrow$ n odd

ϕ $n \rightarrow$ fixed

$$\text{algmax}_K (nC_K) = \begin{cases} n/2, & n \text{ even} \\ n+1/2, & n \text{ odd} \end{cases}$$

$\star$ Given $R(a_1, a_2, \dots, a_n)$
 Max # of SKs $\Rightarrow$ Each $a_i = SK \Rightarrow \text{Max} = 2^n - 1$
 Max # of CKs $\Rightarrow$ Each element with $n/2$ or $n+1/2$ attributes is CK $\Rightarrow \text{Max} = nC_{n/2}$ or $nC_{(n+1)/2}$
 $n \rightarrow$ even $n \rightarrow$ odd

$\phi R(a, b, c, d, e)$; set of all CKs = $\{a, bc\}$

What is the max # of SKs.

Answer

- ① a concatenate any subset of $\{b, c, d, e\} = 2^4$ ways
- ② bc concatenate any subset of $\{a, d, e\} = 2^3$ ways
- ③ Subkey intersection $\Rightarrow$ abc concatenate any subset of $\{d, e\} = 2^2$ ways

$$\Rightarrow 2^4 + 2^3 - 2^2 = 16 + 8 - 4 = 20 //$$

$\phi R = \{a, b, c, d, e\}$ CK = $\{ab, bc, cd\}$ = set of all CKs. How many SKs max?

A = set of all SKs with $ab \rightarrow 2^3$

B = " $bc \rightarrow 2^3$

C = " $cd \rightarrow 2^3$

$$\Rightarrow |A \cup B \cup C| = 3 \cdot 8 - (4 + 4 + 2) + 2 = 24 - 8 = 16 //$$

$A \cap B \rightarrow$ set of SKs with $abc \rightarrow 2^2$

$B \cap C \rightarrow$ " $bcd \rightarrow 2^2$

$A \cap C \rightarrow$ " $acd \rightarrow 2$

$A \cap B \cap C \rightarrow$ " $abcd \rightarrow 2$

Sidquest

SQL $\rightarrow$ Started as a query language

Database definition language $\rightarrow$ DDL $\rightarrow$ Deals with schema

Database manipulation language $\rightarrow$ DML $\rightarrow$ Deals with data

SQL now does both.

- ① Create Table
- ② Drop Table
- ③ Alter Table
- ④ Specify domain of attributes
- ⑤ Integrity Constraints

datatypes

$\text{char}(n) \rightarrow$ Fixed len string

$\text{varchar}(n) \rightarrow$ Variable length string (max length n)

$\text{int} \rightarrow$ Integer

$\text{Smallint} \rightarrow$ Small integer

Machine dependent data size

SQL is not case sensitive



$\rightarrow$ keyword.

example

①

create table department

(dept_name $\text{varchar}(20)$,
building $\text{varchar}(15)$,
budget $\text{numeric}(12,2)$
primary key (dept_name));

② create table student

(sid $\text{char}(9)$,
name $\text{varchar}(20)$,
login $\text{char}(8)$,
age INTEGER ,
gpa REAL);

☆ SQL is not purely Relational → It is instead loosely adapted from Relational model → Therefore it can have null tuple, duplicate tuple etc.

③ CREATE TABLE branch

(branch_name CHAR(15) not null,
branch_city CHAR(30),
assets integer);

④ CREATE TABLE DEPT

(DNAME VARCHAR(10) NOT NULL,
DNUMBER INTEGER NOT NULL,
MGRSSN CHAR(9),
MGRSTART DATE CHAR(9),
PRIMARY KEY (DNUMBER),
UNIQUE (DNAME),
FOREIGN KEY (MGRSSN) REFERENCES EMP);

⑤ DROP TABLE DEPT;

⑥ ALTER TABLE branch
add location CHAR(20);

⑦ ALTER TABLE branch
drop location;

→ May not be supported
by some DBs

Data Manipulation

- ① Insert a record
 - ② Delete a record
 - ③ Modify a record
 - ④ Access or Query record
- not important for gate
Very Important for gate

Examples

① INSERT INTO students
Values (10211, Smith, John, Chemistry, 1996);
→ same order as declaration

② DELETE FROM student
where roll_number = '10211';

Integrity Constraints / Restrictions; (IC)

eg:- Age must be integer — DOMAIN INTEGRITY CONSTRAINT
Name should not be null; — NOT NULL CONSTRAINT
if district names are same, then state names should be same — FUNCTIONAL DEPENDENCY INTEGRITY CONSTRAINT
Rollno should be unique — KEY INTEGRITY CONSTRAINT

→ Integrity constraints are applied on schema

- Domain integrity constrain → Values should be from the domain set & atomic
- Key integrity constrain → Values should be unique [candidate key]
- not null integrity constrain → Value should not be null
- entity integrity constrain → Primary key should not have null value → Every relation should have primary key
- primary key integrity constrain → Not Null + Unique [primary key]

Foreign Key Integrity constrain (Referential IC)

→ There may be multiple relations/tables with different information related to each other.

Foreign key			Primary key				
cid	grade	sid	sid	name	login	age	gpa
Carnatic101	C	53831	50000	Dave	dave@cs	19	3.3
Reggae203	B	53832	53666	Jones	jones@cs	18	3.4
Topology112	A	53650	53688	Smith	smith@ee	18	3.2
History105	B	53666	53650	Smith	smith@math	19	3.8
Toc	A	630	53831	Madayan	madayan@music	11	1.8
			53832	Guldu	guldu@music	12	2.0

Enrolled (Referencing relation) Students (Referenced relation)

→ This student doesn't exist in Student relation
→ Hence violation

eg:- School DB

Student table (Referenced relation)		Enroll table (Referencing relation)		
sid	sname	sid	cid	grade
...	...	120		

120 (Primary Key) → refer to → 120 (Foreign Key)

→ We can say sid of enroll table refers to sid of student table.

eg2.

Enroll relation			Student relation	
cid	grade	sid	sid	...
TOL	A	120	120	
COA	B	120		
DBMS	B+	120		

Referenced multiple times

→ We call foreign key because the attribute is a key in foreign table

→ Foreign key must always refer to a candidate key in the foreign table

99.9% → Most of the time it is a primary key.

eg:-

Figure 5.6

One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

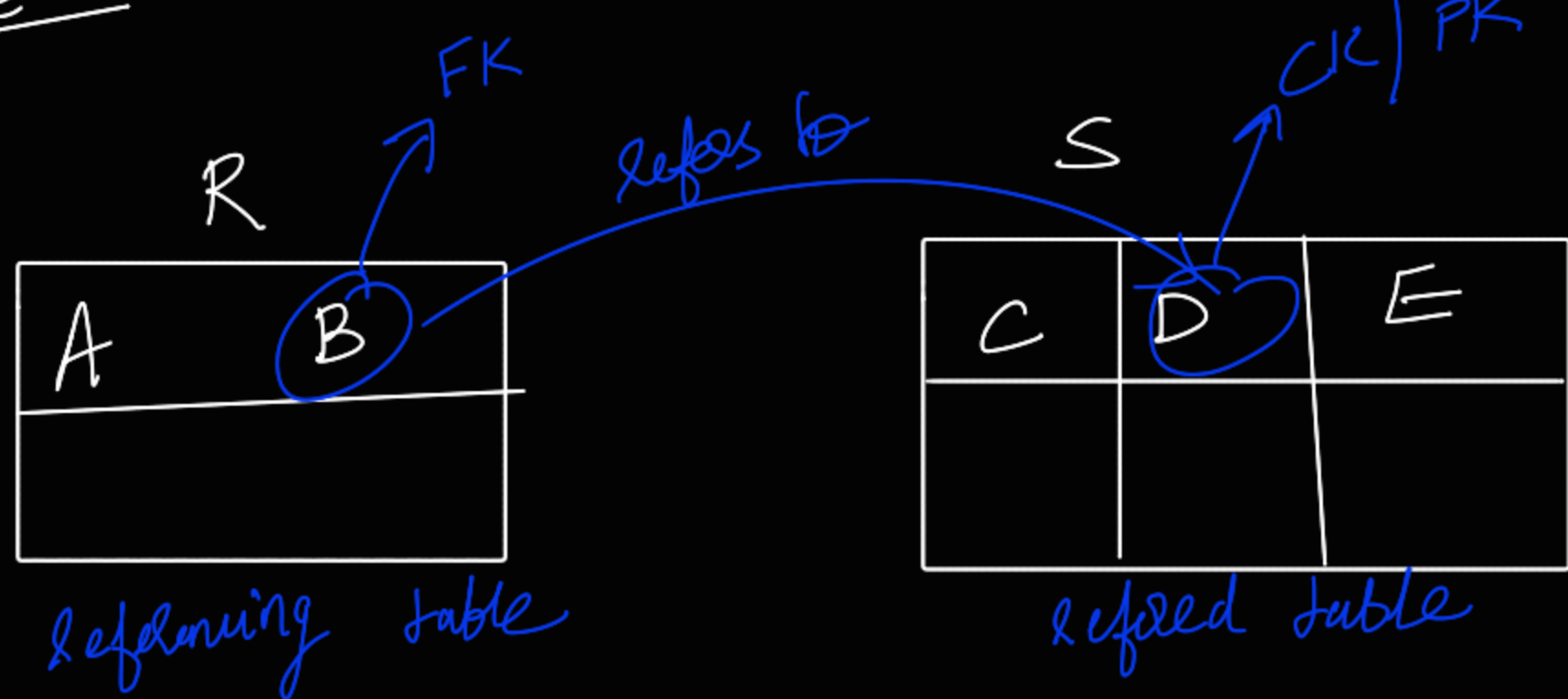
Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire

foreign key

→ Note

- * Dno is not a key in employee table
- * But it is a key in department table
- * Hence Dno is foreign key

Note:-



→ Observation

- Names can be different
- Domain must be same
- D should be a CK in S
- Every NON NULL value in B should exist in D
- B need not be a key in R
- B can have NULL values
- B need not be unique

Refer to same table

eg:- IISC

sid	status	Advisors
1	freshes	2
2	senior	NULL
3	freshes	2
4	senior	NULL
5	senior	NULL
6	freshes	5

Foreign key ☆☆☆

- Need not be unique
- Can be NULL
- Can refer to same table
- Can exist multiple foreign key in a relation
- A set of attributes can also be a foreign key. [In such cases, the entire combination should uniquely match and exist in the parent table.]

From point of view of a particular FK

Referring Table → child Table
Referenced Table → Parent Table

eg:- Army Database

Soldiers Relation

CK	Sol_name
100	

Family Relation

FK	family mem
sid	
100	son
100	chinta

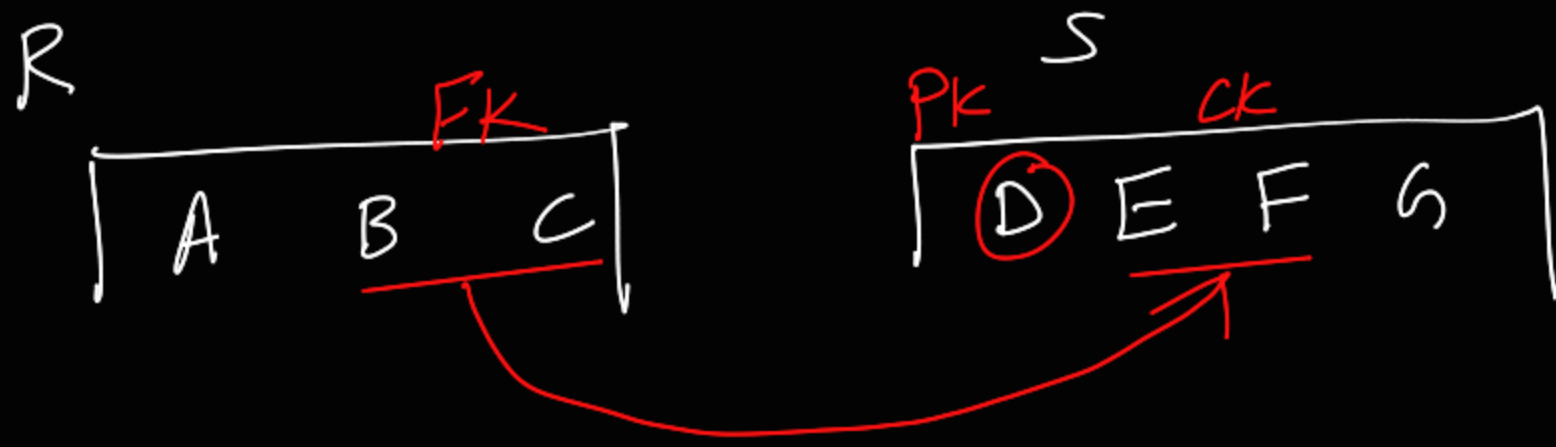
Family.sid refers to soldiers.sid

SQL

CREATE TABLE family
(sid char(3))

→ Refer to which attribute
→ if not explicitly defined → By default refers CK in parent
FOREIGN KEY (sid) REFERENCES soldiers

eg:-



CREATE TABLE R

(

FOREIGN KEY (B,C) references S(E,F)
);

→ should explicitly mention
→ otherwise by default
it considers PK.

Actions taken by DBMS to enforce FK integrity constraints

From any FK point of view

- ① Referencing table child
- ② Referenced table parent

make changes

insertion → No violation → Carry out
deletion
modification } → Violation may happen

- ① No action
* reject the operation
- ② Cascade
- ③ Set NULL
or set default

eg:

P	Q	R	S
2	2	2	2
3	8	8	3
7	3	3	2
5	8	9	7
6	9	5	7
8	5	7	2
9	8		

deletion causes violation

deletion does not cause violation

Cascade

→ Delete (or update) the children also in case of violation

P	Q	R	S
2	2	2	2
3	8	8	3
7	3	3	2
5	8	9	7
6	9	5	7
8	5	7	2
9	8		

Deletion causes violation

if update (8,3) to (10,3)
then all child tuples will be updated to refer to 10

These three tuples from child relation will also be deleted

ie
(3,8) → (3,10)
(5,8) → (5,10)
(9,8) → (9,10)

Set NULL or set default

P	Q	R	S
2	2	2	2
3	8	8	3
7	3	3	2
5	8	9	7
6	9	5	7
8	5	7	2
9	8		

Violation on delete
assume default = 2

will be changed to NULL or default

SQL example

CREATE TABLE orders (

Prod name VARCHAR(80)

REFERENCES products

on delete cascade,

on delete set null,

on delete set default,


```
CREATE TABLE Enrolled (
  sid CHAR(20),
  cid CHAR(20),
  grade CHAR(10),
  PRIMARY KEY (sid, cid),
  FOREIGN KEY (sid) REFERENCES Students
  ON DELETE CASCADE
  ON UPDATE NO ACTION )
```

From any FK point of view

① Referencing table *child* → violation → Reject the operation
 ↳ No other possibility

② Referenced table *parent*

3.11.1 Referential Integrity: GATE CSE 1997 | Question: 6.10. ISRO2016-54

Let $R(a, b, c)$ and $S(d, e, f)$ be two relations in which d is the foreign key of S that refers to the primary key of R . Consider the following four operations R and S

- Insert into R *No Viol*
- Insert into S ✓
- Delete from R ✓
- Delete from S *No Viol*

Which of the following can cause violation of the referential integrity constraint above?

A. Both I and IV
 B. Both II and III ✓
 C. All of these
 D. None of these

if instead of can, they ask will, then we cannot guarantee violation

$S(d, e, f)$ → $R(a, b, c)$

The following table has two attributes A and C where A is the primary key and C is the foreign key referencing A with on-delete cascade.

A	C
2	4
3	4
4	3
5	2
7	2
9	5
6	4

The set of all tuples that must be additionally deleted to preserve referential integrity when the tuple (2, 4) is deleted is:

A. (3, 4) and (6, 4)
 B. (5, 2) and (7, 2)
 C. (5, 2), (7, 2) and (9, 5) ✓
 D. (3, 4), (4, 3) and (6, 4)

Functional Dependency Integrity Constraint

Student relation

sid	sname	district	state
		Jaipur	
		Jaique	

must be same

Value of district determines the value of state

district $\xrightarrow{\text{determines}}$ state
 $\leftarrow \text{does not determine}$

Definition

∀ tuples t_1, t_2 (if $t_1 \text{ district} = t_2 \text{ district}$ then $t_1 \text{ state} = t_2 \text{ state}$)

We say, a) district functionally determines state
 b) state is functionally dependent on district
 c) district → state

eg 2:

IFSC code $\rightarrow$ Bank name
 Bank name $\not\rightarrow$ IFSC code
 SBI $\not\rightarrow$ SBINDOXXXX

eg 3:-

name $\not\rightarrow$ marks
 marks $\not\rightarrow$ name

name	Marks
John	30
John	20
John	20

eg 4:-

A $\rightarrow$ aadhar number
 B $\rightarrow$ pan card number

A $\rightarrow$ B
 B $\rightarrow$ A

$\rightarrow$ Functional dependency is defined on schema & hence, all possible instances must satisfy them.

eg:- In Quiz 30 for a particular course in a particular school.

sid	name	marks
100	a	30
101	b	30
102	c	40
103	a	30

a) same name $\Rightarrow$ same mark for this instance only

b) Not true for all instances

c) Coincidence for some instances cannot be treat as functional dependency.

☆ Functional dependency defined on schema not on instances

- Given an instance of $r(R)$ we say that the instance satisfies the functional dependency $\alpha \rightarrow \beta$ if for all pairs of tuples t_1 and t_2 in the instance such that $t_1[\alpha] = t_2[\alpha]$, it is also the case that $t_1[\beta] = t_2[\beta]$.
- We say that the functional dependency $\alpha \rightarrow \beta$ holds on schema $r(R)$ if, in every legal instance of $r(R)$ it satisfies the functional dependency.

Exercise 1 (2 pts)

Consider the relation instance and functional dependencies below. Which of these FDs hold on the instance and which do not?

A	B	C
0	2	4
0	3	4
1	3	5
0	2	4

$\times A \rightarrow B$ $\checkmark A \rightarrow C$ $\times B \rightarrow C$
 $\times B \rightarrow A$ $\checkmark C \rightarrow A$ $\times C \rightarrow B$

set of attributes can form a functional dependency

a) $\{A, B\} \rightarrow \{C, D\}$

b) $AB \rightarrow CD$ {short hand}
 $BA \rightarrow CD \rightarrow$ equivalent

1. (15 points) Consider a relation schema $R(A,B,C)$ and its relation instance as follows:

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

Which of the following functional dependencies are satisfied by the above relation instance. If the dependency is not satisfied, specify the tuples that cause the violation.

- 1). $AB \rightarrow C$ $\times$ (6,2,3), (6,2,5), (7,8,9), (7,8,5)
- 2). $A \rightarrow B$ $\checkmark$
- 3). $C \rightarrow A$ $\times$ (1,2,3), (6,2,3), (4,2,5), (6,2,5), (1,8,5)
- 4). $BC \rightarrow A$ $\times$ (1,2,3), (6,2,3), (7,2,5), (6,2,5)
- 5). $ABC \rightarrow A$ $\checkmark$
- 6). $AB \rightarrow AC$ $\times$

Recall that a *legal* instance of a relation must satisfy all specified ICs, including all specified FDs. As noted in Section 3.2, ICs must be identified and specified based on the semantics of the real-world enterprise being modeled. By looking at an instance of a relation, we might be able to tell that a certain FD does *not* hold. However, we can never deduce that an FD *does* hold by looking at one or more instances of the relation because an FD, like other ICs, is a statement about *all* possible legal instances of the relation.

¹ $X \rightarrow Y$ is read as X functionally determines Y , or simply as X determines Y .

3.5.15 Database Normalization: GATE CSE 2000 | Question: 2.24

Given the following relation instance.

X	Y	Z
1	4	2
1	5	3
1	6	3
3	2	2

Which of the following functional dependencies are satisfied by the instance?

- A. $XY \rightarrow Z$ and $Z \rightarrow Y$ $\times$
- B. $YZ \rightarrow X$ and $Y \rightarrow Z$ $\checkmark$
- C. $YZ \rightarrow X$ and $X \rightarrow Z$ $\times$
- D. $XZ \rightarrow Y$ and $Y \rightarrow X$ $\times$

Types of Functⁿ dependency

- ① TRIVIAL
- ② NON TRIVIAL FⁿD

eg:- $A \rightarrow A$ $\rightarrow$ Always hold in any relation involving A in all instances possible

eg:- $AB \rightarrow A$ $\rightarrow$ Trivial FⁿD

Defn:- $X \rightarrow Y$ is trivial FⁿD if X is a superset of Y or $Y \subseteq X$, X & Y are set of attributes

Side quest
Trivial

always true

NON TRIVIAL
 $\rightarrow$ an FⁿD which is not trivial if $Y \not\subseteq X$

eg- $R(A,B,C,D)$

- $\checkmark$ ① $A \rightarrow A$
- $\checkmark$ ② $A \rightarrow B$
- $\checkmark$ ③ $AB \rightarrow AC$
- $\checkmark$ ④ $AB \rightarrow A$
- $\times$ ⑤ $A \rightarrow BC$
- $\times$ ⑥ $A \rightarrow AB$
- $\times$ ⑦ $AB \rightarrow C$
- $\times$ ⑧ $ABCD \rightarrow AB$

Some functional dependencies are said to be **trivial** because they are satisfied by all relations. For example, $A \rightarrow A$ is satisfied by all relations involving attribute A . Reading the definition of functional dependency literally, we see that, for all tuples t_1 and t_2 such that $t_1[A] = t_2[A]$, it is the case that $t_1[A] = t_2[A]$. Similarly, $AB \rightarrow A$ is satisfied by all relations involving attribute A . In general, a functional dependency of the form $\alpha \rightarrow \beta$ is **trivial** if $\beta \subseteq \alpha$.

