AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3
Database Design and Normalization
Lecture-1

Today's Target

- Functional dependency
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

(BCS-501- Database Management System)

Unit-III : Data Base Design & Normalization

Data Base Design & Normalization: Functional dependencies, normal forms, first, second, 8 third normal forms, BCNF, inclusion dependence, loss less join decompositions, normalization using FD, MVD, and JDs, alternative approaches to database design

AKTU : B.Tech (V-Sem)

S.N.	CS IT & CS Allied COMBO PACK (3 Subjects)
1	BCS501-Database Management System
2	BCS502: Web Technology
3	BCS503: Design and Analysis of Algorithm

Functional dependency

- A functional dependency, denoted by $\boxed{X \rightarrow Y}$, between two sets of attributes X and Y that are subsets of R specifies a constraint on the possible tuples that can form a relation state r of R.
- The constraint is that, for any two tuples t1 and t2 in r that have t1[X] = t2[X], they must also have t1[Y] = t2[Y].
- The abbreviation for functional dependency is FD or f.d.

f.d.:x->y R

$x \to y$ ✓

x	y
a	1
b	2
c	3
d	4

$x \not\to y$

x	y
a	1
a	2
c	3
d	4

ROLE OF FUNCTIONAL DEPENDENCY IN DATABASE DESIGN

1. Normalization: FDs help identify and eliminate data redundancy by guiding the decomposition of tables into smaller, related tables, which reduces anomalies (insertion, update, and deletion issues).

2. Determining Keys: FDs assist in identifying candidate keys in a relation, ensuring each tuple is unique and the data is properly organized.

3. Schema Optimization: They ensure an efficient database structure by organizing data into tables that accurately reflect real-world relationships without unnecessary duplication.

4. Data Integrity: FDs enforce constraints that maintain data consistency, making sure that related data entries align across tables.

TEACHS

TEACHER	COURSE	TEXT
SMITH	DS	BB
SMITH	DM	HH
HALL	COMPILER	XC
BROWN	DS	ZZ

1. TEXT → COURSE.(YES)
2. TEACHER → COURSE (NO)
3. TEXT → TEACHER (YES)
4. COURSE → TEXT (NO)

Ssn	Pnumber	Hours	Pname	Plocation
123456789	1	32.5	ProductX	Bellaire
123456789	2	7.5	ProductY	Sugarland
666884444	3	40.0	ProductZ	Houston
453453453	1	20.0	ProductX	Bellaire
453453453	2	20.0	ProductY	Sugarland
333445555	2	10.0	ProductY	Sugarland
333445555	3	10.0	ProductZ	Houston
333445555	10	10.0	Computerization	Stafford
333445555	20	10.0	Reorganization	Houston
999887777	30	30.0	Newbenefits	Stafford
999887777	10	10.0	Computerization	Stafford
987987987	10	35.0	Computerization	Stafford
987987987	30	5.0	Newbenefits	Stafford
987654321	30	20.0	Newbenefits	Stafford
987654321	20	15.0	Reorganization	Houston
888665555	20	NULL	Reorganization	Houston

A. Pnumber → {Pname, Plocation} Yes

B. {Ssn, Pnumber} → Hours Yes Pnumber → Hours ✗

All hold true SSN → Hours ✗

EMPLOYEE

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

a. Ssn → Ename

All hold true

Q Which of the following satisfied functional dependencies

1. A->BC (yes)
2. DE->C (yes)
3. C->DE (NO)
4. BC->A (yes)

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

Note x->y

- If all values of x are unique, Irrespective of the y the functional dependencies always hold good.
- If all value of y are same the functional dependencies always good.

NOTE: X->Y

- Y component of a tuple in r depend on, or are determined by, the values of the X component; alternatively, the values of the X component of a tuple uniquely (or functionally) determine the values of the Y component

Functional dependencies can be classified as trivial or non-trivial based on the relationship between the dependent attributes.

ABC → A

1. Trivial Functional Dependency:

A functional dependency $X \rightarrow Y$ is trivial, if $Y \subseteq X$

This means that the dependent attributes Y are already part of the determinant X, so the dependency provides no new information.

LHS RH

Example: $\{A,B\} \rightarrow A$ is trivial because A is already included in $\{A,B\}$.

2 .Non-Trivial Functional Dependency:

- A functional dependency $X \rightarrow Y$ is non-trivial if Y is not a subset of X.
- Here, the dependency implies that Y depends solely on X without being part of it, giving new information about data relationships.
- Example: $AB \rightarrow D$

x	y	z
1	4	2
1	5	3
1	6	3
3	2	2

Which of the following satisfies the functional dependencies

1. xy->z (T) and z->y (F) → False
2. yz->x (T) and y->z (T) – True
3. yz->x (T) and x->z (F) → False
4. xz->y (F) and y->z (T) → False

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

Which of the following satisfies the functional dependencies

1.A->B (F) and BC->A (T) False
2. C->A (F) and CA->B (T) False
3.B->C (T) and AB->C (T) – True
4.A->C (False) and BC->A (True) – False

Attribute closure/closure on attribute set/closure of attribute set

F^+

Attribute closure of an attribute set can be defined as set of attributes which can be functionally determined from it.