Completely Nontrivial $X \rightarrow Y$ is a completely non trivial FⁿD
 $\Rightarrow a \in Y \Rightarrow a \notin X$; i.e no common attributes
or $X \cap Y = \emptyset$

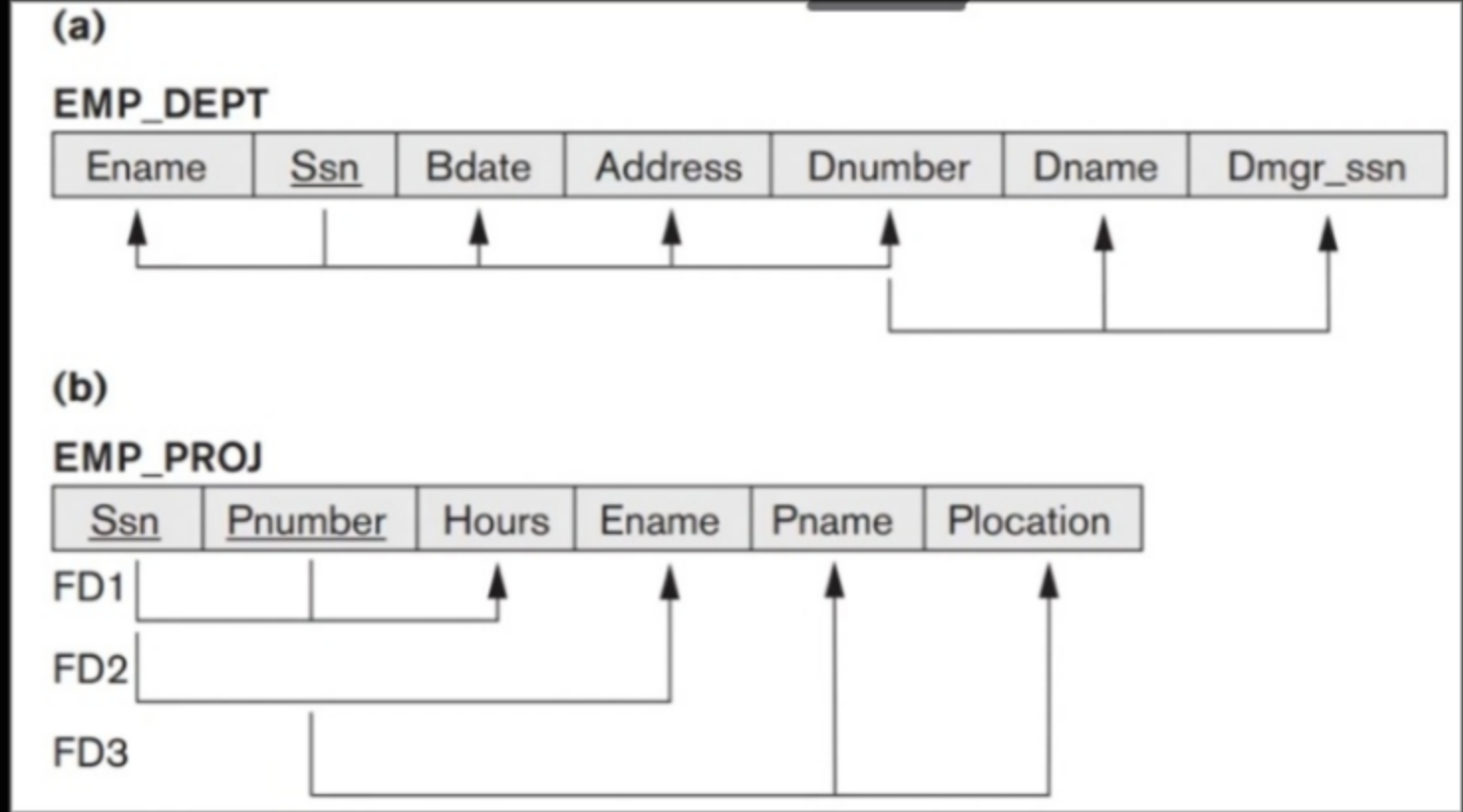
Trivial FⁿD $Y \subseteq X$
 $X \rightarrow Y$ Non Trivial FⁿD $Y \not\subseteq X$
Completely NT FⁿD $X \cap Y = \emptyset$

where X & Y
are set of
attributes

Another notation Diagramm representation
 $A \rightarrow B$ $R(A, B, C, D)$ $AB \rightarrow BC$ $R(A, B, C, D)$

$A \rightarrow BC$ $R(A, B, C, D)$

LHS $\rightarrow$ without arrow
 RHS $\rightarrow$ arrow
 each horizontal level denotes a FD



$ssn \rightarrow \{Ename, Bdate, Address, Dnumber\}$
 $Dnumber \rightarrow \{Dname, Dmgr_ssn\}$

FD1: $\{Ssn, Pnumber\} \rightarrow Hours$
 FD2: $Ssn \rightarrow Ename$
 FD3: $Pnumber \rightarrow \{Pname, Plocation\}$

For the relation Apply(SSN,cName,state,date,major), what real-world constraint is captured by $SSN, cName \rightarrow date$?

- ☐ A student can only apply to one college.
- ☐ A student can apply to each college only once.
- ☐ A student must apply to all colleges on the same date.
- ☒ Every application from a student to a specific college must be on the same date.

Consider a relation $R(A, B, C, D, E)$ with functional dependencies:
 $A, B \rightarrow C \rightarrow 3 \times 3 = 9$ max values for C
 $C, D \rightarrow E \rightarrow 9 \times 3 = 27$ max values for E
 Suppose there are at most 3 different values for each of A, B, and D. What is the maximum number of different values for E?

☒ 27
☐ 9
☐ 3
☐ 27

Functional dependencies $\rightarrow$ Laws [Armstrong's Axioms] / Armstrong's Inferences.

$\rightarrow$ Legal instance $\rightarrow$ satisfies all IC that we have defined over schema

Reflexivity Law
 if $Y \subseteq X$ then
 $X \rightarrow Y$

Augmentation Law
 if $X \rightarrow Y$
 then $XZ \rightarrow YZ$

Transitivity Law
 if $X \rightarrow Y$ & $Y \rightarrow Z$
 then $X \rightarrow Z$

$\rightarrow$ Proof 1
 assume $XZ \not\rightarrow YZ$
 $\Rightarrow$ Violation exist.

	X	Z	Y
t1	a	b	c
t2	a	b	d

 $\Rightarrow X \not\rightarrow Y$
 but contradiction

Proof 2:
 $X \rightarrow Y \Rightarrow XZ \rightarrow YZ$

	X	Z	Y
t1	a	b	c
t2	a	b	c
	a	d	c

Proof

X	Y	Z
a	b	c
a	b	c
a'	b'	c'
a'	b'	c'

Some more derived LAWS

④ split on RHS: if $X \rightarrow YZ$ then $X \rightarrow Y$ & $X \rightarrow Z$
Decomposition
proof $YZ \rightarrow Y$ (Reflexivity Law) $YZ \rightarrow Z$
 $X \rightarrow YZ \Rightarrow$
 $\Rightarrow X \rightarrow Y$ (Transitivity) $X \rightarrow Z$

⑤ Combining on RHS: if $X \rightarrow Y$ & $X \rightarrow Z$ then $X \rightarrow YZ$
Union

☆ Note: We cannot split on LHS; i.e. $XY \rightarrow Z \not\Rightarrow X \rightarrow Z, Y \rightarrow Z$
 eg:- $\{Pincode, H_num\} \rightarrow \{H_Loc\}$
 but $Pincode \not\rightarrow H_Loc$
 $H_num \not\rightarrow H_Loc$

⑥ Combine on LHS: if $X \rightarrow Z$ OR $Y \rightarrow Z \Rightarrow XY \rightarrow Z$
only one of this is enough

⑦ if $A \rightarrow B, BC \rightarrow D$ then does $AC \rightarrow D$

$A \rightarrow B$
 $\Rightarrow AC \rightarrow BC$
 $BC \rightarrow D$
 $\Rightarrow AC \rightarrow D \checkmark$

Pseudo transitivity.

⑧ T/F if $A \rightarrow B$ & $C \rightarrow D$ then $AC \rightarrow BD$?

$A \rightarrow B$ $C \rightarrow D \Rightarrow$
 $\Rightarrow AC \rightarrow BC$ $BC \rightarrow BD$
 $\Rightarrow AC \rightarrow BD \checkmark$
Composition.

⑨ T/F if $A \rightarrow B$ then $B \rightarrow A$ X

- Reflexivity: If $Y \subseteq X$, then $X \rightarrow Y$
- Augmentation: If $X \rightarrow Y$, then $XZ \rightarrow YZ$
- Transitivity: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- Union: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
- Decomposition: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
- Pseudotransitivity: If $X \rightarrow Y$ and $WY \rightarrow Z$, then $WX \rightarrow Z$
- Composition: If $X \rightarrow Y$ and $Z \rightarrow W$, then $XZ \rightarrow YW$

} Armstrong Axioms. } also called Armstrong Inferences

Given a non trivial FⁿD, we can split RHS, or use other laws, to get completely non trivial FⁿD.

eg $AB \rightarrow BC$
 $\Rightarrow AB \rightarrow B$ — trivial — Not interesting.
 $\Rightarrow AB \rightarrow C$ — completely non trivial
 $(LHS \cap RHS) = \emptyset$

CLOSURE OF SET ATTRIBUTE

In Relation R, set of FⁿDs S on R given,
 Now, let X be a set of attributes of relation R

then, closure of X = X^+ = set of all attributes that X can determine using

closure of X = $X^+ = \{Y \mid X \rightarrow Y \text{ is valid in } R\}$
 X^+ — sometimes denoted with * also

eg:- in relation R $\{A \rightarrow B, B \rightarrow C, D \rightarrow B\}$
 then $A^+ = \{A, B, C\}$ $B^+ = \{B, C\}$, $(BD)^+ = \{B, D, C\}$
 $D^+ = \{B, C, D\}$ $(AC)^+ = \{A, C, B\}$

7 Closure

Given a set of functional dependencies F, and a set of attributes X, the closure of X with respect to F, written X_F^+ , is the set of all the attributes whose values are determined by the values of X because of F. Often, if F is understood, then the closure of X is written X^+ .

$\emptyset$ $A \rightarrow B$ then $A^+ = \{A, B, C, D\}$
 $B \rightarrow C$ $BC^+ = \{B, C, D\}$
 $BC \rightarrow D$ $CD^+ = \{C, D\}$

★ Not subset of closure is not a closure valid FⁿD but

↳ In R(A, B, C, D, E, F) given FⁿDs : $AB \rightarrow C, BC \rightarrow AD, D \rightarrow E$ & $CF \rightarrow B$
 what is $\{AB\}^+$? $\{A, B\}^+ \rightarrow \{A, B, C, D, E\}$

3.5.25 Database Normalization: GATE CSE 2006 | Question: 70

The following functional dependencies are given:

$AB \rightarrow CD, AF \rightarrow D, DE \rightarrow F, C \rightarrow G, F \rightarrow E, G \rightarrow A$

Which one of the following options is false?

- A. $\{CF\}^+ = \{ACDEFG\}$ ✓
- B. $\{BG\}^+ = \{ABCDG\}$ ✓
- C. $\{AF\}^+ = \{ACDEFG\}$ ✗
- D. $\{AB\}^+ = \{ABCDG\}$ ✓

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

Keys in terms of FⁿD

→ if X is a superkey then X^+ is entire relation R
 → if X^+ is entire relation R then X^+ must be a superkey
 it can be a CK or PK but need not be.
 → if X does not repeat then X^+ gives entire relation R

① if a relation scheme R ; X is a SK if $X^+ = R$ and vice versa

R = set of all attributes;

X = set of any attribute

$$X \subseteq R$$

② In a relation scheme R ; X is a CK if $X^+ = R$; for any $Y \subset X \Rightarrow Y^+ \neq R$ and vice versa

Example. Consider a table $R(A, B, C, D)$ and that $F = \{A \rightarrow B, B \rightarrow C\}$.

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

- A is not a candidate key, because $A^+ = \{A, B, C\}$ which does not include D . *A is not even superkey.*
- ABD is not a candidate key even though $ABD^+ = \{A, B, C, D\}$. This is because $AD^+ = \{A, B, C, D\}$, namely, there is a proper subset AD of ABD such that AD^+ includes all the attributes.
- AD is a candidate key.

Q $R(A, B, C, D, E)$
 $F^D: \{AB \rightarrow C, AE \rightarrow D, D \rightarrow B\}$
 Find set of all CKs?

A & E is not on R.H.S
 $\therefore$ Any CK should include AE
 $\{AE\}^+ \rightarrow \{A, B, C, D, E\}$
 $\therefore$ No other possible

Q $R(A, B, C, D, E, F, G)$ with $F^D: AB \rightarrow C$
 $CD \rightarrow E$
 $EF \rightarrow G$
 $FG \rightarrow E$
 $DE \rightarrow C$
 $BC \rightarrow A$

Find all CKs.

★ if in the set of F^D s certain attributes do not appear on R.H.S, then these attributes must be part of superkey / candidate key

Soln Note B, F & D not on R.H.S
 $\Rightarrow$ Any CK should include BDF

① $\{B, D, F\}^+ = \{B, D, F\} \Rightarrow$ NO single or 2 or 3 attribute CK possible.

$$\textcircled{2} \{A, B, D, F\}^+ = \{A, B, C, D, E, F, G\} = R$$

$$\textcircled{3} \{C, B, D, F\}^+ = R \rightarrow \text{Since } BC \rightarrow A$$

$$\textcircled{4} \{E, B, D, F\}^+ = R \rightarrow \text{Since } DE \rightarrow C$$

$$\textcircled{5} \{G, B, D, F\}^+ = R \rightarrow \text{Since } FG \rightarrow E$$

All 4 attr CKs

★ if a X is not a superkey then any subset of X^+ is not a superkey

For asking # of superkeys the strategy is to find CKs first & then use inclusion exclusion to find their supersets

→ To find set of all CKs

① Find all mandatory attributes that do not appear on RHS. Set $n = \{\# \text{ mandatory attributes, otherwise } 1, \# \text{ Man. attr} = 0\}$

② Find all n letter CKs by fail & ellor

③ if any n letter CK is not a key, then rule out all possible subset of its closure

④ Any superset of CK is not a CK.

⑤ Increment n & continue,

Part 1 - Functional Dependencies and Normalization

1. 25% Given the relation:

- $R(A, B, C, G, H, I)$

And the functional dependencies:

- $A \rightarrow BCG$
- $BC \rightarrow GH$
- $B \rightarrow G$
- $G \rightarrow A$
- $C \rightarrow I$

Answer the following questions and show your work.

- Find $\{B\}^+$
- Find $\{G\}^+$
- What are the super keys?
- What are the candidate keys?

Super keys

$$\begin{aligned} B^+ &= R \\ G^+ &= R \\ A^+ &= R \end{aligned} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} \text{candidate keys}$$

$$\begin{aligned} C^+ &= I \\ H^+ &= H \\ I^+ &= I \end{aligned} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} \text{keys not possible using only their combination}$$

$$\begin{aligned} \# \text{ CKs} &= 2^5 + 2^5 + 2^5 \\ &\quad - 2^1 - 2^1 - 2^1 \\ &\quad + 2^3 \\ &= 56 \end{aligned}$$

1. Find all candidate keys for relation $R(ABCDE)$. Following functional dependencies exist in relation R :

$$A \rightarrow B, BC \rightarrow E, ED \rightarrow A$$

2. Given relation $R(ABCDEF)$ and following functional dependencies:

$$F = \{A \rightarrow BC, E \rightarrow ABC, F \rightarrow CD, CD \rightarrow BEF\}$$

Determine all candidate keys!

3. Given two sets of functional dependencies:

- $A \rightarrow B, A \rightarrow C, CD \rightarrow E, B \rightarrow D$
- $A \rightarrow BC, AD \rightarrow E, DB \rightarrow B, A \rightarrow D$

Solution

→ C, D do not appear on RHS of any F^+
→ hence any CK should have C, D

here $\{CD\}^+ = \{C, D\} \Rightarrow$ CK is at least 3 token

$$\{ACD\}^+ = \{A, B, C, D, E\} \rightarrow \text{CK} \quad \left. \begin{array}{l} \\ \end{array} \right\} \text{Candidate keys}$$

$$\{BCD\}^+ = R$$

$$\{ECD\}^+ = R$$

$\Rightarrow$ 4 token CKs not possible because it will be a superset of 3 token CK

Solution

→ All token appear on RHS of some F^+

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

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

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

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

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

$$F^+ = \{A, B, C, D, E, F\} = R \quad \underline{\text{CK}}$$

→ All 2 token CKs if any should not include F but should include D

$$\begin{aligned} \{AD\}^+ &= R^+ \\ \{CD\}^+ &= R^+ \\ \{ED\}^+ &= R^+ \end{aligned} \quad \left. \begin{array}{l} \\ \\ \end{array} \right\} \text{CKs.}$$

Consider the relation scheme $R = (E, F, G, H, I, J, K, L, M, N)$ and the set of functional dependencies

$$\{ \{E, F\} \rightarrow \{G\}, \{F\} \rightarrow \{I, J\}, \{E, H\} \rightarrow \{K, L\}, \{K\} \rightarrow \{M\}, \{L\} \rightarrow \{N\} \}$$

on R . What is the key for R ?

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

$$\rightarrow \{E, F, H\}^+ \rightarrow \{E, F, G, H, I, J, K, L, M, N\}$$

a) B must be part of every key.

$$B^+ = B$$

$$\{AB\}^+ = \{A, B, C, D\} \quad \text{CK} \quad \left. \begin{array}{l} \text{Only Cks} \\ \text{since Supersets} \\ \text{not CK.} \end{array} \right\}$$

$$\{DB\}^+ = R$$

$$\{CB\}^+ = R$$

For each relational schema given below and its corresponding functional dependencies (FDs):

- a. $R(A, B, C, D)$: $AB \rightarrow C, C \rightarrow D$ and $D \rightarrow A$
- b. $R(A, B, C, D)$: $AB \rightarrow C, BC \rightarrow D$ and $CD \rightarrow A$ and $AD \rightarrow B$
- c. $R(A, B, C, D, E)$: $AB \rightarrow C, C \rightarrow D, D \rightarrow B$ and $D \rightarrow E$

answer the following questions (for each of the above three cases separately):

1. (10 Points) find all candidate keys of the relation.

$$\left. \begin{array}{l} \{AB\}^+ \rightarrow R \\ \{BC\}^+ \rightarrow R \\ \{CD\}^+ \rightarrow R \\ \{AD\}^+ \rightarrow R \end{array} \right\} \text{only Cks}$$

c) A part of keys

$$\left. \begin{array}{l} \{AB\}^+ = \{A, B, C, D, E\} = R \\ \{AD\}^+ = R \\ \{AC\}^+ = R \end{array} \right\} \text{only Cks since} \\ \text{superset not possible}$$

5.20 Database Normalization: GATE CSE 2002 | Question: 2.25

From the following instance of a relation schema $R(A, B, C)$, we can conclude that:

A	B	C
1	1	1
1	1	0
2	3	2
2	3	2

- A. A functionally determines B and B functionally determines C
- B. A functionally determines B and B does not functionally determine C
- ☒ C. B does not functionally determine C
- D. A does not functionally determine B and B does not functionally determine C



2.3 Candidate Keys: GATE CSE 2014 Set 2 | Question: 21

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

$$2^3 = 8 //$$

An instance of a relational scheme $R(A, B, C)$ has distinct values for attribute A . Can you conclude that A is a candidate key for R ?

Coincidence $\neq$ F^{no}

No

Inferred F^{no}

$\rightarrow$ F^{no} that can be derived

eg:- FDset $F = \{a \rightarrow b, b \rightarrow c\}$
from F ; we derive $a \rightarrow c$ (inferred by F)

eg2. FDset $F = \{a \rightarrow b, b \rightarrow c\}$
from F ; we cannot infer $c \rightarrow a$ (can't be inferred)

eg:- Given $R(A, B, C, D, E, F)$
 $F: \{AB \rightarrow C, BC \rightarrow AD, D \rightarrow E, CF \rightarrow B\}$

Can we infer $AB \rightarrow D$?

$$\{AB\}^+ = \{A, B, C, D, E\} \quad \left. \begin{array}{l} \text{RHS is a} \\ \text{subset of} \\ \text{closure of LHS.} \end{array} \right\}$$

Method 1

Using F^{no} Laws $\rightarrow$ Among axioms & derived Laws

Method 2

☆ if $RHS \subseteq LHS^+ \rightarrow LHS \rightarrow RHS$ can be inferred
if $RHS \not\subseteq LHS^+ \rightarrow LHS \rightarrow RHS$ can't be inferred.

Consider the relation $R(A, B, C, D, E)$ and the set of functional dependencies $S1 = \{AB \rightarrow C, AE \rightarrow D, D \rightarrow B\}$. Which of the following sets $S2$ of FDs does NOT follow from $S1$?

- 10
- ☐ $S2 = \{AD \rightarrow C\}$
 - ☐ $S2 = \{AD \rightarrow C, AE \rightarrow B\}$
 - ☒ $S2 = \{ABC \rightarrow D, D \rightarrow B\}$
 - ☐ $S2 = \{ADE \rightarrow BC\}$

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

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

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

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

Notation:

Set of FDs = S

if FD " f " can be inferred from S then we write

$$S \models f \quad \left. \begin{array}{l} S \models f \\ \text{or } S \vdash f \end{array} \right\} \begin{array}{l} f \text{ follows from } S \\ S \text{ infers } f \end{array}$$

$S \not\models f \rightarrow S$ does not infer f

2. (15 points) Given a set of functional dependencies $F = \{AG \rightarrow B, B \rightarrow CD, BD \rightarrow E, CE \rightarrow F\}$ over $R = ABCDEFG$.

(a) Prove that $F \models AG \rightarrow BDF$ (This also means that F logically implies $AG \rightarrow BDF$, or $AG \rightarrow BDF$ can be deduced from F).

(b) Compute $(E)^+$. (X^+ is the set of all attributes A for which $X \rightarrow A$ can be deduced from F .)

(c) Find a key of R .

$$(AG)^+ = \{A, B, C, D, E, F, G\} = R$$

$\Rightarrow AG = CK$ (a) Yes it is inferred.

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

c) AG is a CK.

AG not on RHS $\Rightarrow AG$ part of every CK
 $\Rightarrow AG$ is already minimal $\Rightarrow$ only CK.

3.5.45 Database Normalization: GATE IT 2005 | Question: 70

In a schema with attributes A, B, C, D and E following set of functional dependencies are given

- $A \rightarrow B$
- $A \rightarrow C$
- $CD \rightarrow E$
- $B \rightarrow D$
- $E \rightarrow A$

Which of the following functional dependencies is NOT implied by the above set?

- A. $CD \rightarrow AC$
- B. $BD \rightarrow CD$
- C. $BC \rightarrow CD$
- D. $AC \rightarrow BC$

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

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

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

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

CLOSURE of FD set

$\rightarrow$ set of all FDs that can be derived.

FD set F

then

$F^+ =$ closure of FD set F

$$= \{f \mid F \models f\}$$

Sidenote

in FD $X \rightarrow Y$

X & Y are non empty
 always

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

$\left. \begin{array}{l} A \rightarrow B \\ A \rightarrow AB \end{array} \right\}$ They are not treated as same even though they have same completely non trivial forms.

$\downarrow$ Treat them as different

Example. Assume that there are 4 attributes A, B, C, D, and that $F = \{A \rightarrow B, B \rightarrow C\}$. Then, F^+ includes all the following FDs:

$A \rightarrow A, A \rightarrow B, A \rightarrow C, B \rightarrow B, B \rightarrow C, C \rightarrow C, D \rightarrow D, AB \rightarrow A, AB \rightarrow B, AB \rightarrow C, AC \rightarrow A, AC \rightarrow B, AC \rightarrow C, AD \rightarrow A, AD \rightarrow B, AD \rightarrow C, AD \rightarrow D, BC \rightarrow B, BC \rightarrow C, BD \rightarrow B, BD \rightarrow C, BD \rightarrow D, CD \rightarrow C, CD \rightarrow D, ABC \rightarrow A, ABC \rightarrow B, ABC \rightarrow C, ABD \rightarrow A, ABD \rightarrow B, ABD \rightarrow C, ABD \rightarrow D, BCD \rightarrow B, BCD \rightarrow C, BCD \rightarrow D, ABCD \rightarrow A, ABCD \rightarrow B, ABCD \rightarrow C, ABCD \rightarrow D.$

what is the size of F^+

Case 1: when $|LHS| = 1$
LHS has only one attribute. $\} = 7+3+1+1 = 12 \text{ FDs}$

LHS = A	LHS = B	LHS = C	LHS = D
$A^+ = \{A, B, C\}$ $3-1 = 7 \text{ FDs}$	$B^+ = \{B, C\}$ 3 FDs	$C^+ = \{C\}$ 1 FD	$D^+ = \{D\}$ 1 FD

Case 2: When $|LHS| = 2$
i.e. LHS has only 2 attributes

$AB^+ = \{A, B, C\} = 7 \text{ FDs}$
 $AC^+ = \{A, B, C\} = 7$
 $AD^+ = \{A, B, C, D\} = 15$
 $BC^+ = \{B, C\} = 3$
 $BD^+ = \{B, C, D\} = 7$
 $CD^+ = \{C, D\} = 3$
 $\} 42 \text{ FDs}$

Case 3: $|LHS| = 3$

$ABC^+ = \{A, B, C\} \rightarrow 7$
 $ABD^+ = \{A, B, C, D\} \rightarrow 15$
 $BCD^+ = \{B, C, D\} \rightarrow 7$
 $ACD^+ = \{A, B, C, D\} \rightarrow 15$
 $\} = 44 \text{ FDs}$

Case 4
 $ABCD^+ = \{A, B, C, D\}$
 $= 15$

$\Rightarrow$ Total of $42+12+44+15 = 113 \text{ FDs in } F^+$
 $\Rightarrow |F^+| = 113$

Covering

$\rightarrow$ FD set F given, FD set G given.
 $\rightarrow$ Does F cover G?

F covers G if F infers every FD in G

$F \Rightarrow G$
notation
F covers G
~~G covers F~~
if $G \subseteq F^+$
also implies
 $G^+ \subseteq F^+$
 $\} \rightarrow$ use this definition

Equivalence of FD set

$\rightarrow$ FD set given F
 $\rightarrow$ FD set given G

F equivalent G
 $F \equiv G$
 $\} \Rightarrow F \Rightarrow G \text{ \& } G \Rightarrow F$
 $F^+ = G^+$

$\rightarrow$ use this definition for solving

exercise to show that the following two sets of FDs are equivalent:

$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H\}$
and $G = \{A \rightarrow CD, E \rightarrow AH\}$

does $F \Rightarrow G$?

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

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

$\} A \rightarrow CD$
 $E \rightarrow AH \Rightarrow F \Rightarrow G$

does $G \Rightarrow F$?

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

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

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

$\} \Rightarrow G \Rightarrow F$

Yes they are

equivalent $F \equiv G$
 $F^+ = G^+$

Recap properties of Closure of F^+

Given FD set F

then $F^+ = \{f \mid f \text{ is a FD ; } F \models f\}$

- ① $F^+ \equiv F$
- ② F^+ is unique for all equivalent FD sets F, G, H
 $\hookrightarrow$ ie. $F^+ = G^+ = H^+$

Minimal Cover / Canonical Cover / Irreducible Cover

Given set of FDs $F \rightarrow$ Minimal Cover is the smallest set of FDs which is equivalent to F
 $F_m \rightarrow$ or in some literature they denote $F_c \rightarrow$ canonical

properties

① Need not be unique

② $F_m \equiv F \equiv F^+$

③ $F_m \subseteq F^+$

④ We can't remove any FD or attribute from F_m .
 $\hookrightarrow$ If we are able to do so without disturbing equivalence, then it is not a minimal cover.

eg:- $F = \begin{cases} E \rightarrow G \\ E \rightarrow S \\ G \rightarrow S \end{cases}$ is it minimum cover?
 $\rightarrow$ NO, we can delete $E \rightarrow S$

$G = \begin{cases} E \rightarrow G \\ G \rightarrow S \end{cases}$

$F \equiv G \rightarrow$ we cannot delete further FDs from G
 $\Rightarrow G$ is irreducible set of FDs.

Extraneous attributes

$\hookrightarrow$ Extra attribute on LHS of FD.

Definition: Set of FDs $F = \{XA \rightarrow Y, \dots\}$; FD: $XA \rightarrow Y$

$\rightarrow A$ is extraneous on LHS if $X \rightarrow Y \in F^+$

$\rightarrow$ In other words X^+ contains Y ; $Y \subseteq X^+$

Example FD set $F = \{AB \rightarrow C, A \rightarrow B\}$. Is B extra in $AB \rightarrow C$; is A extra in $AB \rightarrow C$

Solution $A^+ = \{A, B, C\} \Rightarrow C \in A^+ \Rightarrow B$ is extraneous in $AB \rightarrow C$

Define $G = \{A \rightarrow C, A \rightarrow B\}$, $F \equiv G$

$\rightarrow$ NO
 $B^+ = \{B\}$

4 Given $R(P, Q, R, S)$ FD set $F: \{ P \rightarrow QR, Q \rightarrow R, P \rightarrow Q, PQ \rightarrow R \}$

$$\begin{array}{l} P^+ = \{P, Q, R\} \quad R \subset P^+ \\ Q^+ = \{R, Q\} \quad R \subset Q^+ \end{array} \quad \left| \begin{array}{l} \text{We can say} \\ Q \text{ is extraneous in } PQ \rightarrow R \\ \& P \text{ is extraneous in } PQ \rightarrow R \end{array} \right.$$

☆ $\rightarrow$ It is wrong to delete both
 $\rightarrow$ We delete any one and then modify FD set & revisit

Q) Given FD set $F: \{ \text{AC} \rightarrow G, \text{ACD} \rightarrow B, BC \rightarrow D, CG \rightarrow B, CG \rightarrow D, CE \rightarrow A, CE \rightarrow G, D \rightarrow E, D \rightarrow G \}$

$\text{AC} \rightarrow G \rightarrow C^+ = \{C\}$ A not extraneous
 $\text{ACD} \rightarrow B \rightarrow \{AD^+\} = \{A, D, E, G\}$ C not extraneous.
 $BC \rightarrow D \rightarrow \{AC\}^+ = \{A, C, G, B, D, E\}$ DE $\text{AC}^+ \Rightarrow D$ extraneous.

Removing Redundant FD

① But first Make sure that RHS of every FD in FD set F has only single attribute.

$$\alpha \rightarrow AB \xrightarrow[\text{decomposing}]{\text{split of RHS}} \begin{cases} \alpha \rightarrow A \\ \alpha \rightarrow B \end{cases}$$

② To check if a given FD f is redundant or not $\rightarrow$ First remove out FD from F to get F' then from remaining FD set F' check whether
 $\{RHS\ f\} \subseteq \{LHS\ of\ f\}^+$

Alternative Definition $\rightarrow f$ is redundant if $\{F\} - \{f\} \equiv F$

$X \rightarrow Y$ is redundant in F if $F - \{X \rightarrow Y\} \models X \rightarrow Y$
 or $Y \subseteq X^+$ in $F - \{X \rightarrow Y\}$

$F1 = \{$	$\text{AC} \rightarrow G$	NR
$\div$	$CD \rightarrow B$	R
	$BC \rightarrow D$	NR
	$CG \rightarrow B$	NR
	$CG \rightarrow D$	
	$CE \rightarrow A$	
	$CE \rightarrow G$	
	$D \rightarrow E$	
	$D \rightarrow G\}$	

$\{AC\}^+ = \{A, C\} \rightarrow$ not redundant
 $\{CD\}^+ = \{C, D, E, G, B, A\} \rightarrow$ redundant
 $\{BC\}^+ = \{B, C\} \rightarrow$ not redundant
 $\{CG\}^+ = \{C, G, D, E, A\} \rightarrow$ not redundant

Notice here $CG^+ \neq \{C, G, D, E, A, B\}$ since we cannot use $CD \rightarrow B$ since we have already marked it redundant.

★ Finding Minimal Covers

- 1) Simplify the RHS $\rightarrow$ RHS has only one attribute (Decomposition)
 - 2) For all FDs on the LHS $\rightarrow$ Find a extraneous attribute
 - 3) Eliminate all the redundant FDs
- $\rightarrow$ order doesn't matter $\rightarrow$ can be mixed also
- $\rightarrow$ we can't delete any more attribute of any more FD
- $\rightarrow$ remember minimal FD is not unique.
- This order generally saves time $\downarrow$

$R(W, X, Y, Z)$ $F: \{X \rightarrow W, WZ \rightarrow Y, Y \rightarrow WXZ\}$

$X \rightarrow W$
 ~~$WZ \rightarrow X$~~
 $WZ \rightarrow Y$
 $Y \rightarrow W$
 $Y \rightarrow X$
 $Y \rightarrow Z$

$X \rightarrow W$
 $WZ \rightarrow Y$
 ~~$Y \rightarrow W$~~
 $Y \rightarrow X$
 $Y \rightarrow Z$

$Z^+ = \{Z\}$
 $W^+ = \{W\}$

$\{WZ\}^+ = \{W, Z, Y, X\}$
 without $WZ \rightarrow X$

$X \rightarrow W$
 $WZ \rightarrow Y$
 $Y \rightarrow X$
 $Y \rightarrow Z$

Final minimal cover.

$\{Y\}^+ = \{Y, X, Z, W\}$
 w/o $Y \rightarrow W$

Φ $F = \{A \rightarrow BC, B \rightarrow AC, C \rightarrow AB\}$

~~$A \rightarrow B$~~
 $A \rightarrow C$
 $B \rightarrow A$
 $B \rightarrow C$
 $C \rightarrow A$
 $C \rightarrow B$

$A \rightarrow C$
 $B \rightarrow A$
 ~~$B \rightarrow C$~~
 $C \rightarrow A$
 $C \rightarrow B$

$A^+ = \{A, C, B\}$
 w/o $A \rightarrow B$

$A \rightarrow C$
 $B \rightarrow A$
 ~~$C \rightarrow A$~~
 $C \rightarrow B$

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

Minimal cover F_{m1}

$C^+ = \{B, A, C\}$
 w/o $C \rightarrow A$

$\{A\}^+ = \{A\}$
 w/o $A \rightarrow C$

$B^+ = \{B, C, A\}$
 w/o $B \rightarrow C$

$B^+ = \{A, B, C\}$
 w/o $B \rightarrow A$

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

is also a minimal cover F_{m2}

Alternatively $\rightarrow$

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

F_{m3}

If several sets of FDs qualify as minimal covers of E by the definition above, it is customary to use additional criteria for minimality. For example, we can choose the minimal set with the smallest number of dependencies or with the smallest total length (the total length of a set of dependencies is calculated by concatenating the dependencies and treating them as one long character string).

Observation

- $\rightarrow$ # of FD's in minimal cover doesn't make sense
- $\rightarrow$ Minimal cover is not unique
- $\rightarrow$ # of FD's in one minimal cover may not be same as another
- $\rightarrow$ Minimal cover is not minimum. Better word is Irreducible set of FD's.

Korth Book

Navathe Book

if LHS is same
then merge

eg. $\{A \rightarrow B, A \rightarrow C\}$

$\Rightarrow \{A \rightarrow BC\}$

$\Rightarrow$ This is canonical
form without
merging they are
called minimal covers

Keep only one
attribute in RHS

eg. $\{A \rightarrow B, A \rightarrow C\}$

$\Rightarrow$ combine or not
both are minimum cover
both are canonical covers

Take care in MSQ

Given Relation R with n atomic attributes, how many completely non trivial functional dependencies are possible.

☆

Case 1: 1 term in LHS = $nC_1 \times (2^{n-1} - 1)$ possibilities
 || 2: 2 terms in LHS = $nC_2 \times (2^{n-2} - 1)$ ||
 ||
 Case n-1: n-1 terms in LHS = $nC_{n-1} \times (2^1 - 1)$ ||

$$\sum_{i=0}^n (nC_i) = 2^n \rightarrow \text{Binary story} \Rightarrow \sum_{i=1}^{n-1} (nC_i) = 2^n - 2$$

$$\rightarrow \sum_{i=1}^{n-1} (nC_i (2^{n-i} - 1)) = \sum_{i=1}^{n-1} (nC_i \cdot 2^{n-i}) + 2 - 2^n$$

$\rightarrow$ But what is this

$$\sum_{i=0}^n [nC_i 2^{n-i}] = 3^n$$

Team | substitute story

$$= \sum_{i=1}^{n-1} [nC_i 2^{n-i}] + (nC_0 \cdot 2^n) + (nC_n \cdot 2^0) = 3^n \Rightarrow \sum_{i=1}^{n-1} [nC_i 2^{n-i}] = 3^n - [2^n + 1]$$

$\therefore$ Our required answer $\sum_{i=1}^{n-1} (nC_i (2^{n-i} - 1)) = 3^n - 2^n - 1 + 2 - 2^n$

$$= \underline{\underline{3^n - 2^{n+1} + 1}}$$

$n = 4$

$$\Rightarrow 81 - 32 + 1$$

$$= 81 - 31 = 50$$

★ Prime attribute → It is a member of at least one CK.
 Non prime attribute → Not a member/part of any CK.

eg: $R(A, B, C)$ FD set $F: \{A \rightarrow B; B \rightarrow C\}$
 ① Find CKs
 $A^+ = \{A, B, C\} \rightarrow$ only CK.
 $\Rightarrow$ Prime attribute = A
 Non prime attribute = $\{B, C\}$

eg: $R(A, B, C)$ FD set $F: \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$
 ① Find CKs
 $A^+ = B^+ = C^+ = R$ } all CKs
 $\Rightarrow$ All are prime attributes

Trivial F^D vs non Trivial F^D

Trivial F^D
 always hold
 generally if $Y \subseteq X$
 then $X \rightarrow Y$ is trivial
 True for all relations

Non trivial
 $Y \not\subseteq X$
 Completely non trivial F^D $X \cap Y = \emptyset$
 Trivial F^D $Y \subseteq X$ } not interesting

eg: $XY \rightarrow YZ$
 non trivial
 $XY \rightarrow Z$ Completely non trivial part
 $XY \rightarrow Y$ trivial part

★ From now onwards we will consider only those non-trivial F^D with RHS $\rightarrow$ single attribute

Partial F^D vs Fully dependent

$\Rightarrow A$ is fully dependent BC
 if $BC \rightarrow A$ but
 $B \not\rightarrow A$
 $C \not\rightarrow A$

Fully functionally Dependent Definition: Full F^D

① $\alpha \rightarrow \beta$ is full F^D if $\forall A \in \alpha \{ \alpha - A \} \not\rightarrow \beta$
 ② $\alpha \rightarrow \beta$ is full- F^D if only if $\forall \alpha_s \subset \alpha \Rightarrow \alpha_s \not\rightarrow \beta$

Partial functional Dependency: Partial- F^D

① $\alpha \rightarrow \beta$ is partial F^D if for atleast one proper subset of α_s (i.e. $\alpha_s \subset \alpha$) $\alpha_s \rightarrow \beta$
 ② $\alpha \rightarrow \beta$ is partial F^D if $\exists \alpha_s \subset \alpha \mid \alpha_s \rightarrow \beta$
 $\hookrightarrow$ proper subset

#note
 $A \subset B \rightarrow$ proper subset
 $A \subseteq B \rightarrow$ subset

26/61

checking for partial or full $F^N D$
 $\hookrightarrow$ closure of all proper subsets.

eg:- $R(A, B, C, D, E)$ FD set $F: \{CD \rightarrow E, D \rightarrow A, A \rightarrow E\}$

$CD \rightarrow E?$ $C^+ = \{C\}$ $D^+ = \{D, A, E\}$	$\Rightarrow$ partial	$D \rightarrow A?$ full $F^N D$	$BCD^+ = \{E, A, B, C, D\} = R$ BCD is only CK.
---	-----------------------	------------------------------------	--

eg: $R(A, B, C) \leftrightarrow R(D, E) \rightarrow R, F$

$\downarrow$ means
 $F^N D$ both ways
 eg: $A \leftrightarrow B$
 $\Rightarrow A \rightarrow B$
 $B \rightarrow A$