1. R(A,B,C)

A->B

B->C

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

2. R(A B C D E F)

A->B

C->DE

AC->F

D->AF

E->CF

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

3.R(A B C D E F G)

A->B

BC->DE

AEG->G

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

4..R(A B C D E)

A->BC

CD->E

B->D

E->A

B+ $\{B, D\}$

AKTU QUESTIONS

Q.1	What is functional dependency? Explain its role in database design.	AKTU 2018-19
Q.2	Explain the functional dependency? Explain trivial and non trivial functional dependency.	AKTU 2012-13/ AKTU 2021-22

Q ≡ (functional determined or not)

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-2

Today's Target

- Armstrong Axiom rule
- Equivalence of functional dependency
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

AKTU 2018-19,2021-22, 2022-23

ARMSTRONG AXIOM RULE/ INFERENCE RULE FOR FUNCTIONAL DEPENDENCY

Axiom is a statement that is taken to be true and serve as a premise or starting point for further argument.

1. Primary rules(RAT)

a. Axiom of Reflexivity: If A is a set of attributes and B is a subset of A, then A holds B. If B⊆A then A→B. This property is trivial property.

The axiom states:

If B is a subset of A then . A->B

- This means if we know the values of all attributes in A, we automatically know the values of all attributes in B because B is included in A.

Consider a Customer table with attributes {CustomerID, Name ,Age ,Address }

X={CustomerID,Name}

Y={Name}

CustomerID,Name}→{Name}

If you know both the CustomerID and the Name, you definitely know the Name, because it is already part of the information you have.

b. Axiom of Augmentation: If $A \rightarrow B$ holds and Y is the attribute set, then $AY \rightarrow BY$ also holds. That is adding attributes to dependencies, does not change the basic dependencies.

Consider a table Student(ID, Name, Age, Grade):

- Given: ID→Name (ID uniquely determines Name).
- By Augmentation: ID, Age →Name, Age.
- we augmented both sides by adding the attribute Age, and the dependency still holds.

c. Axiom of Transitivity: Same as the transitive rule in algebra, if $A \rightarrow B$ holds and $B \rightarrow C$ holds, then $A \rightarrow C$ also holds.

Consider a table Employee(ID, Department, Manager):

- Given:ID → Department
- Department → Manager (Department determines Manager).
- By Transitivity :ID → Manager.
- This shows that knowing the ID is enough to determine the Manager because of the chain of dependencies.

2. Secondary Rules

These rules can be derived from the above axioms.

a. Union: If A→B holds and A→C holds, then A→BC holds.

If X→Y and X→Z then X→YZ.

b. Composition: If A→B and X→Y hold, then AX→BY holds.

c. Decomposition: If A→BC holds then A→B and A→C hold.

d. pseudo Transitivity: If A→B holds and BC→D holds, then AC→D holds.

AKTU 2022-23

Soundness and Completeness of Armstrong's Axioms

- Armstrong's axioms are a set of rules used in relational databases to infer all functional dependencies (FDs) from a given set.
- These rules help maintain data integrity and are fundamental in the process of database normalization.

Soundness Definition:

- **Soundness** ensures that any functional dependency derived using Armstrong's axioms is logically correct and valid for the given set of FDs.
- **Explanation:** If we derive a new FD from the given set using Armstrong's axioms, that new FD must be true and consistent with the original set.

Example: Given FDs:

A → B

B → C

Using transitivity (one of Armstrong's axioms):

A → B

and B → C

imply A → C.

Since A → C logically follows from the given FDs, the derivation is sound.

Completeness

Let's consider a relation R with attributes {A, B, C, D} and the following functional dependencies:

$A \rightarrow B$

$B \rightarrow C$

$C \rightarrow D$

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

Objective: Show that using Armstrong's axioms, we can derive all possible functional dependencies that follow logically from the given set, demonstrating completeness.

Using Armstrong's Axioms to Derive New FDs

Applying Transitivity:

From $A \rightarrow B$ and $B \rightarrow C$, we can derive:

$A \rightarrow C$

From $A \rightarrow C$ and $C \rightarrow D$, we can derive:

$A \rightarrow D$

Applying Reflexivity:

We can derive trivial FDs like:

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

Applying Augmentation:

- From $A \rightarrow B$, we can augment with attribute C:
  - $AC \rightarrow BC$ (though it may not always be useful in this context).
- Similarly, from $B \rightarrow C$, augment with D:
  - $BD \rightarrow CD$.