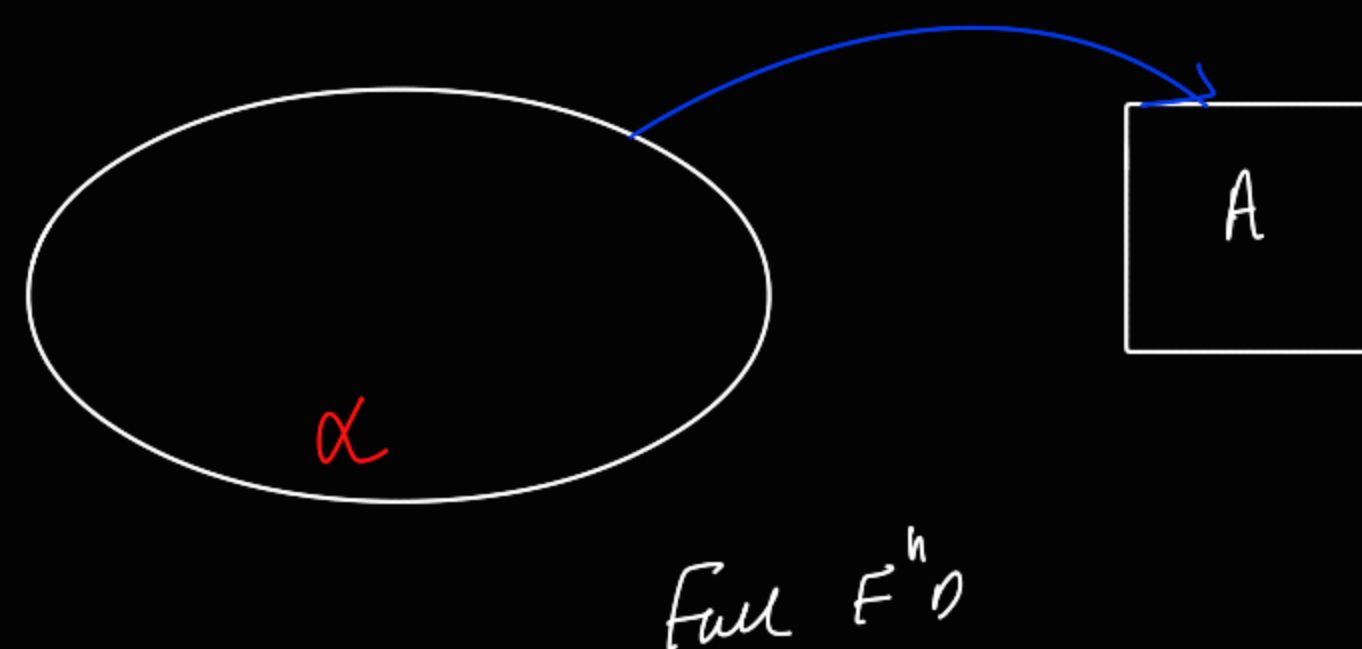
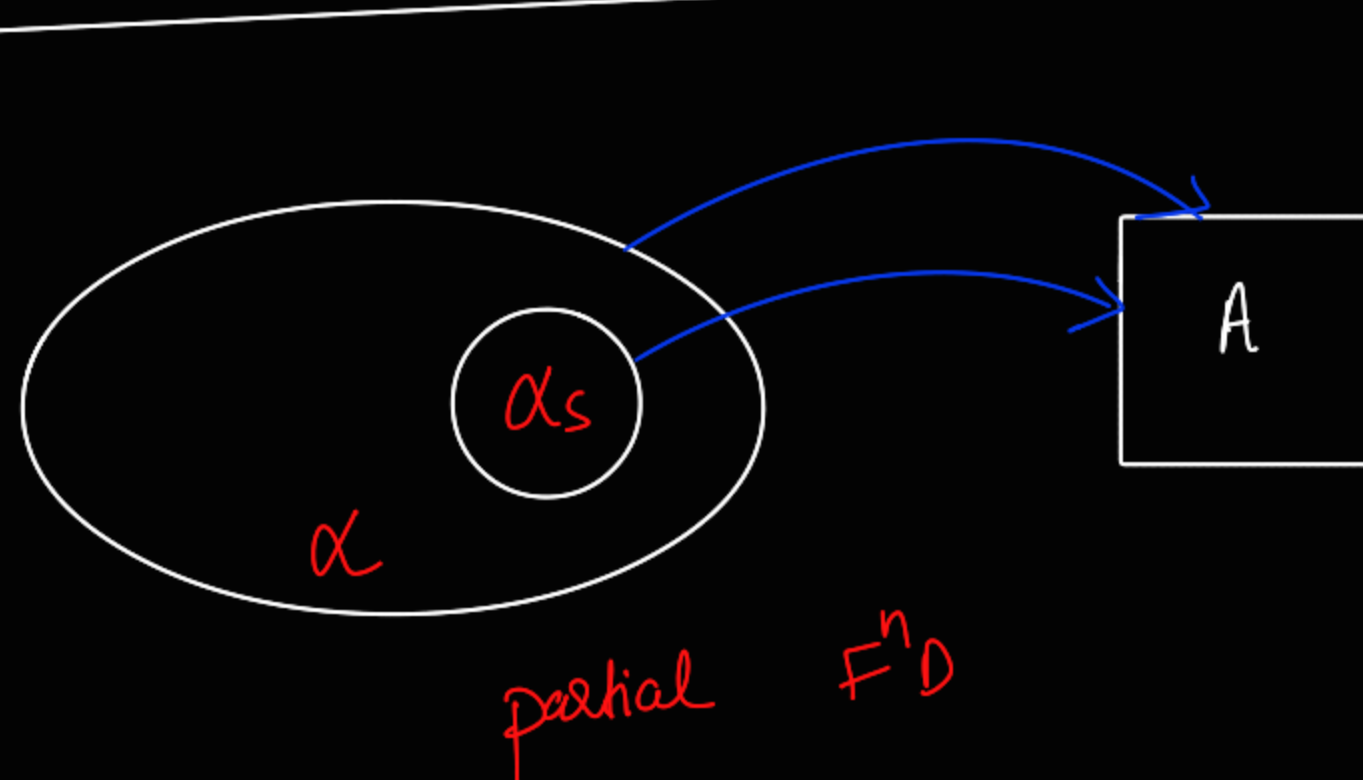
$R(A, B) \rightarrow R, D$

$R, E \rightarrow R, C$

$ABC \rightarrow DE$
 $DE \rightarrow ABC$
 $AB \rightarrow D$
 $DE \rightarrow F$
 $E \rightarrow C$

$\Phi) DE \rightarrow C?$
 $D^+ = \{D\}$
 $E^+ = \{E, C\}$
 $\Rightarrow$ partial

$\Phi) ABC \rightarrow D?$
 $AB \rightarrow D$
 $\Rightarrow$ partial



Transitive FD

$\alpha \neq \beta$ are set of attributes on some relation R . & A is a single attribute. then
 $\alpha \rightarrow A$ is transitive if $\exists \beta$ such that $\alpha \rightarrow \beta$, $\beta \not\rightarrow \alpha$, $A \notin \alpha, \beta$

Let α and β be sets of attributes such that $\alpha \rightarrow \beta$ holds, but $\beta \rightarrow \alpha$ does not hold. Let A be an attribute that is not in α , is not in β , and for which $\beta \rightarrow A$ holds. We say that A is transitively dependent on α .

$\alpha \rightarrow \beta \rightarrow A$
 $\nwarrow$
 $A \notin \alpha, \beta$
 ☆☆☆☆☆

eg:- $R(A, B, C); A \rightarrow C$

is $AB \rightarrow C$ transitive dependency

$AB \rightarrow A \rightarrow C \Rightarrow$ transitive dependency
 $\Rightarrow$ partial dependency also

but now $R(A, B, C); A \rightarrow C; A \rightarrow B$

then is $AB \rightarrow C$ transitive & partial
 $A \rightarrow C \Rightarrow$ partial but

$AB \rightarrow A \rightarrow C$ not transitive ☆

Observations for transitive dependency & partial dependency

- ① All partial dependencies are not transitive dependencies. Some can be
- ② if $\alpha \rightarrow A$ is a transitive dependency if and only if $A \not\rightarrow \alpha$
- ③ if $\alpha \rightarrow A$ is a partial dependency & α is a C.K then $\alpha \rightarrow W.S.K \rightarrow A$
is a transitive dependency.

eg:- $R(ABCD)$; FD set $\{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$

$A \rightarrow D$? $\rightarrow$ full FD (not partial)
 $\rightarrow$ Transitive since $A \rightarrow BC \rightarrow D$
 $BC \not\rightarrow A$
 $D \notin ABC$

Normal Forms

$\rightarrow$ A relation schema may be in some forms 1NF, 2NF, 3NF, BCNF

1NF $\rightarrow$ Domain must be atomic, Multiple values in a cell not allowed
 $\rightarrow$ In Relational model, every relation is in 1NF

2NF $\rightarrow$ A relation is not in 2NF if a CK can determine Non-prime attribute in a partial dependency.

$\rightarrow$ A relation is in 2NF if every CK key fully determines every nonprime attribute.

Violation: $\Rightarrow$ Some CK $\xrightarrow{\text{partial}}$ Some Non prime attr } Partial dependency.

Violation $\Rightarrow$ Some proper subset of any CK $\rightarrow$ Non prime Attribute

Side note

$\rightarrow$ check for 2NF

- ① Find all CK
- ② Find all non prime attribute
- ③ Check for violation

eg:- $R(ABCDEF)$ FD set $\{B \rightarrow F\}$ not in 2NF
 key = ABC

eg:- $R(ABCDE)$ $\{E \rightarrow A, C \rightarrow B, AB \rightarrow D\}$

① CKs = $(CE)^+ = CBEAD = R \rightarrow$ no other CKs

② nonprime = ABD

$CE \rightarrow A \rightarrow$ partial
 $CE \rightarrow B \rightarrow$ partial
 $CE \rightarrow D$

eg:- $R(ABCDE)$ $A \rightarrow B$; $AC \rightarrow B$; $D \rightarrow E$; $AC \rightarrow BDE$
 $\{A\}^+ = \{A, B, C, D, E\} \rightarrow R \rightarrow \text{only CK}$ | $AC \rightarrow B$ — partial
 $AC \rightarrow D$
 $AC \rightarrow E$
 non prime = B, D, E

Q) In 2NF can we have partial dependency?
 Yes Prime attribute can be partial dependent on CK
 Non prime attribute can be partial dependent on other Non prime's

Q) T/F R is in 2NF if some non prime attribute is fully Dep on all can. key
 False $\downarrow$ must be all
 $\rightarrow$ if A must be all
 $\rightarrow$ if given some then false.

☆ Q) if Relation R has no non prime attribute, then R is in 2NF? True

☆ Q) If R has no composite candidate key, then R must be in 2NF? True.

The following statements are all equivalent. They all define what 2NF is about.

- A relation is in 2NF if every nonprime attribute is fully functional dependent on every candidate key.
- A relation is in 2NF if there does not exist an FD $X \rightarrow Y$ such that Y is a nonprime attribute and X is a proper subset of a candidate key.
- A relation is in 2NF if there does not exist a nonprime attribute Y that is partially dependent on any candidate key.

Q) Can a non-prime attribute determine prime attribute?

No. because, if it can, then it would have been part of some CK.

Let $A \rightarrow B$

$A = \text{non prime} \Rightarrow A \notin \text{any CK}$

$B = \text{prime} \Rightarrow \exists X \mid XB = \text{CK}, A \neq X$

But $A \rightarrow B \Rightarrow AX \rightarrow XB \Rightarrow XA = \text{CK}$

Q) Can a non prime attribute determine another non prime? Yes

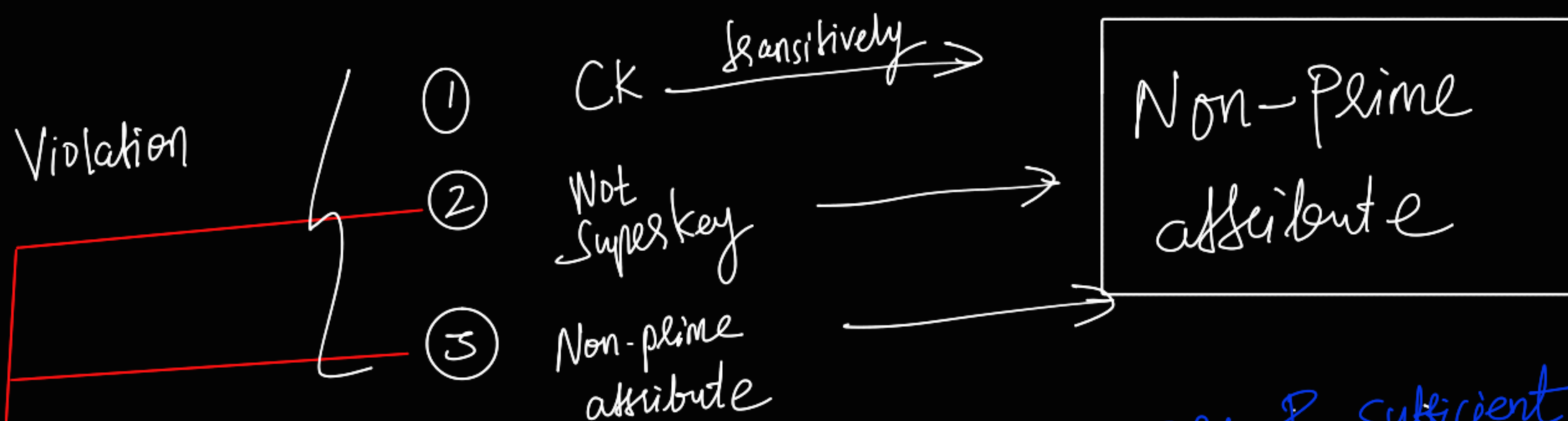
Q) Can a prime attribute determine another prime? Yes

3NF

Definition

- Every Non prime attribute is non transitively determined by all candidate key.
- No non prime attribute is transitively determined by any candidate key.

Violation $\rightarrow$ ① When not a superkey determines non-prime attribute
 ② When non prime attribute determines non-prime attribute



Note * Conditions one and two are necessary & sufficient but condition 3 is only necessary & not sufficient

why. $\rightarrow$ Let α be a non-super key then if $\alpha \rightarrow \text{non prime}$
 $\alpha \not\rightarrow \text{CK}$ since if $\alpha \rightarrow \text{CK}$ then α is a super key.

but $\text{CK} \rightarrow \alpha$ & $\alpha \rightarrow \text{non prime}$
 $\Rightarrow \text{CK} \xrightarrow{\text{transitively}} \alpha \rightarrow \text{non prime} \Rightarrow \text{CK} \xrightarrow{\text{transitively}} \text{non prime}$

③ Let A, B be non-prime attribute, & if $A \rightarrow B$, then
 $A \not\rightarrow \text{CK}$; since A is non prime & non prime $\not\rightarrow$ prime
 $\Rightarrow \text{CK} \rightarrow A \rightarrow B \Rightarrow \text{CK} \xrightarrow{\text{transitively}} B$

Property of 3NF

- if a CK partially determines non prime attribute then
 - 2NF violation
 - CK $\xrightarrow{\text{transitively}}$ non prime
 - 3NF violation

explanation

I) if CK $\xrightarrow{\text{transitively}}$ non prime, then let α be a proper subset of CK that determines non prime - ie $\exists \alpha \mid \alpha \subset \text{CK}; \alpha \rightarrow \text{non prime}$

II) Then $\alpha \not\rightarrow \text{CK}$ since if $\alpha \rightarrow \text{CK}$ then α is another CK $\rightarrow$ contradiction

III) Therefore $\text{CK} \rightarrow \alpha \rightarrow \text{non prime}$
 $\Rightarrow \text{CK} \xrightarrow{\text{transi}} \text{non prime}$



eg:- I) $R(ABCD)$ FD set $F: \{A \rightarrow C, B \rightarrow A, A \rightarrow D, AD \rightarrow C\}$
 $\rightarrow B^+ = \{A, B, C, D\} = R \Rightarrow B$ is the only CK $\Rightarrow$ 2NF
 $\rightarrow ACD \rightarrow$ non prime
 $\rightarrow B \rightarrow A \rightarrow D \Rightarrow B \xrightarrow{\text{transitivity}} D \Rightarrow$ 3NF violation

II) $R(ABCD)$
 FD set $F: \{C \rightarrow D, CD \rightarrow A, AB \rightarrow C, BD \rightarrow A\}$
 $\rightarrow B^+ = \{B\}$
 $AB^+ = \{A, B, C, D\} = R$
 $CB^+ = \{A, B, C, D\} = R$
 $DB^+ = \{A, B, C, D\} = R$
 $\rightarrow$ No non prime $\Rightarrow$ 2NF & 3NF

III) $R(ABCD)$
 FD set $\{A \rightarrow D, C \rightarrow A, D \rightarrow B, AC \rightarrow B\}$
 $\rightarrow C^+ = \{A, B, C, D\} \rightarrow$ only CK $\Rightarrow$ 2NF
 $\rightarrow ABD \rightarrow$ non prime
 $C \rightarrow A \rightarrow D \Rightarrow C \xrightarrow{\text{trans}} D \Rightarrow$ 3NF violation

IV) $R(ABCD)$
 FD: $\{BD \rightarrow C, AB \rightarrow D, AC \rightarrow B, BD \rightarrow A\}$
 $AB^+ = \{A, B, C, D\} = R$
 $BD^+ = R$
 $AC^+ = R$
 $\rightarrow$ No non prime attribute $\Rightarrow$ 2NF & 3NF

Q) if a relation is in 3NF, is it in 2NF. $\rightarrow$ True

if A then B
 if not B then not A } both are equivalent
 if not 2NF then not 3NF
 if 3NF then 2NF

Q) $R(ABCD)$ $B \rightarrow C$ $B^+ = \{C, B\}$
 $AC \rightarrow D$ $BD^+ = R$
 $ABD \rightarrow C$ $BA^+ = R$
 $BCD \rightarrow A$
 $\Rightarrow C \rightarrow$ non prime } non prime $\rightarrow$ non prime } however this is not sufficient condⁿ
 only one non prime
 but $B \rightarrow C \Rightarrow$ not a SK $\rightarrow$ non prime $\Rightarrow$ 3NF violation
 $BD \rightarrow C$ } partial $\Rightarrow$ 2NF violation
 $B \rightarrow C$

Q) if no non superkey $\rightarrow$ non prime then is it 3NF? True.

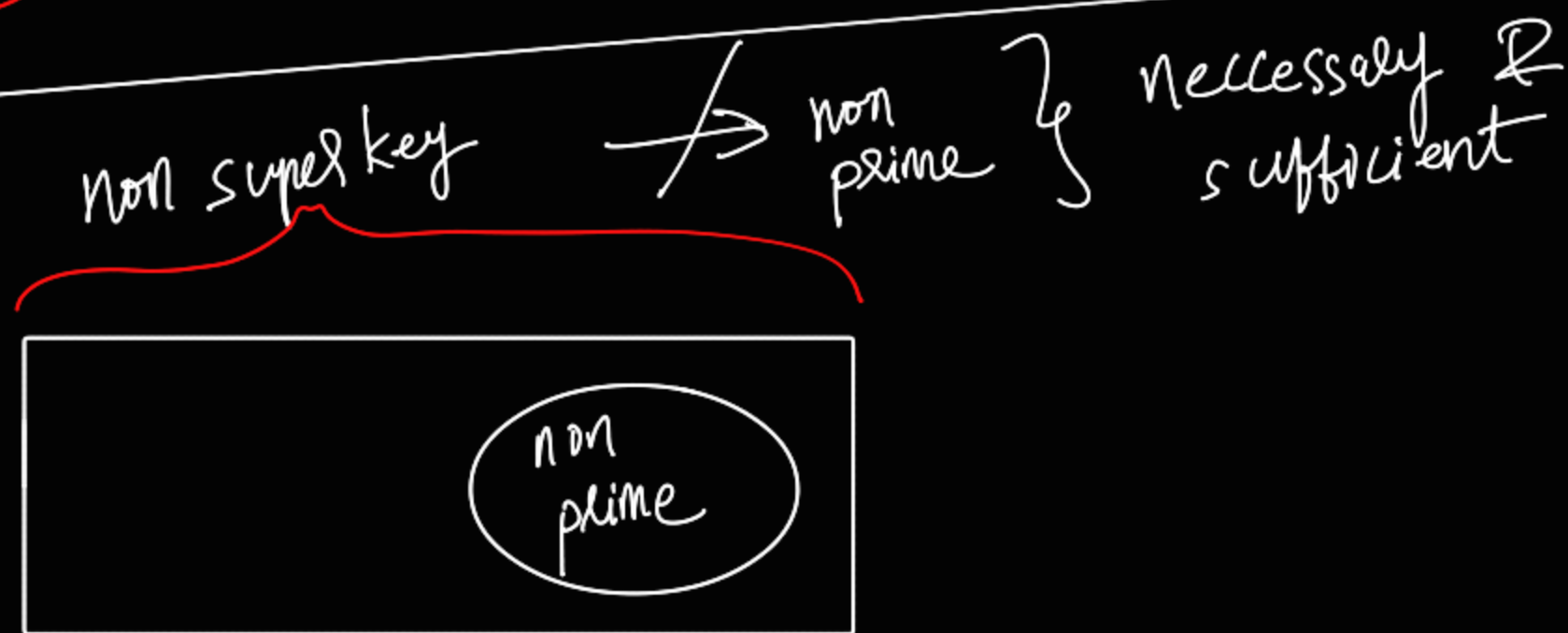
if not 3NF $\Rightarrow \alpha \rightarrow \beta \rightarrow$ non prime for some β and some CK $= \alpha$

then $\beta \not\rightarrow$ any CK } since $\beta \rightarrow$ any CK $\Rightarrow \beta \rightarrow \alpha \Rightarrow$ contradiction } therefore β cannot be a superkey.

$\Rightarrow$ if not 3NF $\Rightarrow$ non superkey $\rightarrow$ non prime
 if no non-superkey $\rightarrow$ non prime $\Rightarrow$ 3NF

There fore Violation 2 is necessary & sufficient.

☆ ϕ if no non prime $\rightarrow$ another non prime then is it 3NF? **False**
 See example above.



all non primes are non superkey
 all non sks are not non primes

ϕ R(ABCD)
 $AB \rightarrow C$
 $ABD \rightarrow C$
 $ABC \rightarrow D$
 $AC \rightarrow D$

1) $AB^+ = \{A, B, C, D\}$ — CK $\rightarrow$ no other CK possible
 $A^+ = A$
 $B^+ = B$ } proper subset of AB $\nrightarrow$ R $\Rightarrow$ 2NF

non prime = C, D
 but $AC \rightarrow D$ } non superkey $\rightarrow$ non prime
 $AB \rightarrow AC \rightarrow D$ } 3NF violation

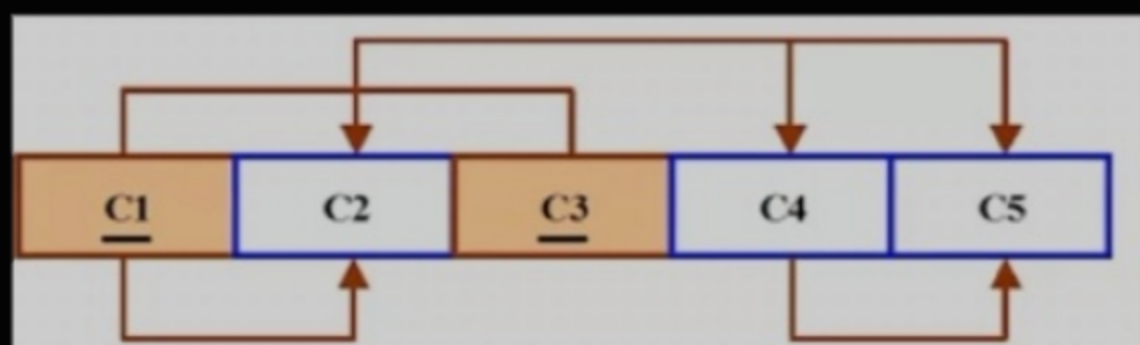
ϕ R(ABCD)
 $ABD \rightarrow C$
 $A \rightarrow B$
 $AB \rightarrow C$
 $B \rightarrow A$

$AD^+ \rightarrow \{A, B, C, D\} = R$
 $BD^+ \rightarrow R$
 $\Rightarrow$ non prime = C
 $A^+ = \{A, B, C\} \equiv$ not 2NF

but $AB^+ = \{A, B, C\}$
 $AB \rightarrow C$ } not sk $\rightarrow$ non prime
 $\Rightarrow AD \rightarrow AB \rightarrow C$ } 3NF violation

ϕ R(ABCD) FD set { $ACD \rightarrow B$ $AC \rightarrow D$ $D \rightarrow C$ $AC \rightarrow B$	$A^+ = A$ $AC^+ = ABCD$ $AB^+ = AB$ $AD^+ = ABCD$ $C^+ = C$ $D^+ = DC$	non prime B only $\Rightarrow$ 2NF	non sks $CD^+ = \{C, D\}$ $CD \nrightarrow B$ $\equiv$ 3NF
--	---	--	---

all other non sks contain B



$R(C_1 C_2 C_3 C_4 C_5)$
 $C_1 \rightarrow C_2$
 $C_1 C_3 \rightarrow C_2 C_4 C_5$
 $C_4 \rightarrow C_5$

$$C_1^+ = \{C_1 C_2\}$$

$$C_1 C_3^+ = \{C_1 C_2 C_3 C_4 C_5\} = R \text{ only CK}$$

$$C_3^+ = \{C_3\}$$

non prime = $\{C_2 C_4 C_5\}$

but $C_1 \rightarrow C_2$

$\Rightarrow$ not 2NF $\Rightarrow$ not 3NF

$C_1 C_3 \rightarrow C_4 \rightarrow C_5$
 CK $\rightarrow$ NSK $\rightarrow$ NP

BCNF

Boyce - codd Normal form

Definition

$\rightarrow$ A relation R is in BCNF in every Non trivial FD in R of the form $X \rightarrow A$, X is a SK of R

★

what makes a relation in 3NF but not in BCNF

BCNF	3NF
if $\alpha \rightarrow X$ in R then α should be superkey where X is any attribute	if $\alpha \rightarrow$ non prime in R then α should be superkey
if $\alpha \rightarrow X$ in R then α should be superkey where X is any attribute	if $\alpha \rightarrow X$ in R then either α should be SK or X should be prime

$\Rightarrow$ Every BCNF is in 3NF

$\Rightarrow$ Every non 3NF is also non BCNF

if R is 3NF & not in BCNF then

a) $\exists$ a non trivial FD $\alpha \rightarrow X$ | $X \rightarrow$ prime attribute ; $\alpha \rightarrow$ not a superkey
 non SK $\rightarrow$ prime

Consider a table/Relation R has one candidate key, then which of the following is always true?

(A) If R is in 2NF, then it is also in 3NF ✗

✓ (B) If R is in 3NF, then it is also in BCNF ★

(C) If R is in 2NF, but it is not in 3NF

(D) None of the above.

Only one CK

3NF $\Rightarrow \alpha \rightarrow X$ • if X is non prime
 α is SK

• if α not SK then X prime

BCNF = $\alpha \rightarrow X$ α is SK

Only one CK $\Rightarrow$ non SK are either $\rightarrow$ subset of CK
 $\rightarrow$ made of non prime
 $\rightarrow$ combination of prime & non prime

if R is in 3NF but not in BCNF

then $\exists$ a nontrivial FD $\alpha \rightarrow X$ $\Rightarrow \exists$ at least two overlapping CKs

non SK $\rightarrow$ prime

How can non SK be made

① $\rightarrow$ Subset of CK

② $\rightarrow$ Combination of primes such that they do not form a key or subset of CK.

$\rightarrow$ Combination of nonprimes $X \rightarrow$ since nonprime $\alpha \rightarrow$ prime X is contradiction of definition

③ $\rightarrow$ Combination of prime & nonprime

$\hookrightarrow$ But we know from below result we need at least two overlapping CKs.

We have $\rightarrow$ $\left\{ \begin{array}{l} \text{subset of } K \rightarrow \text{not SK} \\ \text{subset of } KX \rightarrow \text{CK} \\ KY \rightarrow \text{CK} \end{array} \right\} \Rightarrow$ $X \& Y$ are part of
At least two different overlapping CKs

~~for~~
 $X, Y \rightarrow$ prime attribute
 $B \rightarrow$ non prime
 $K \rightarrow$ set of primes without X, Y

Consider the following relation with functional dependencies as shown below
 $R\{sno, sname, age, cno, cname, group\}$
 $R(A, B, C, D, E, F)$
 $A \rightarrow B, C$
 $C \rightarrow F$
 $D \rightarrow E$
 In which normal form is the relation R in?
 (A) First normal form
 (B) Second normal form
 (C) Third normal form
 (D) Boyce Codd normal form

→ AD should be part of every CK
 → $AD^+ = \{A, B, C, D, E, F\} \rightarrow$ only CK
 $A^+ = \{A, B, C, F\}$
 $D^+ = \{D, E\}$
 → nonprime = $\{B, C, E, F\}$
 → $A \rightarrow B \Rightarrow$ not 2NF
 since $AD \rightarrow A \rightarrow B$ CK partial transitive → non prime

GATE IT 2005 | Question: 22

asked Nov 3, 2014 • edited Oct 4, 2020 by Gatecse

11,980 views

A table has fields F_1, F_2, F_3, F_4, F_5 with the following functional dependencies

51

• $F_1 \rightarrow F_3, F_2 \rightarrow F_4, (F_1 \cdot F_2) \rightarrow F_5$

In terms of Normalization, this table is in

- ✓ A. 1 NF
- B. 2 NF
- C. 3 NF
- D. None of these

→ $F_1, F_2 \rightarrow$ part of every CK
 → $\{F_1, F_2\}^+ = \{F_1, F_2, F_3, F_4, F_5\} \rightarrow$ only CK
 → $F_3, F_4, F_5 \rightarrow$ non prime
 $F_1^+ = \{F_1, F_3\} \Rightarrow$ subset of CK → non prime.
 ⇒ 1NF form only

GATE CSE 1987 | Question: 2n

asked Nov 9, 2016 • recategorized Apr 22, 2021 by Lakshman Bhaiya

6,681 views

State whether the following statements are TRUE or FALSE:

36 A relation r with schema (X, Y) satisfies the function dependency $X \rightarrow Y$. The tuples $\langle 1, 2 \rangle$ and $\langle 2, 2 \rangle$ can both be in r simultaneously. True

$X^+ = \{X, Y\}$
 $X = CK$
 here Y repeats, but that is OK.

GATE CSE 2020 | Question: 36

asked Feb 12, 2020 • retagged Nov 30, 2022 by Lakshman Bhaiya

12,585 views

Consider a relational table R that is in 3NF, but not in BCNF. Which one of the following statements is TRUE?

55

- A. R has a nontrivial functional dependency $X \rightarrow A$, where X is not a superkey and A is a prime attribute. ✓
- B. R has a nontrivial functional dependency $X \rightarrow A$, where X is not a superkey and A is a non-prime attribute and X is not a proper subset of any key.
- C. R has a nontrivial functional dependency $X \rightarrow A$, where X is not a superkey and A is a non-prime attribute and X is a proper subset of some key
- D. A cell in R holds a set instead of an atomic value.

GATE CSE 2007 | Question: 62, UGCNET-June2014-II: 47

asked Sep 21, 2014 • edited Jun 9, 2021 by SoujanyaReddy13

34,176 views

Which one of the following statements is FALSE?

98

- A. Any relation with two attributes is in BCNF ✓
- B. A relation in which every key has only one attribute is in 2NF ✓
- C. A prime attribute can be transitively dependent on a key in a 3 NF relation ✓
- D. A prime attribute can be transitively dependent on a key in a BCNF relation ✗

D false
 $CK \rightarrow A \rightarrow$ prime attribute
 $\Rightarrow A \rightarrow$ not a SK
 $\Rightarrow$ not SK → prime attribute
 $\Rightarrow$ BCNF violation

GATE CSE 1990 | Question: 3-ii

asked Nov 19, 2016 • edited Apr 17, 2021 by Lakshman Bhaiya

22,449 views

Indicate which of the following statements are true:

85

A relational database which is in 3NF may still have undesirable data redundancy because there may exist:

- A. Transitive functional dependencies ✓
- B. Non-trivial functional dependencies involving prime attributes on the right-side. ✓
- C. Non-trivial functional dependencies involving prime attributes only on the left-side. ✗
- D. Non-trivial functional dependencies involving only prime attributes. ✓

A relation is in 3NF if it is in 2NF and no non-prime attribute should be determined by anything but a super key (no transitive dependency). This means 3NF still allows FDs like $A \rightarrow B$, where A is not a super key as long as B is a prime attribute. That is, a proper subset of some candidate key (which means some combination of prime attributes) determining some other (if its same it becomes trivial) prime attribute(s).

Reference: <https://gatecse.in/demystifying-database-normalization/>

Now lets see the options

- A. Transitive functional dependencies
Is possible in 3NF as long as RHS is a prime attribute.
- B. Non-trivial functional dependencies involving prime attributes on the right-side
Possible as explained above. But any attribute(s) determining a prime attribute automatically becomes a prime(s) attribute implying the left-side here must also be prime-attribute(s). ✗ Reasoning not true as explained below.
- C. Non-trivial functional dependencies involving prime attributes only on the left-side
Not possible in 3NF
- D. Non-trivial functional dependencies involving only prime attributes
This is possible and infact same as option B as explained for option B.

GATE CSE 2012 | Question: 2



asked Aug 5, 2014 • edited Jun 21, 2021 by Lakshman Bhaiya

26,187 views



Which of the following is TRUE?

72



- A. Every relation in 3NF is also in BCNF ✗
- B. A relation R is in 3NF if every non-prime attribute of R is fully functionally dependent on every key of R ✗
- C. Every relation in BCNF is also in 3NF ✓
- D. No relation can be in both BCNF and 3NF ✗

Let $K \rightarrow$ = candidate key X be a prime attribute
(K = set of primes without X) B be a non prime

Proof of why $\alpha X \rightarrow Y$
 $X \rightarrow$ non prime $\alpha \rightarrow$ set of prime holds
 $Y \rightarrow$ prime

then $XB \rightarrow Y$ possible?
if yes then, $KXB \rightarrow$ superkey $\neq$ not CK $\Rightarrow$ Subset is CK
 $KB \neq XB \rightarrow$ not CK since $B \rightarrow$ non prime
Subset of $K \rightarrow$ not CK since $KY \rightarrow$ CK
Subset of $KX \rightarrow$ must be candidate key.

We have $\Rightarrow$ $\left. \begin{array}{l} \text{subset of } K \rightarrow \text{not SK} \\ \text{subset of } KX \rightarrow \text{CK} \\ KY \rightarrow \text{CK} \end{array} \right\} \Rightarrow$ $X \neq Y$ all part of
Atleast two different overlapping CKs

$X, Y \rightarrow$ prime attribute
 $B \rightarrow$ non prime
 $K \rightarrow$ set of primes without X, Y

Example. $R(ABCD)$ prime $\rightarrow (ABC)$; $D \rightarrow$ non prime
 $AD \rightarrow B$ minimally keys CA, CB $CA \rightarrow D$
 $CB \rightarrow A$

$R(ABCD)$ $AD \rightarrow B$ $\rightarrow C$ part of all CK.
 $CA^+ \rightarrow \{A, B, C, D\}$ } only
 $CB^+ \rightarrow \{A, B, C, D\}$ CKs
 $CD^+ \rightarrow \{C, D\}$
 $AD^+ \rightarrow \{A, D, B\}$
 $BD^+ \rightarrow \{B, D\}$
 $AB \rightarrow \{A, B\}$

$\Rightarrow A, B, C \rightarrow$ prime
 $D \rightarrow$ non prime
 $AD \rightarrow B$
 $\left. \begin{array}{l} \text{prime} \\ \text{non} \\ \text{prime} \end{array} \right\} \rightarrow B$ prime

3NF check all non SK without D
 $A, B, C, AB \rightarrow$ non gives D $\equiv$ 3NF.

$C \rightarrow C$
 $A \rightarrow A$
 $B \rightarrow B$ } No proper subset gives non prime

$\equiv$ 2NF

Natural Join

- ① Find Common attribute
- ② Equate values, see and enumerate equal values for the common attribute
- ③ Common attribute is not repeated

$R_1 \bowtie R_2$

Notation

- Not all tuples may be represented
- Addition spurious information may appear

eg. ☆

S	Name	SSN	DOB	Grade	T	Grade	Salary
A	121	2367	2	2	80		
A	132	3678	3	3	70		
B	101	3498	4	4	70		
C	106	2987	2				

R	Name	SSN	DOB	Grade	Salary
A	121	2367	2	80	
A	132	3678	3	70	
B	101	3498	4	70	
C	106	2987	2	80	

$R = S \bowtie T$

R	A	B	S	B	C
X	Y	Z	U		
X	Z	V	W		
Y	Z	Z	V		
Z	V	Z	V		

$R \bowtie S$

A	B	C
X	Z	U
X	Z	V
Y	Z	U
Y	Z	V
Z	V	W

- If Now common attribute → Combine every row of R with every row of S } Cartesian or Cross product

Anomalies — Undesired behaviour in DB

Root cause of anomaly is redundancy of information

- FDS are decided by DB owner
- FDS are designed in the conceptual model & not in the logical model
- FDS are part of database & not relation or schema
- FDS apply on entire database

→ Redundancy is created by FDS

- ① Update Anomaly
- ② Insertion Anomaly
- ③ Deletion Anomaly

College, department, building Example

Solution → Decomposing

15.1.1 Problems Caused by Redundancy

Storing the same information **redundantly**, that is, in more than one place within a database, can lead to several problems:

- **Redundant storage:** Some information is stored repeatedly.
- **Update anomalies:** If one copy of such repeated data is updated, an inconsistency is created unless all copies are similarly updated.
- **Insertion anomalies:** It may not be possible to store some information unless some other information is stored as well.
- **Deletion anomalies:** It may not be possible to delete some information without losing some other information as well.

The essential idea is that many problems arising from redundancy can be addressed by replacing a relation with a collection of 'smaller' relations. Each of the smaller relations contains a (strict) subset of the attributes of the original relation. We refer to this process as *decomposition* of the larger relation into the smaller relations.

We can deal with the redundancy in Hourly_Emps by decomposing it into two relations:

Hourly_Emps2(ssn, name, lot, rating, hours_worked)
Wages(rating, hourly_wages)

Decomposition

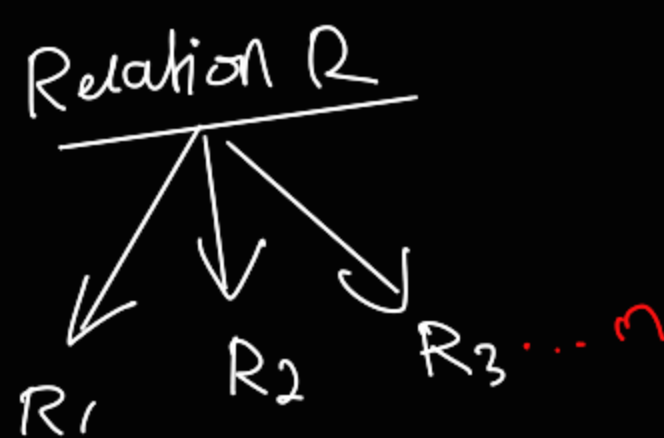
Two properties of decompositions are of particular interest.

The **lossless-join property** enables us to recover any instance of the decomposed relation from corresponding instances of the smaller relations.

The **dependency preservation property** enables us to enforce any constraint on the original relation by simply enforcing some constraint on each of the smaller relations. That is, we need not perform joins of the smaller relations to check whether a constraint on the original relation is violated.

- ① Attributes should be preserved $\bigcup_{i=1}^n R_i = R$
- ② No new attributes. $R_i \subset R$

equivalent



- Each new relation scheme contains a subset of the attributes of R, and
- Every attribute of R appears as an attribute of at least one of the new relations.

→ Attribute preservation property.

Definition
The decomposition of a schema $R = A_1 \dots A_n$ is its replacement by a collection $D_R = \{R_1, R_2, \dots, R_m\}$ of subsets of R such that $R = R_1 \cup R_2 \cup \dots \cup R_m$

Note: schemas R_i 's do not have to be disjoint!

Lossless decomposition → will never give wrong information
Lossy decomposition → may give wrong information for some instances.

- Not losing any original tuple
- But get spurious tuples which were not in original info
- Getting wrong info $\equiv$ loss of info

$$R \xrightarrow{\text{decompose}} R_1, R_2, \dots, R_n \Rightarrow \text{Tuples}(R_1 \bowtie R_2 \bowtie \dots \bowtie R_n) \supset \text{Tuples}(R)$$

$$R \xrightarrow{\text{decompose}} R_1, R_2, \dots, R_n \Rightarrow \text{Tuples}(R_1 \bowtie R_2 \bowtie \dots \bowtie R_n) = \text{Tuples}(R)$$

Binary decomposition → To check lossy or not → common attribute(s) must be a SK in atleast R_1 or $R_2 \Rightarrow$ Lossless Decomposition

→ Decomposition D is Lossless if $R_1 \cap R_2$ is a superkey in R_1 or R_2 , however it may or maynot be SK in R

ie. $R_1 \cap R_2 \rightarrow R_1$
 $R_1 \cap R_2 \rightarrow R_2$ } is any of the FD hold then Lossless Binary decomposition

Lossless-Join Decomposition also called Non-additive join decomposition

Dependency preservation in Decomposition

→ Database → Set of F^h FDs = F
 Universal relation (R) → decomposed to $R_1, R_2, R_3, \dots$

$\begin{matrix} (R_1) & (R_2) & (R_3) \\ F_1 \text{ holds} & F_2 \text{ holds} & F_3 \text{ holds} \end{matrix} \quad \left. \vphantom{\begin{matrix} (R_1) \\ (R_2) \\ (R_3) \end{matrix}} \right\} F_i \rightarrow \text{contains those } F^h \text{ FDs which use attributes of } R_i$

→ given a set of F^h FDs → Take or convert to set of all possible FDs in which

- ① FDs are non trivial
- ② FDs have RHS only 1 attribute

Then all other F^h FDs can be inferred from the above set.
 So we can work on these only for simplicity & avoiding trivial

eg:- Universal R(ABCE)
 $R_1(ABC)$
 $R_2(ADE)$

F^h set $F = \{ A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A \}$

Common Attribute (A)
 $A^+ = \{ A, B, C, D, E \} = R$
 so lossless

$F_1 = F^h$ FDs holding in R_1

$F_1 = \{ A \rightarrow BC, BC \rightarrow A \}$
 ↪ don't miss this

$F_2 = \{ A \rightarrow DE, E \rightarrow AD \}$

$E^+ = R$
 $D^+ = D$

In above question is $F_1 \cup F_2 \Rightarrow F$?

$F_1 \cup F_2 = \{ A \rightarrow BC, BC \rightarrow A, A \rightarrow DE, E \rightarrow AD \}$

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

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

$B^+ = \{ B \}$

$C^+ = \{ C \}$

so $CD \rightarrow E \nRightarrow B \rightarrow D$
 is not preserved

→ $F_1 \cup F_2$ where F_i is the F^h FDs preserved by R_i
 (ie F_i is the set of attributes arising on attributes of R)

↪ Does not preserve F

Decomposition → Lossless ↗ Not dependency preserving

Did we really lose this dependency? → NO, since $R_1 \bowtie R_2 = R$

and R has all F^h FDs

→ Without join → FD's cannot be enforced → so we say Non dependency preserving.

Entire Database in one Table

→ More redundancy, null values, anomaly,

→ More memory.

→ No need to join to retrieve data.

Decomposing DB into lossless Relations

→ Less redundancy

→ Less memory

→ Less Anomaly

→ To get information, we may have to join table

→ if we can $(F_1 \cup F_2 \cup F_3 \dots F_L)$ combinely cover F , then we don't have to join table to enforce dependency.

→ This situation is called dependency preservation.

→ If we need to join tables to enforce dependency, then non dependency preserving.

eg: Non dependency preserving

A	B	C
1	Smith	Sorian
2	John	Lena
3	Smith	Sorian
4	Smith	Sorian
5	John	Lena

R_1

R_2

A	B
1	Smith
2	John
3	Smith
4	Smith
5	John

A	C
1	Sorian
2	Lena
3	Sorian
4	Sorian
5	Lena

→ Here, if I

want to add new person say "John Mon" then I

can do it in R_1 & R_2

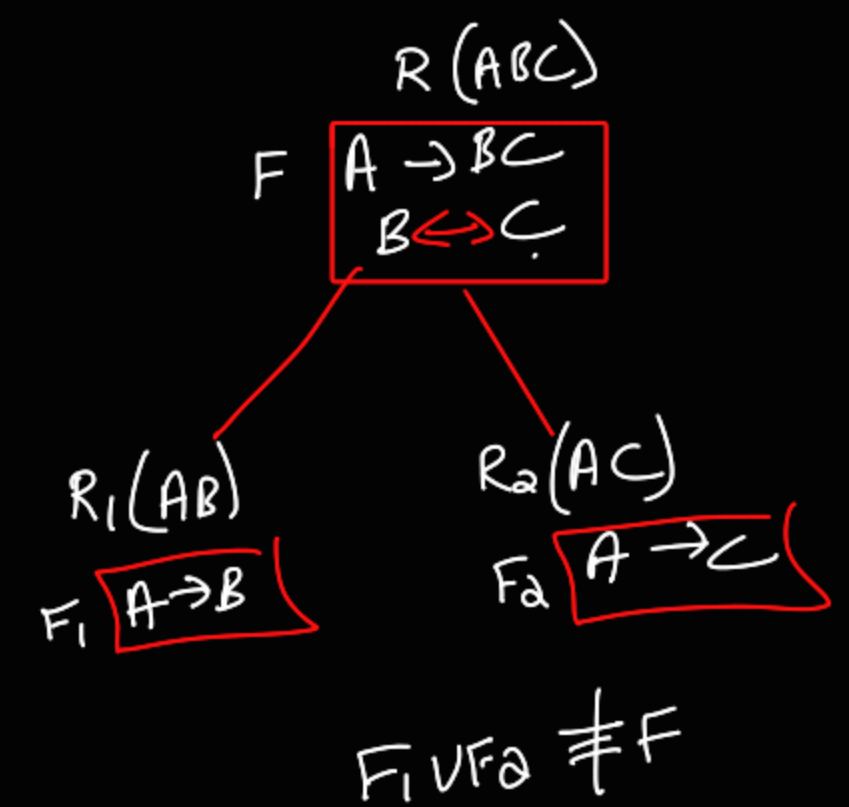
without violating $F_1 \cup F_2$

by adding {1, John} in R_1

& {1, Mon} in R_2 . But

this violates the dependency $B \rightarrow C$;

→ This means that when entering new elements / updating / deleting to enforce dependency, I need to join relation to check.



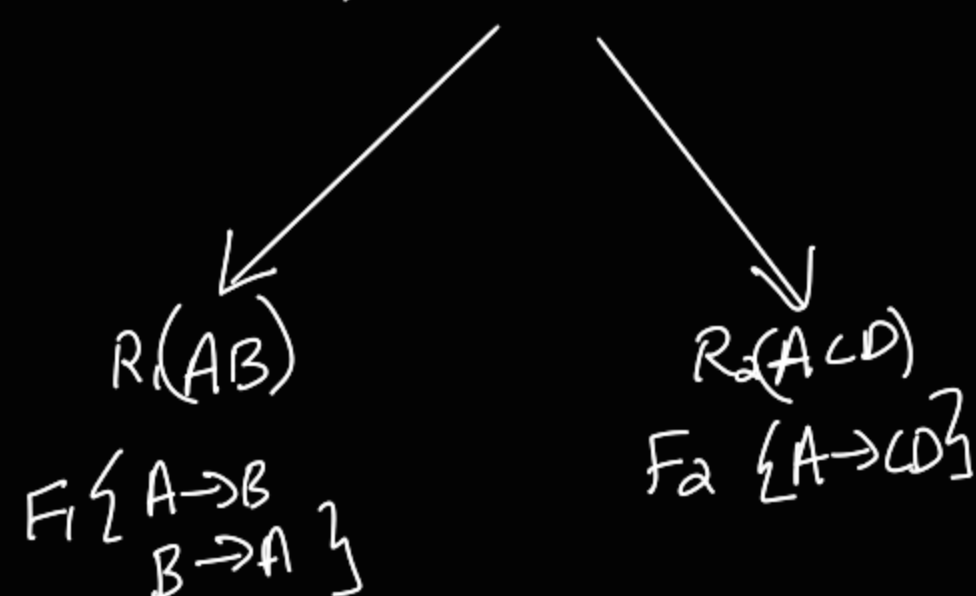
eg:- Dependency preserving.

SN	s_id	Fname	Sname
1	7892	John	Lena
2	6291	Smith	Sorian
3	1123	John	Lena
4	3790	Smith	Sorian
5	2406	Smith	Turtios

SN	s_id
1	7892
2	6291
3	1123
4	3790
5	2406

SN	Fname	Sname
1	John	Lena
2	Smith	Sorian
3	John	Lena
4	Smith	Sorian
5	Smith	Turtios

$R(ABCD)$
 $F \{ A \rightarrow B, B \rightarrow A, B \rightarrow CD \}$



$$F_1 \cup F_2 = F$$

→ Here if I want to add new person say "John Mon". then I can just enforce / satisfy F_1 & F_2 without violating F . I don't need to check by joining the relation

→ In decomposition most important is lossless & dependency preserving is desirable

→ F covers $F_1 \cup F_2 \cup F_3 \dots$ } for any decomposition
 $\{F_1 \cup F_2 \cup F_3 \dots\}^+ \subseteq F^+$

→ $F^+ = F_1 \cup F_2 \dots \rightarrow$ preserving
 $F' \subset F_1 \cup F_2 \dots \rightarrow$ non preserving.

eg:- $R(ABC)$ $F: \{A \rightarrow B, B \rightarrow C, A \rightarrow C\}$ | $F_1: \{A \rightarrow B\}$ $F_2: \{B \rightarrow C\}$ $\{F_1 \cup F_2\} = F$
 $R_1(AB)$ $R_2(BC)$ $\therefore$ Dependency preserving.

eg:- $R(ABC)$ $F: \{AB \rightarrow C, C \rightarrow B\}$ | $R_1(AB)$ $F_1: \{ \}$ $R_2(BC)$ $F_2: \{C \rightarrow B\}$ $F_1 \cup F_2 \neq F$ NOP
 $R_1 \cap R_2^+ = B^+ = \{B\}$ lossy.

eg. $R(ABC)$ $F: \{AB \rightarrow C, C \rightarrow B\}$ | $R_1(AC)$ $F_1: \{ \}$ $R_2(BC)$ $F_2: \{C \rightarrow B\}$ $F_1 \cup F_2 \neq F$ NOP
 $\{R_1 \cap R_2\}^+ = C^+ = \{C, B\} \Rightarrow$ lossless

eg: $R(ABCD)$ $F: \{A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow A\}$
 $R_1(AB)$ $F_1: \{A \rightarrow B, B \rightarrow A\}$
 $R_2(BC)$ $F_2: \{B \rightarrow C, C \rightarrow B\}$
 $R_3(CD)$ $F_3: \{C \rightarrow D, D \rightarrow C\}$

$F_1 \cup F_2 \cup F_3 = F \Rightarrow$ dependency preserving.

How much decomposition to do?

→ Decided by normal forms

1NF
2NF
3NF
BCNF

Redundancy do to
FNOs decrease

BCNF

0% redundancy due to FO

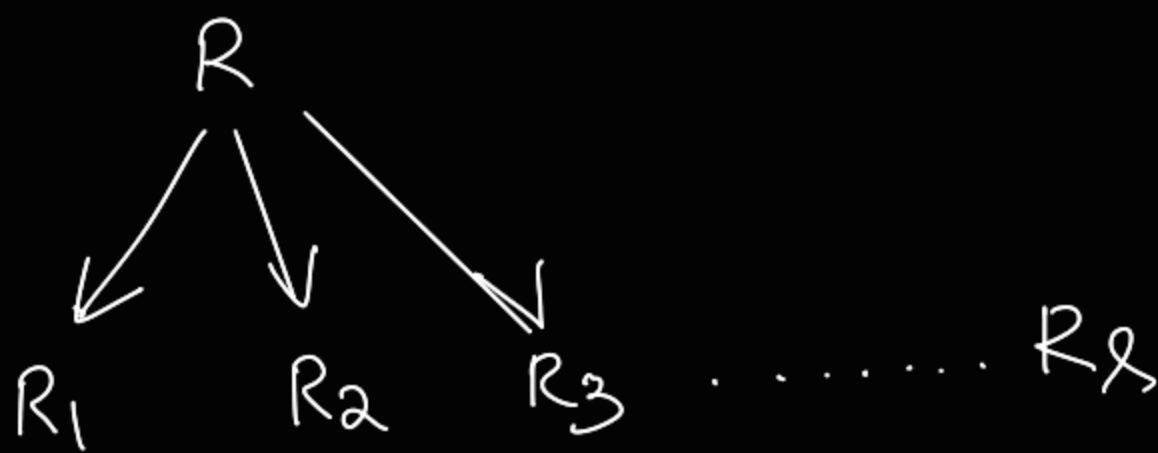
but there can be redundancy due to other factors.

Normalisation

→ Decompose until each relation is in a particular normal form.

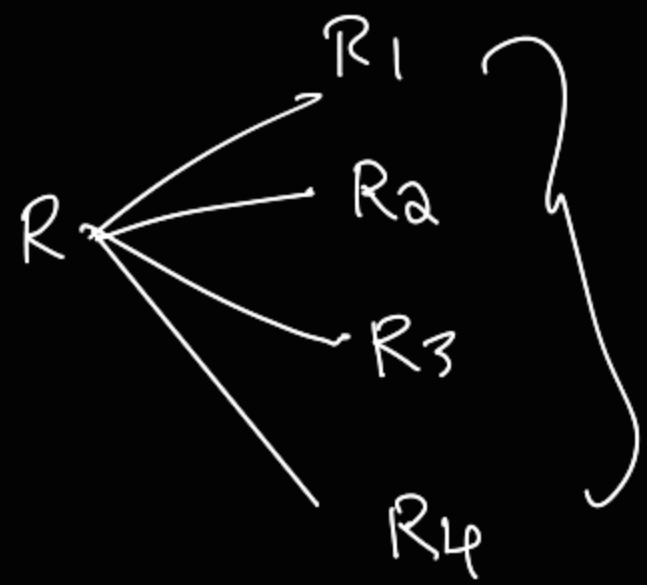
How to check lossy or lossless for Non Binary decomposition?

The chase test



why successive combining method doesn't work

if this method answers $\begin{cases} \text{lossless} \longrightarrow \text{lossless} \\ \text{Lossy} \longrightarrow \text{may/maynot be lossy} \end{cases}$



if there is a way to combine them taken 2 at a time successively in a lossless manner then decomposition is lossless.
But if there is no way to do this $\rightarrow$ we cannot determine anything.

eg:- $R(ABCD)$
 $F \{ A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow A \}$

Case 1
 $R_1(AB) \rightarrow R_1 \cap R_2 = B \rightarrow R_1, R_2 \}$ lossless
 $R_2(BC) \rightarrow R_2 \cap R_3 = C \rightarrow R_2, R_3 \}$ lossless
 $R_3(CD) \rightarrow R_3 \cap R_4 = D \rightarrow R_3, R_4 \}$ lossless
 Since there is a way to combine $\equiv R_1 R_2 R_3$ is a lossless decomposition.

eg $\rightarrow R(ABCDEH)$
 $F \{ A \rightarrow BC, E \rightarrow C, D \rightarrow EH, AH \rightarrow D \}$

$R_1(ABC)$
 $R_2(EC)$
 $R_3(AHD)$
 $R_4(DEH)$

$A^+ = \{A, B, C\}$ - SK in R_1

$B^+ = \{B\}$

$C^+ = \{C\}$

$D^+ = \{EDHC\} \rightarrow$ SK in R_2, R_4

$E^+ = \{E, C\} \rightarrow$ SK in R_2

$(AH)^+ = \{A, B, C, D, E, H\} \rightarrow$ SK in R_1, R_2, R_3, R_4

$R_{13} \rightarrow \text{lossless} = R_{13}(ABCHD)$

$R_{24} \rightarrow \text{lossless} = R_{24}(CDEH)$

$CDH^+ = \{C, D, E, H\} \rightarrow$ SK in R_{24}

$\Rightarrow R_{13} \cap R_{24} \rightarrow R_{24}$

$R_{1234} \rightarrow$ is lossless, so there is a way

to combine all subrelation

taking 2 at a time successively $\Rightarrow$ lossless

eg:- $R(ABCDE)$

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

$R_1(AD)$

$R_2(AB)$

$R_3(BE)$

$R_4(CDE)$

$R_5(AE)$

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

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

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

$D^+ = \{D\}$

$E^+ = \{E\}$

$R_{14} \rightarrow \text{lossy}$
 $R_{24} \rightarrow \text{lossy}$
 $R_{34} \rightarrow \text{lossy}$
 $R_{45} \rightarrow \text{lossy}$

$R_{31} \rightarrow \text{lossy}$
 $R_{32} \rightarrow \text{lossy}$
 $R_{35} \rightarrow \text{lossy}$

$R_{25} \rightarrow \text{lossy}$
 $R_{15} - \text{possible}$
 $R_1, R_2 - \text{possible}$

$\rightarrow R_1 \cap R_2 = A$ skin $R_1 \} R_{12} = (ABD)$
 $A^+ = \{A, C, D\}$

now $R_{123} \rightarrow \text{lossy}$
 $R_{124} \rightarrow \text{lossy}$
 $R_{125} \rightarrow \text{lossy}$

$\rightarrow R_1 \cap R_5 = A$ skin $R_1 \} R_{15} = (ADE)$

now $R_{152} \rightarrow \text{lossy}$
 $R_{153} \rightarrow \text{lossy}$
 $R_{154} \rightarrow DE^+ = \{D, E, C, A\}$
 $\rightarrow \text{possible}$

$\rightarrow R_{154} = ACDE$
 $R_2 = AB$
 $R_3 = BE$

now $R_{1542} \rightarrow \text{lossy}$
 $R_{1543} \rightarrow \text{lossy}$

$\rightarrow$ We have exhausted all possibilities of combining 2 subrelations at a time successively in a lossless manner.

$\Rightarrow$ We can't comment about the actual decomposition

CHASE TEST / Algorithm

$R(ABCE) \xrightarrow[\text{compose}]{\text{decompose}} R_1 R_2 R_3 R_4 \} \text{non binary decomposition}$

Create matrix

$R_1(AB)$

$R_2(AC)$

$R_3(BCD)$

$R_4(ADE)$

	a	b	c	d	e
R_1	a	a	a	a	b
R_2	a		a		
R_3		a	a	a	b
R_4	a		a	a	a

$\rightarrow$ columns are attributes

Assume FD

1) $A \rightarrow C$

2) $BC \rightarrow E$

3) $E \rightarrow D$

Rows are relations

eg. $R(ABCD)$
 $R_1 \rightarrow ABC$
 $R_2 \rightarrow CD$

FD set $F \rightarrow \{A \rightarrow B, B \rightarrow C, \underline{C \rightarrow D}\}$

	A	B	C	D
$R_1(ABC)$	a	a	a	a
$R_2(CD)$			a	a

→ Row of all a's
 $\Rightarrow$ lossless

- ①
- ②
- ③
- ④

⑤ Row of all entries $\Rightarrow$ Lossless ; Else Lossy

eg. ⑥ $R(ABCD)$
 $F(A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow A)$

	A	B	C	D
$R_1(AB)$	a	a	a	a
$R_2(BC)$		a	a	a
$R_3(CD)$			a	a

$\exists$ row of a entries $\Rightarrow$ Lossless

eg. ⑦ $R(SAIP)$
 $S \rightarrow A$
 $SI \rightarrow P$
 $R_1(SA)$
 $R_2(SIP)$

	S	A	I	P
$R_1(SA)$	a	a		
$R_2(SIP)$	a	a	a	a

→ Row of all a's

eg. $A \rightarrow BCDE$
 $ABC \quad CDE$

	A	B	C	D	E
ABC	a	a	a		
CDE			a	a	a

Intuition
 What this method is checking.

it is checking, if the common element between relations can determine the attributes not in relation

Common element repeating
 $\Rightarrow$ only if a particular attribute repeats between relation, can we meaningfully apply the Fⁿs to fill the rest of table.

Stopping condⁿ $\rightarrow$ if a relation is able to infer all other attributes, not in itself uniquely, then, we say lossless

Examples & PQDs

1. $R(ABCDE)$

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

- $R_1(AD)$
- $R_2(AB)$
- $R_3(BE)$
- $R_4(CDE)$
- $R_5(AE)$

Chase test

	A	B	C	D	E
$R_1(AD)$	a		b/a	a	
$R_2(AB)$	a	a	b/a	a	
$R_3(BE)$	a	a	b/a	a	a
$R_4(CDE)$	a		a	a	a
$R_5(AE)$	a		b/a	a	a

all row entries

here $A \rightarrow C, C \rightarrow D$
 $\Rightarrow A \rightarrow D$
but instead of using this directly chase algorithm assumes value for intermediate C first & then finds D.

$\rightarrow$ A new variable b $\rightarrow$ is the assumption that we will get this attribute in the future and we can hence use this attribute to determine others.
this about this is that this attribute is the intermediate step in some transitive dependency.

Another way

Let $R(A, B, C, D)$ be a relational schema with the following functional dependencies: $A \rightarrow B, B \rightarrow C, C \rightarrow D$ and $D \rightarrow B$. The decomposition of R into $(A, B), (B, C), (B, D)$

- A. gives a lossless join, and is dependency preserving
- B. gives a lossless join, but is not dependency preserving
- C. does not give a lossless join, but is dependency preserving
- D. does not give a lossless join and is not dependency preserving

	A	B	C	D
$R(AB)$	a	a	a	a
$R(BC)$		a	a	a
$R(BD)$		a	a	a

row of a's $\Rightarrow$ Lossless decomposition

$R(AB) \Rightarrow F_1 = \{A \rightarrow B\}$

$R_2(BC) \Rightarrow F_2 = \{B \rightarrow C, C \rightarrow B\}$

$R_3(BD) \Rightarrow F_3 = \{D \rightarrow B, B \rightarrow D\}$

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

$F_1 \cup F_2 \cup F_3 \Rightarrow F \Rightarrow$ dependency preserving

Example 3.24: Consider the relation $R(A, B, C, D)$ with the FD $B \rightarrow AD$ and the proposed decomposition $\{A, B\}, \{B, C\}$, and $\{C, D\}$. Here is the initial tableau:

	A	B	C	D
$R(AB)$	a	a		b
$R(BC)$	a	a	a	b
$R(CD)$			a	a

$\rightarrow$ Further no progress possible
 $\rightarrow$ No row of all 'a' entries
 $\Rightarrow$ Lossy decomposition

Example 3.22: Suppose we have relation $R(A, B, C, D)$, which we have decomposed into relations with sets of attributes $S_1 = \{A, D\}$, $S_2 = \{A, C\}$, and $S_3 = \{B, C, D\}$.

suppose the given FD's are $A \rightarrow B, B \rightarrow C$, and $CD \rightarrow A$.

	A	B	C	D
$R_1(AD)$	a	b	a	a
$R_2(AC)$	a	b	a	
$R_3(BCD)$	a	a	a	a

row of all 'a' entries

$\Rightarrow$ Lossless decomp

Exercise 3.4.1: Let $R(A, B, C, D, E)$ be decomposed into relations with the following three sets of attributes: $\{A, B, C\}$, $\{B, C, D\}$, and $\{A, C, E\}$. For each of the following sets of FD's, use the chase test to tell whether the decomposition of R is lossless. For those that are not lossless, give an example of an instance of R that returns more than R when projected onto the decomposed relations and rejoined.

- $B \rightarrow E$ and $CE \rightarrow A$
- $AC \rightarrow E$ and $BC \rightarrow D$
- $A \rightarrow D$, $D \rightarrow E$, and $B \rightarrow D$
- $A \rightarrow D$, $CD \rightarrow E$, and $E \rightarrow D$

Exercise 3.4.2: For each of the sets of FD's in Exercise 3.4.1, are dependencies preserved by the decomposition?

b)

	A	B	C	D	E
$R_1(ABC)$	a	a	a	a	a
$R_2(BCD)$		a	a	a	
$R_3(ACE)$	a		a		a

row of all 'a' entry

$\Rightarrow$ Lossless

$$F_1 = \{ \}$$

$$F_2 = \{ BC \rightarrow D \}$$

$$F_3 = \{ AC \rightarrow E \}$$

$\bigcup F_i = F \Rightarrow$ Dependency preserving.

c)

	A	B	C	D	E
$R_1(ABC)$	a	a	a	a	a
$R_2(BCD)$		a	a	a	
$R_3(ACE)$	a		a	a	a

$\rightarrow A \rightarrow E$

$\rightarrow$ row of all 'a' entry $\Rightarrow$ Lossless

$\downarrow$ intermediary for $A \Rightarrow D \rightarrow E$

$$F_1 = \{ \}$$

$$F_2 = \{ B \rightarrow D \}$$

$$F_3 = \{ A \rightarrow E \}$$

$\Rightarrow$ Non Dependency preserving.

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

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

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

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

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

d)

	A	B	C	D	E
$R_1(ABC)$	a	a	a	b	a
$R_2(BCD)$		a	a	a	
$R_3(ACE)$	a		a	b	a

$\rightarrow$ No progress possible

$\rightarrow$ No row of all 'a' entry

$\Rightarrow$ Lossy.

intermediary

$$F_1 = \{ \}$$

$$F_2 = \{ \}$$

$$F_3 = \{ AC \rightarrow E \}$$

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

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

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

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

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

Consider a schema $R(A, B, C, D)$ and functional dependencies $A \rightarrow B$ and $C \rightarrow D$. Then the decomposition of R into $R_1(A, B)$ and $R_2(C, D)$ is

- dependency preserving and lossless join
- lossless join but not dependency preserving
- dependency preserving but not lossless join
- not dependency preserving and not lossless join

	A	B	C	D
$R_1(AB)$	a	a		
$R_2(CD)$			a	a

$\rightarrow$ Observation $\rightarrow$ Even in non binary decomposition, if any one of the subrelation has no common attribute with any of the other subrelations then the decomposition is Lossy.

intuition $\rightarrow$ There is no way for R_1, R_2, R_3 to infer attribute E, F also R_4 cannot infer A, B, C, D $\Rightarrow$ Lossy

	A	B	C	D	E	F
R_1	a	a				
R_2			a	a		
R_3	a			a		
R_4					a	a

Q) $R(LMNP)$
 $F = \{M \rightarrow O, NO \rightarrow P, P \rightarrow L, L \rightarrow MN\}$
 $R_1(LMNP)$
 $R_2(MO)$

$R_1 R_2 = M \rightarrow O$
 $R_1 R_2 \rightarrow R_2 \Rightarrow$ Lossless
 is dependency preserving?

$F_1 = \{MN \rightarrow P, P \rightarrow MNL, L \rightarrow MNP\}$
 $F_2 = \{M \rightarrow O\}$

$F_1 \cup F_2 \neq F$
 $\Rightarrow F_1 \cup F_2 \neq F$
 $\rightarrow$ Non dependency preserving

Q) What is highest normal form? in above Question

$\rightarrow R_2$ is in BCNF

$R_1(LMNP)$
 $F_1 = \{MN \rightarrow P, P \rightarrow MNL, L \rightarrow MNP\} \rightarrow$ set of all non trivial
 $\Rightarrow$ All LHS is SK so BCNF

Q)

Consider the relation $R(P, Q, S, T, X, Y, Z, W)$ with the following functional dependencies.

$PQ \rightarrow X; P \rightarrow YX; Q \rightarrow Y; Y \rightarrow ZW$

Consider the decomposition of the relation R into the constituent relations according to the following two decomposition schemes.

- $D_1: R = \{R_1(P, Q, S, T); R_2(P, T, X); R_3(Q, Y); R_4(Y, Z, W)\}$
- $D_2: R = \{R_1(P, Q, S); R_2(T, X); R_3(Q, Y); R_4(Y, Z, W)\}$

Which one of the following options is correct?

☒ A. D_1 is a lossless decomposition, but D_2 is a lossy decomposition
☐ B. D_1 is a lossy decomposition, but D_2 is a lossless decomposition
☐ C. Both D_1 and D_2 are lossless decompositions
☐ D. Both D_1 and D_2 are lossy decompositions

option (A)

D_1

	P	Q	S	T	X	Y	Z	W
$R_1(P, Q, S, T)$	a	a	a	a	a	a	a	a
$R_2(P, T, X)$	a			a	a	a	a	a
$R_3(Q, Y)$		a				a	a	a
$R_4(Y, Z, W)$						a	a	a

$\Rightarrow$ Lossless.

D_2

	P	Q	S	T	X	Y	Z	W
$R_1(P, Q, S)$	a	a	a					
$R_2(T, X)$				a	a			
$R_3(Q, Y)$		a				a		
$R_4(Y, Z, W)$						a	a	a

R_2 has no common attribute with any other relation
 $\Rightarrow$ Lossy.

Q)

Let the set of functional dependencies $F = \{QR \rightarrow S, R \rightarrow P, S \rightarrow Q\}$ hold on a relation schema $X = (PQRS)$. X is not in BCNF. Suppose X is decomposed into two schemas Y and Z , where $Y = (PR)$ and $Z = (QRS)$.

Consider the two statements given below.

I. Both Y and Z are in BCNF
 II. Decomposition of X into Y and Z is dependency preserving and lossless

Which of the above statements is/are correct?

☐ A. Both I and II
☐ B. I only
☒ C. II only
☐ D. Neither I nor II

option (C)

$R_1(P, R)$
 $R_2(Q, R, S)$
 $R_1 R_2 = R$

$\rightarrow R^+ = \{R, P\}$ is CK in R_1
 $F_1 = \{R \rightarrow P\}$
 $\Rightarrow$ Lossless

$\rightarrow R_1 = Y$ is in BCNF

$R_2 = Z(Q, R, S)$

$F_2 = \{QR \rightarrow S, S \rightarrow Q\}$

CK in R_2 not a SK
 $\Rightarrow$ not BCNF

$F_1 \cup F_2 = F \Rightarrow$
 dependency preserving

Converting into 2NF

eg:- $R(a,b,c)$
 $F = \{b \rightarrow c\}$
 $CK = ab$
nonprime = $\{c\}$

$ab \rightarrow c$ is a 2NF violation since
 $ab \rightarrow c$ is partial
 $b \rightarrow c$ holds
 $c \rightarrow$ nonprime.

But why we don't want partial dependency & why to convert
→ To avoid redundancy.

eg:-

a	b	c
1	2	haha
2	2	haha
3	2	haha
3	1	mole
4	1	mole

redundancy.

decompose to $R_1(ab)$ & $R_2(bc)$ → we take common attribute to be a super key so we can't split into $R_1(ab)$ $R_2(ac)$ since a is not sk. in either of them.
↓
lossless decomposition

PROCEDURE

- Let's assume R not in 2NF
- So there is at least one violation of 2NF
- We have violation $X \rightarrow Y$ where X is proper subset of CK & Y is nonprime
- So put $\{X^+\}$ separated from original table.
- So our decomposition

$R_1(R - X^+ + X)$, $R_2(X^+)$
↓ common attribute X to make lossless
↓ common attribute is SK.

→ if R_1 is not in 2NF → then apply recursively.

The above algorithm always gives lossless 2NF decomposition but it may/maynot be dependency preserving.

But it is always possible to create 2NF decomposition such that it is lossless & dependency preserving.

eg:- $R(ABCDEF)$
 $F = \{A \rightarrow BC, D \rightarrow E, AD \rightarrow F\}$
 $A^+ = \{A, B, C\}$
 $AD^+ = \{A, B, C, D, E, F\} = R \rightarrow CK$ (only)
 $B, C, E, F \rightarrow$ non prime
 $A \rightarrow BC \rightarrow$ violation
 $D \rightarrow E \rightarrow$ violation

$R_2(a^+), R_3(d^+), R_1(R_1 - a^+ + a - d^+ + d)$

$R_2(ABC), R_3(DE), R(A, D, F) \Rightarrow$ Lossless & dependency preserving.

$A \rightarrow BC$
 $A = SK$ in R_2
 $D \rightarrow E$
 $D = SK$ in R_3

→ Every Relation can be decomposed into 2NF such that it will be lossless and dependency preserving

eg 2:- $R(SFLPNH)$
 $S \rightarrow FL$
 $P \rightarrow N$
 $SP \rightarrow H$
 $CK \rightarrow SP$ — only CK
 $P \rightarrow N$ violation
 $S \rightarrow FL$ violation

$R_1(SPH), R_2(PN), R_3(SFL)$
 $SP \rightarrow H$ $P \rightarrow N$ $S \rightarrow FL$
 } lossless & dependency preserving

$R(A, B, C, D)$
 FDset: $\{AB \rightarrow CD, B \rightarrow D, C \rightarrow D\}$
 AB — only CK
 $C, D \rightarrow$ non prime
 $B \rightarrow D \rightarrow$ violation
 $R_1(A, B, C)$ $R_2(B, D)$
 Lossless but not dependency preserving.

Synthesis $AB \rightarrow C, B \rightarrow D, C \rightarrow D$

$R_1(ABC)$ $R_2(BD)$ $R_3(CD)$
 2NF, lossless & dependency preserving.

Understanding Redundancy

→ Trivial FDs do not cause any redundancy

Take example

Sid	S_name	DOB	district	state

$S_id \rightarrow CK$
 $district \rightarrow state$ (non trivial FD)

→ Assume 1000 students from 'jaipur'
 → 'Rajasthan' repeats 1000 times
 Redundancy

eg 2. $R(ABC) \{AB \rightarrow C, C \rightarrow B\}$
 $CK - AB, AC$

$AB \rightarrow C$
 $\downarrow$
 $CK \rightarrow$ cannot create redundancy since CK never repeat

$C \rightarrow B$
 $\rightarrow$ cause redundancy
 $LHS \neq CK$

Observation

- $\rightarrow$ Only non trivial FD can create redundancy.
- $\rightarrow$ FD involving non SKs on RHS cause redundancy
- $\rightarrow$ FD involving SKs on LHS doesn't cause redundancy.
- $\rightarrow$ BCNF has 0% redundancy due to FD since BCNF doesn't have FDs of the form $\alpha \rightarrow X$ where α is not a SK.
- $\rightarrow$ BCNF is still not free from redundancy in general but only free from redundancy due to FD.

eg:- Courses (sid, cid)

many to many relation

- $\rightarrow$ a student can take many courses
- $\rightarrow$ Many students can take a single course

eg: Movies (sid, movies)

- $\rightarrow$ A student watch many movies
- Many students watch single movie

Course

sid	cid
John	DBMS
John	AI
John	ML
Smith	ML
Doh	ML
King	ML

$sid \not\rightarrow cid$
 $cid \not\rightarrow sid$

No redundancy $\rightarrow$ No FD caused redundancy.

- $\rightarrow$ Individually course & movies relation has no information.
- $\rightarrow$ But what happens when we merge this. (Natural Join)

sid	movies
John	Hangover
John	Ton Wick
John	Dune

no redundancy

sid	cid	movies
John	DBMS	Hangover
John	AV	Hangover
John	ML	Hangover
John	DBMS	Jon Wick
John	AV	Jon Wick
John	ML	Jon Wick
John	DBMS	Dune
John	AV	Dune
John	ML	Dune

it is in BCNF

→ No nontrivial FD

→ Still there is redundant data in the natural join

→ Because more than one many to many relationship in a single table

→ This causes Multivalued dependency
M V D

→ Every value of course causes repetition of movies
→ Every value of movie causes repetition of course

MVD

→ Concepts of MVD comes in 4NF, 5NF
→ These concepts not required for GATE

Relation R with an associated set of functional dependencies, F , is decomposed into BCNF. The redundancy (arising out of functional dependencies) in the resulting set of relations is

- A. Zero
B. More than zero but less than that of an equivalent 3NF decomposition
C. Proportional to the size of F^+
D. Indeterminate

→ it only mentioned redundancy

→ not explicitly saying arising out of functional dependencies.

Decomposing BCNF

→ When can we say given a set of relation $S = \{R_1, R_2, R_3, R_4\}$ that the set S is in BCNF → when all relations are in BCNF.

→ Database S is in BCNF then all relations are in BCNF.

eg:- $R(ABCD)$
 $\{A \rightarrow B, C \rightarrow D\}$
 $R_1(A, B)$ $R_2(C, D)$
BCNF BCNF

but it is not good, No common attributes
→ Lossy

→ even if Lossy we still call it BCNF

Another goal is to have each individual relation R_i in the decomposition D be in BCNF or 3NF. However, this condition is not sufficient to guarantee a good database design on its own. We must consider the decomposition of the universal relation as a whole, in addition to looking at the individual relations. To illustrate this point, consider the EMP_LOCS(Ename, Plocation) relation in Figure 14.5, which is in 3NF and also in BCNF. In fact, any relation schema with only two attributes is automatically in BCNF.⁶ Although EMP_LOCS is in BCNF, it still gives rise to spurious tuples when joined with EMP_PROJ (Ssn, Pnumber, Hours, Pname, Plocation), which is

Example 3.17 suggests that perhaps all we have to do is break a relation schema into two-attribute subsets, and the result is surely in BCNF. However, such an arbitrary decomposition will not satisfy condition (2), as we shall see in Section 3.4.1. In fact, we must be more careful and use the violating FD's to guide our decomposition.

→ if Asked for BCNF decomposition → give Lossless BCNF decomposition & if possible dependency preserving.

→ if given decomposition & asked about BCNF → ignore Lossy / Lossless → check all subrelation for BCNF

BCNF decomposition Algorithm

→ guarantee lossless BCNF decomposition

- ① If R is already in BCNF → stop
- ② Else take violation: $X \rightarrow Y \mid X \neq SK$.
- ③ Find $R_1 = X^+$, $R_2 = R - X^+ \cup \{X\}$
- ④ Repeat iteratively.

LHS will be common
ie X will be common in R_1 & R_2
 $X \rightarrow R_1$ since $R_1 = X^+$
so X is SK in $R_1 \Rightarrow$ lossless

eg:- $R(A, B, C)$ perform BCNF decomposition

$A \rightarrow B$
↓
violation

CK → AC → only CK

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

$R_1 = \{A, B\}$ $F_1 = \{A \rightarrow B\}$

$R_2 = \{A, C\}$ $F_2 = \{ \}$

→ Lossless
→ BCNF

φ $R(A, B, C, D)$

$A \rightarrow B$
 $A \rightarrow C$ } violation

key (AD) only CK.

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

$R_1(ABC)$

$R_2(AD)$

$A \rightarrow BC$

AD is CK

A is CK

BCNF

$R(ABCD)$

$A \rightarrow B ; B \rightarrow C$

AD → only CK

$R_1(ABC)$

$R_2(AD)$

$F_1 = A \rightarrow B$

$B \rightarrow C \rightarrow$ violation

key A

$R_1(BC)$

$B \rightarrow C$

$R_2(AD)$

$R_3(AB)$

$A \rightarrow B$

Lossless

☆ → BCNF decomposition need not be unique

BCNF decomposition or both lossless & dep preserving may not be possible for all relations. ☆

Example 3.19: Consider a relation with schema
 $\{ \overset{T}{\text{title}}, \overset{Y}{\text{year}}, \overset{S}{\text{studioName}}, \overset{P}{\text{president}}, \overset{A}{\text{presAddr}} \}$
 That is, each tuple of this relation tells about a movie, its studio, the president of the studio, and the address of the president of the studio. Three FD's that we would assume in this relation are

$\text{title year} \rightarrow \text{studioName}$
 $\text{studioName} \rightarrow \text{president}$
 $\text{president} \rightarrow \text{presAddr}$

$R(TY, SP, A)$
 $TY \rightarrow S$
 $S \rightarrow P$
 $P \rightarrow A$
 $\Rightarrow S \rightarrow A$
 TY is in $\neq$ CKs.
 $TY^+ = R$ — only CK
 violation.

$S^+ = \{SPA\} = R_1$
 $F_1 = \{S \rightarrow PA; P \rightarrow A\}$ violation
 $R_2 = \{TYS\}$
 $F_2 = \{TY \rightarrow S\}$
 $R_3(PA)$
 $R_4(SP)$
 $\{P \rightarrow A\}$
 $\{S \rightarrow P\}$
 $\rightarrow$ BCNF decompose

alternate
 $R_1(PA)$
 $R_2(TYSP)$
 $R_3(SP)$
 $R_4(TYS)$
 $F_2 = \{TY \rightarrow S; S \rightarrow P\}$ violation

$\phi) R(ABCDE) \quad F = \{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$
 $A^+ = \{ABCDE\} = R$ — CK
 $E^+ = R$
 $CD^+ = R$
 $BC^+ = R$
 $B^+ = \{B, D\}$
 $\Rightarrow B \rightarrow D$ is violation.
 $R_1(BD)$ — BCNF
 $R_2(ABCE) \quad F_2 = \{A \rightarrow BCE, E \rightarrow ABC, BC \rightarrow AE\}$
 all are SKs

$\phi) R(ABC) \quad AB \rightarrow C \quad AB \text{ } \neq \text{ } AC \text{ are CKs}$
 $C \rightarrow B$ violation
 $R_1(CB) \quad F_1 = \{C \rightarrow B\}$
 $R_2(AC)$
 $R_1 \cap R_2 = C \rightarrow R_1 = \text{lossless}$
 but not dependency preserving

$AB \rightarrow C$
 this dependency can't be preserved without putting A, B, C in a single table.

- ① BCNF
 - ② lossless
 - ③ dependency preserving
- Can't be achieve together for this example

For some relations, there is NO way to get all three following properties at once:

1. No redundancy due to FDs i.e. BCNF
2. Recoverability of Information i.e. Lossless Decomposition
3. Preservation of Dependencies

☆ BCNF + lossless is always possible
however need not be unique.

3NF Decomposition — Synthesis Algorithm → Always Lossless & Dependency preserving.

→ Obvious algorithm → BCNF decomposition → Lossless 3NF guaranteed.
Synthesis algorithm → Lossless & dependency preserving 3NF guaranteed.

Synthesis Algorithm (Synthesys 3NF with lossless & dependency preserving given any R)

① Find minimal cover of FDs → F_m

② In F_c : if RHS is same, then merge the F^h s

③ For every FD $X \rightarrow A_1 A_2$ make a table $R(XA_1A_2)$

④ If none of the relation from step 3 contains some CK of the original relation R take any one CK & make one separate relation for it.

⑤ If for some i, j $R_i \subset R_j$ then delete R_i

→ It is called synthesis algo
• because checking 3NF is more computationally expensive than applying the synthesis algorithm
• So in practice, this is applied regardless of check 3NF

→ If the relation is already in 3NF, then this algorithm has no check for 3NF, but still it gives a 3NF subrelations.

eg:- $R(ABCDE)$ with $F: \{AB \rightarrow C, C \rightarrow B, A \rightarrow D\}$ find 3NF decompos

→ AE doesn't appear on RHS

→ $\{AE\}^+ = \{A, D, E\}$

→ $\{AEB\}^+ = \{A, B, C, D, E\} = R$ } only CKs

→ $\{AEC\}^+ = R$

→ $D \rightarrow$ non prime

→ $A \rightarrow D$ is a violation

non SK → non prime

Minimal Cover F_m

$AB \rightarrow C$

$C \rightarrow B$

$A \rightarrow D$

→ Already minimal

$R_1(ABC)$

$R_2(CB)$

delete
since
 $R_2 \subset R_1$

$R_3(AD)$

$R_4\{AEB\}$

for CK

→ $R_1(ABC)$ $R_3(AD)$ $R_4(AEB)$ → 3NF
lossless
FD preserving

→ If the relation is BCNF, then synthesis algorithm will not take it out of BCNF.

Since all FDs in F_m are of the form $\alpha \rightarrow x$ | $\alpha \rightarrow SK$; so all sub relations will also be in the form $\alpha \rightarrow x$ | $\alpha \rightarrow SK$.

☆ → For all relation it is always possible to decompose such that the sub relations are

- ① 2NF, lossless & FD preserving
 - ② 3NF, lossless & FD preserving
 - ③ BCNF & Lossless
- using synthesis

2. Exhibit a lossless join and dependency preserving decomposition of relation schema $R(A, B, C, D, E, F)$ by algorithm *Compute3NF*, assuming the FDs $\{A \rightarrow BCD, BC \rightarrow DE, B \rightarrow D, D \rightarrow A\}$ hold on R . Note that this requires showing (a) a minimal cover, and (b) a proof that one of the relations in the decomposition contains a candidate key.

CK
 $\rightarrow F$ must be part
 $AF^+ = \{A, B, C, D, E, F\}$
 $DF^+ = R$
 $BF = R$
 $CEF^+ = \{C, E, F\} \rightarrow$ not sk
 $\rightarrow$ non primes = $\{C, E\}$

$BC \rightarrow E$ $\leftarrow$ 2NF violation
 non sk
 non prime
 $A \rightarrow C$

Minimal Cover

$A \rightarrow BCD$
 $BC \rightarrow DE$
 $B \rightarrow D$
 $D \rightarrow A$

$A \rightarrow B$
 $A \rightarrow C$
 ~~$A \rightarrow D$~~
 ~~$BC \rightarrow D$~~
 $BC \rightarrow E$
 $B \rightarrow D$
 $D \rightarrow A$

$\rightarrow$ redundant
 $\rightarrow$ redundant

$A \rightarrow B$
 $A \rightarrow C$
 $BC \rightarrow E$
 $B \rightarrow D$
 $D \rightarrow A$

$\Rightarrow F_m = \{A \rightarrow BC, BC \rightarrow E, B \rightarrow D, D \rightarrow A\}$

$R_1(ABC) \quad R_2(BCE) \quad R_3(BD) \quad R_4(AD) \quad R_5(AF)$

$\rightarrow$ 3NF, lossless, dependency preserving

3NF, BCNF, 4NF, and beyond

Anomaly/normal form	3NF	BCNF	4NF
Lose FD's?	No	Possible	Possible
Redundancy due to FD's	Possible	No	No
Redundancy due to MVD's	Possible	Possible	No

❖ Of historical interests

- 1NF: All column values must be atomic

Q: A Decomposition D of Relation R (with FD set F) is dependency preserving then:

- Each dependency of F can be Enforced by dealing with an individual relation in D . ✓
- Each dependency of F can be Enforced combining one or more than one relations of D But may not be enforced by individual relations of D . ✗
- We need to Join all relations of D to Enforce dependencies of F . ✗
- Non

① While deletion, do we need to check any FD violation $\Rightarrow$ NO

$\rightarrow$ Deletion never violate F^D constraint

$\rightarrow$ ~~SD~~ insertion/modification $\rightarrow F^D$ constraint to be checked.

eg. $R(ABC) \quad FD \ F \{A \rightarrow B; B \rightarrow C; A \rightarrow C\}$

$R_1(AB)$

$R_2(BC)$

How to enforce $A \rightarrow C$ in this decomposition.
 $\rightarrow$ Since dependency preserving, we don't have to join.
 $\rightarrow$ It is enforced by transitivity.

B	A	C
x	1	2
x	1	3

We can guarantee this will never happen

A	B
1	x
1	x

$A \rightarrow B$

B	C
x	2
x	3

$B \rightarrow C$

$\rightarrow$ can't happen even

transitively $A \rightarrow C$

eg. $R(ABC)$ $F = \{ AB \rightarrow C, C \rightarrow B \}$



$C \rightarrow B$

B	C
2	3
2	4

no violation
in R_1

A	C
1	3
1	4

no violation
in R_2

A	B	C
1	2	3
1	2	4

$\rightarrow$ data violates $AB \rightarrow C$
 $\rightarrow$ can't check without N join

$\rightarrow$ Violation can happen if we don't check the table using natural join.

$\rightarrow$ We have to join R_1, R_2 before every modification / insertion to enforce all FDs if R_1, R_2 is not dependency preserving.

Q: A Decomposition D of Relation R (with FD set F) is NOT dependency preserving then:

- ☒ A. Each dependency of F can be Enforced by dealing with an individual relation in D.
- ☒ B. Some dependencies of F may be Enforced by dealing with an individual relation in D.
- ☒ C. Some dependencies of F need join of one or more relations of D to enforce them.
- ☒ D. All dependencies of F need join of one or more relations of D to enforce them.

$\rightarrow$ BCNF is not preferred since we can't always get FD preservation
 $\rightarrow$ 3NF is preferred $\rightarrow$ sacrifice some redundancy for reducing cost while checking of integrity constraint.

$\rightarrow$ Decomposition can be lossy or non FD preserving or not independent of the normal form the decomposition is in.

$\rightarrow$ All decomposition is in any normal form, if all sub relations are atleast in that form. $\rightarrow$ Highest common form among all sub relations.

★ For a database relation R:

- Every BCNF Decomposition is Lossless & Dependency Preserving. ☒
- Some BCNF Decomposition is Lossless & Dependency Preserving. ☒

$\rightarrow$ There may exist relation R in which BCNF, lossless & FD preservation all three are not possible

Which one of the following statements about normal forms is FALSE?

- ☒ A. BCNF is stricter than 3NF
- ☒ B. Lossless, dependency-preserving decomposition into 3NF is always possible
- ☒ C. Lossless, dependency-preserving decomposition into BCNF is always possible
- ☒ D. Any relation with two attributes is in BCNF

GATE CSE 1994 | Question: 3.6

asked Oct 5, 2014 • recategorized Apr 25, 2021 by Lakshman Bhalya

State True or False with reason

False

30

There is always a decomposition into Boyce-Codd normal form (BCNF) that is lossless and dependency preserving.

$\exists R$ | no decomposition has all 3 properties together

GATE CSE 2001 | Question: 2.23

asked Sep 14, 2014 • edited Jun 19, 2018 by Pooja Khatri

54,141 views

129 $R(A, B, C, D)$ is a relation. Which of the following does not have a lossless join, dependency preserving BCNF decomposition?

- A. $A \rightarrow B, B \rightarrow CD$
- B. $A \rightarrow B, B \rightarrow C, C \rightarrow D$
- C. $AB \rightarrow C, C \rightarrow AD$
- D. $A \rightarrow BCD$

B) $A^+ = ABCD$
 $B^+ = BCD$
 $C^+ = CD$
 $A = \text{only key}$
 $\Rightarrow B \rightarrow CD$
 $C \rightarrow D$ } violation

A) $A^+ \rightarrow CK$ — only CK.
 $B \rightarrow CD$ — violation.

$R_1(BCD)$ $R_2(AB)$ } BCNF
 $B \rightarrow BCD$ $A \rightarrow B$
 $R_1 R_2 \rightarrow R_1 \Rightarrow \text{lossless}$
 $F_1 \cup F_2 = F \Rightarrow F_D^h \text{ preserving}$

$R_1(BCD)$ $R_2(AB)$ $R_3(CD)$ $R_4(BC)$
 $B \rightarrow CD$ $A \rightarrow B$ $C \rightarrow D$ $B \rightarrow C$
 $C \rightarrow D$ — violation

$R_2(AB)$ $R_3(CD)$ $R_4(BC)$
 $A \rightarrow B$ $C \rightarrow D$ $B \rightarrow C$ $\equiv$ lossless & Dependency preserving.

C) $AB \rightarrow C$
 $C \rightarrow AD$
 B is in all key
 $AB = CK$
 $BC = CK$

$C \rightarrow AD$ — violation
 $R_1(CAD)$ $R_2(BC)$
 $C \rightarrow AD$
 $\Rightarrow$ BCNF & lossless
 But not dependency preserving.

D) $A \rightarrow BCD \rightarrow$ Already in BCNF

Let's see option C more carefully:

$R(ABCD)$

$AB \rightarrow C$
 $C \rightarrow AD$

similar to this

$\Rightarrow R_1(ABC)$

$R_2(ACD)$
 $C \rightarrow AD$

BCNF violation

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

but now

has to be the decomposition to preserve F_D^h
 $R_1 R_2 = AC \rightarrow R_2 \Rightarrow \text{Lossless}$

GATE CSE 2002 | Question: 2.24

asked Sep 15, 2014 • edited Feb 9, 2018 by DJ_1

11,151 views

51 Relation R is decomposed using a set of functional dependencies, F , and relation S is decomposed using another set of functional dependencies, G . One decomposition is definitely BCNF, the other is definitely 3NF, but it is not known which is which. To make a guaranteed identification, which one of the following tests should be used on the decompositions? (Assume that the closures of F and G are available).

- A. Dependency-preservation
- B. Lossless-join
- C. BCNF definition
- D. 3NF definition

3 doesn't tell anything.

both satisfy

Every Boyce-Codd Normal Form (BCNF) decomposition is

- 3 A. dependency preserving
- B. not dependency preserving
- C. need be dependency preserving
- D. none of these

- Which one of the following statements are not correct?
- 4 S1: 3NF decomposition is always lossless join and dependency preserving.
- S2: 3NF decomposition is always lossless join but may or may not be dependency preserving.
- S3: BCNF decomposition is always lossless join and dependency preserving.
- S4: BCNF decomposition is always lossless join but may or may not be dependency preserving.
- A. Only S1
B. Only S4
C. Both S1 and S4
D. Both S2 and S3

All statements are wrong

Question poorly framed

they were

think option (D)

3 levels of schema/abstraction

① External schema → what user wants

② Conceptual schema → what data is stored & how they are related (what we saw so far)

③ Internal schema / Physical schema → how data is represented in memory & stored

- Which datastructure used
- Which indexing type
- What is indexed
- How it is retrieved

External Schema (User's virtual view into the data base)

- ① Privacy
- ② What user wants
- ③ Derived & not stored

eg:- External schema has → course info (faculty, course, # no. of student)

Conceptual schema has → enroll (sid, lid)
course (cid, faculty)

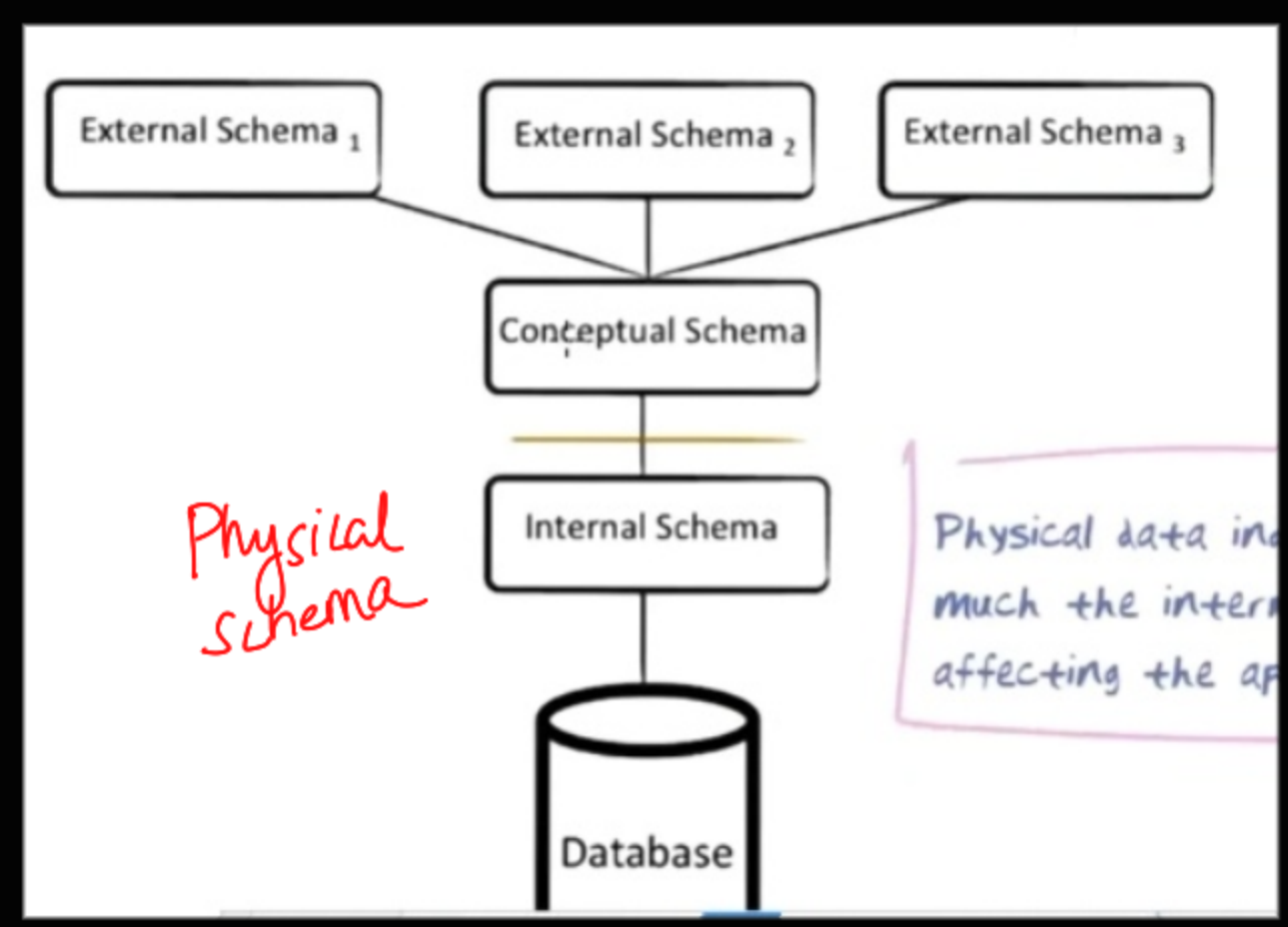
- not even stored
- This is derived
- This is what user wants

Data Independence

- ① Physical data independence → Independence of conceptual schema from physical changes
- ② Logical data independence

eg:- if we create new index → conceptual schema does not change

- External schema independent from conceptual schema change.
- we may have to change the application program
- Expectation from DBMS software → changes in conceptual schema does not force change in external application program.



State True or False with reason

False

Logical data independence is easier to achieve than physical data independence.

Changes in physical data independence is easier to achieve than Logical data independence

What does physical data independence allow the database administrator to do?

- a) Change the conceptual schema without changing the internal schema ☒
- b) Change the internal schema without changing the conceptual schema ☒
- c) Change the external schema without changing the conceptual schema ☐
- d) Change the application programs without changing the internal schema ☐

Normalisation Practice Questions

Consider the following relational schemes for a library database:

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

with the following functional dependencies:

- I. Title Author $\rightarrow$ Catalog_no
- II. Catalog_no $\rightarrow$ Title Author Publisher Year goclasses.in
- III. Publisher Title Year $\rightarrow$ Price

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

- A. Both Book and Collection are in BCNF
- B. Both Book and Collection are in 3NF only
- C. Book is in 2NF and Collection in 3NF ☒
- D. Both Book and Collection are in 2NF only goclasses.in

$B(T, A, C, P, Y, Q)$
 $C(T, A, C)$

$F = \{ TA \rightarrow C, C \rightarrow TAPY, PTY \rightarrow Q \}$

TA is CK in both tables B & C

$\rightarrow F_C = \{ TA \rightarrow C, C \rightarrow TA \} \rightarrow \text{BCNF}$
 $PTY \rightarrow Q \rightarrow \text{violation of BCNF}$

B | $TA^+ = B$
 $C^+ = B$ } keys.
 $P, Y, Q \rightarrow$ non prime. (not in 3NF)

For a database relation $R(a, b, c, d)$, where the domains a, b, c, d include only atomic values, only the following functional dependencies and those that can be inferred from them hold

- $a \rightarrow c$
- $b \rightarrow d$

<https://gateoverflow.in/2265/gate-cse-1997-question-6>

This relation is

- A. in first normal form but not in second normal form ☒
- B. in second normal form but not in first normal form
- C. in third normal form
- D. none of the above

Given the relational schema $R = (U, V, W, X, Y, Z)$ and the set of functional dependencies:

$\{U \rightarrow V, U \rightarrow W, WX \rightarrow Y, WX \rightarrow Z, V \rightarrow X\}$

Which of the following functional dependencies can be derived from the above set?

- ☒ A. $VW \rightarrow YZ$
- ☒ B. $WX \rightarrow YZ$
- ☒ C. $VW \rightarrow U$
- ☒ D. $VW \rightarrow Y$

$VW^+ = \{V, W, X, Z, Y\}$
 $WX^+ = \{W, X, Y, Z\}$

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

$\{CH \rightarrow G, A \rightarrow BC, B \rightarrow CFH, E \rightarrow A, F \rightarrow EG\}$ is a set of functional dependencies (FDs) so that F^+ is exactly the set of FDs that hold for R .

How many candidate keys does the relation R have?

- A. 3
- ☒ B. 4
- C. 5
- D. 6

$\rightarrow D$ has to be in every CK.

$AD^+ = \{A, B, C, D, F, H, G, E\} = R \Rightarrow \text{CK}$
 $ED^+ = R$
 $FD^+ = R$
 $BD^+ = R$ } 4 cks only

$CDHG \rightarrow CHDG$
 $CHD \rightarrow CHDG$
 $CHD \rightarrow CHD$
 $CHD \rightarrow CHD$

$\left. \begin{array}{l} \text{above} \\ \text{above} \\ \text{above} \\ \text{above} \end{array} \right\} = \text{CK only}$

$\left. \begin{array}{l} \text{above} \\ \text{above} \\ \text{above} \\ \text{above} \end{array} \right\} \text{not CK.}$

The symbol $\rightarrow$ indicates functional dependency in the context of a relational database. Which of the following options is/are TRUE?

- A. $(X, Y) \rightarrow (Z, W)$ implies $X \rightarrow (Z, W)$ $\times$
 B. $(X, Y) \rightarrow (Z, W)$ implies $(X, Y) \rightarrow Z$ $\checkmark$
 C. $((X, Y) \rightarrow Z \text{ and } W \rightarrow Y)$ implies $(X, W) \rightarrow Z$ $\checkmark$
 D. $(X \rightarrow Y \text{ and } Y \rightarrow Z)$ implies $X \rightarrow Z$ $\checkmark$
- BCD true

A functional dependency $F : X \rightarrow Y$ is termed as a useful functional dependency if and only if it satisfies all the following three conditions:

- X is not the empty set.
- Y is not the empty set.
- Intersection of X and Y is the empty set.

For a relation R with 4 attributes, the total number of possible useful functional dependencies is

We have seen general formula
 $= 3^n - 2^{n+1} + 1$

here $n=4$
 $\Rightarrow 3^4 - 2^5 + 1$
 $\Rightarrow 81 - 32 + 1 = 50$

more way to derive general formula

$$nC_1(2^{n-1}-1) + nC_2(2^{n-2}-1) + \dots + nC_{n-1}(2^{n-(n-1)}-1)$$

We can add $nC_n \cdot (2^{n-n}-1)$ since this term = 0

$$= \sum_{k=1}^n nC_k(2^{n-k}-1) = \sum_{k=1}^n nC_k \cdot 2^{n-k} - \sum_{k=1}^n nC_k$$

Binomial trick

$$= \sum_{k=1}^n nC_k \cdot 2^{n-k} \cdot 1 = \sum_{k=1}^n nC_k \cdot 2^{n-k}$$

$(1+2)^n$ if summation starts from 0

$$\Rightarrow 3^n - (1)2^n - (2^n - 1) = 3^n - 2 \cdot 2^n + 1 = 3^n - 2^{n+1} + 1$$

!! Exercise 3.1.3: Suppose R is a relation with attributes $A_1, A_2, \dots, A_n$. As a function of n , tell how many superkeys R has, if:

- a) The only key is A_1 . 2^{n-1}
 b) The only keys are A_1 and A_2 . $\rightarrow 2^{n-1} + 2^{n-1} - 2^{n-2} = 2^n - 2^{n-2}$
 c) The only keys are $\{A_1, A_2\}$ and $\{A_3, A_4\}$. $\rightarrow 2^{n-2} + 2^{n-2} + 2^{n-4} = 2^{n-1} + 2^{n-4}$
 d) The only keys are $\{A_1, A_2\}$ and $\{A_1, A_3\}$. $\rightarrow 2^{n-2} + 2^{n-2} - 2^{n-3} = 2^{n-1} - 2^{n-3}$

! Exercise 3.2.4: Show that each of the following are *not* valid rules about FD's by giving example relations that satisfy the given FD's (following the "if") but not the FD that allegedly follows (after the "then").

- a) If $A \rightarrow B$ then $B \rightarrow A$. $\times$
 b) If $AB \rightarrow C$ and $A \rightarrow C$, then $B \rightarrow C$. $\times$
 c) If $AB \rightarrow C$, then $A \rightarrow C$ or $B \rightarrow C$. $\times$

! Exercise 3.2.5: Show that if a relation has no attribute that is functionally determined by all the other attributes, then the relation has no nontrivial FD's at all.

$R(A_1 A_2 \dots A_n)$
 there is no FD of form $A_1 A_2 A_3 \dots A_n = A_i$

In a relational database relation, we say a non-empty set of attributes X is closed (with respect to a given set of functional dependencies FD) if $X^+ = X$ (where X^+ is the closure of X). Consider a relation with schema $R\{A, B, C, D\}$ and an unknown set of FD's.

If we are told which sets of attributes are closed, we can discover the FD's.

Assume that the only closed sets are $\{A, B\}, \{A, B, C, D\}$.

Which of the following is/are true for R ?

- ✓ A. The number of candidate keys is 2.
- ✓ B. The number of non-prime attributes is 2.
- ✓ C. R is in 2NF.
- D. R is in 3NF.

A, B, C

$$\begin{aligned} A^+ &= \{A, B\} \\ B^+ &= \{A, B\} \\ CA^+ &= \{A, B, C, D\} \\ CB^+ &= \{A, B, C, D\} \\ DA^+ &= \{A, B, C, D\} \\ DB^+ &= \{A, B, C, D\} \\ CD^+ &= \{A, B, C, D\} \\ C^+ &= R \\ D^+ &= R \end{aligned}$$

$$\begin{aligned} A &\rightarrow D \\ B &\rightarrow A \\ C &\rightarrow ABCD \\ D &\rightarrow ABCD \end{aligned}$$

A & B Nprime

$A \rightarrow B$ — 3NF violation

no composite keys $\Rightarrow$ 2NF

★ Doing exhaustively in alphabetic order allows us not to miss anything

Consider a relation with schema $R(A, B, C, D)$ and FD's $AB \rightarrow C, C \rightarrow D$, and $D \rightarrow A$.

What are all the nontrivial FD's that follow from the given FD's? You should restrict yourself to FD's with single attributes on the right side.

Case 1: LHS 1 attribute	LHS 2 attribute	LHS 3 attribute
$C \rightarrow D$	$AB \rightarrow C$	$ABC \rightarrow D$
$D \rightarrow A$	$AB \rightarrow D$	$ABD \rightarrow C$
$C \rightarrow A$	$AC \rightarrow D$	$BCD \rightarrow A$
	$BC \rightarrow D$	
	$BC \rightarrow A$	
	$BD \rightarrow A$	
	$BD \rightarrow C$	
	$CD \rightarrow A$	

$\Rightarrow 14$ FD's

★ In this question $\rightarrow$ Searching case by case alphabetically is required or will miss FDs and double count.

Consider a relation with attributes $R(A, B, C, D, E)$ that satisfies the following functional dependencies:

$AC \rightarrow B$

$BD \rightarrow C$

Consider a relation with attributes $R(A, B, C, D, E)$ that satisfies the following functional dependencies:

$AC \rightarrow B$

$BD \rightarrow C$

$CE \rightarrow D$

$DA \rightarrow E$

$EB \rightarrow A$

What is the number of keys that contain the attribute A?

$$\begin{aligned} A^+ &= \{A\} & ABC &= ABC \\ AB^+ &= \{A, B\} & ABD^+ &= \{A, B, C, D, E\} = R \text{ ①} \\ AC^+ &= \{A, B\} & ABE &= \{A, B, E\} \\ AD^+ &= \{A, D\} & ABD^+ &= \{A, B, C, D, E\} = R \text{ ②} \\ AE^+ &= \{A, E\} & ABE &= \{A, B, E\} \\ ABC^+ &= \{A, B, C\} & ACD^+ &= \{A, B, C, D, E\} = R \text{ ③} \\ ABD^+ &= \{A, B, C, D\} & ACE^+ &= \{A, B, C, D, E\} = R \text{ ④} \\ ABE^+ &= \{A, B, E\} & ADE^+ &= \{A, D, E\} \end{aligned}$$

★ going in alphabetic order exhaustively pays off. Just be patient & count