List of All Possible Inferred FDs

AKTU 2018-19,2021-22, 2022-23

From the given set and using Armstrong's axioms, we derived:

$A \rightarrow C$

$A \rightarrow D$

$B \rightarrow C$

$C \rightarrow D$

$A \rightarrow B$ (given)

Reflexive FDs like $A \rightarrow A$, $B \rightarrow B$, etc.

- These are all the possible functional dependencies that can be logically inferred.

Armstrong's axioms allowed us to derive every FD that follows from the original set ($A \rightarrow C$ and $A \rightarrow D$).This shows completeness, as no FD that logically follows from the given set was left out.

- Completeness means that **Armstrong's axioms are capable of deriving all possible functional dependencies that logically follow from the given set.**

- If there are any FDs that can be inferred based on the given set, Armstrong's axioms should be able to derive them. No FD should be missed.

Equivalence of functional dependency

R(A C D E H)	
F A->C AC->D E->AD E->H	G A->CD E->AH

using G

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

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

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

$F \subseteq G$

using F

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

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

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

using F

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

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

$G \subseteq F$

using G

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

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

$F == G$

Equivalence of functional dependency

R(P Q R S)

X:	Y:
P->Q	P->QR
Q->R	R->S
R->S	

using

$P^+ = \{P, Q, R, S\}$

$Q^+ = \{Q\}$

$R^+ = \{R, S\}$

$X \not\subseteq Y$

using X

$P^+ = \{P, Q, R, S\}$

$Q^+ = \{Q, R, S\}$

$R^+ = \{R, S\}$

Using X

$P^+ = \{P, Q, R, S\}$

$R^+ = \{R, S\}$

$Y \subseteq X$

using Y

$P^+ = \{P, Q, R, S\}$

$R^+ = \{R, S\}$

$X \not\equiv Y$

Equivalence of functional dependency

R(A B C)

F	G
A->B	A->BC
B->C	B->A
C->A	C->A

Using G

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

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

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

$F \subseteq G$

Using F

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

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

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

$F = G$

Using F

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

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

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

$G \subseteq F$

Using G

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

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

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

Equivalence of functional dependency

R(W X Y Z)

F:	G:
W->X	W->XY
WX->Y	Z->XY
Z->WY	
Z->Y	

Using G

$W^+ = \{W, X, Y\}$

$(WX)^+ = \{W, X, Y\}$

$Z^+ = \{Z, X, Y\}$

$G \not\supseteq F$

Using F

$W^+ = \{W, X, Y\}$

$WX^+ \ \{W, X, Y\}$

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

Using F

$W^+ = \{W, X, Y\}$

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

$G \subseteq F$

Using G

$W^+ = \{W, X, Y\}$

$Z^+ = \{Z, X, Y\}$

$F \not\supseteq G$

AKTU QUESTION

Q.1	Describe Armstrong axiom in detail. What is the role of these rules in database development process.	AKTU2021-22 AKTU 2022-23
Q.2	List the Armstrong axioms for functional dependencies ? What do you understand by soundness and completeness of these axioms.	AKTU 2018-19 AKTU 2020-21
Q.3	What are the RAT axioms	AKTU 2018-19

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-3

Today's Target

- Canonical cover
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

Role of the axiom rules in database development process

- The Armstrong axioms are a set of rules used in database theory to infer all possible functional dependencies (FDs) in a relational database.
- They play an important role in database normalization and schema design.

1. Validation of Functional Dependencies:

- Armstrong axioms help in checking whether a given set of FDs logically implies another FD.
- This is crucial to ensure the database schema respects all the necessary relationships.

2. Normalization Process:

- They are used to decompose tables by finding candidate keys and reducing redundancy, ensuring the schema is in a higher normal form.

3. Schema Optimization:

- The axioms help simplify the set of FDs, making it easier to understand and optimize the schema.

(MINIMAL COVER/IRREDUCIBLE SET OF F.D./ Canonical Cover)

- Whenever a user updates the database, the system must check whether any functional dependencies are violated in this process.
- If there is a violation of dependencies in the new database state, the system must roll back. Working with a huge set of functional dependencies can cause unnecessary added computational time.
- This is where the canonical cover comes into play. A canonical cover of a set of functional dependencies F is a simplified set of functional dependencies with the same closure as the original set F
- A canonical cover is a minimal, equivalent set of functional dependencies with no redundancy in attributes or FDs

Procedure of canonical cover

In database management systems (DBMS), a canonical cover is a set of functional dependencies that is equivalent to a given set of functional dependencies but is minimal in terms of the number of dependencies.

The process of finding the canonical cover of a set of functional dependencies involves three main steps:

1.Reduction:

- The first step is to reduce the original set of functional dependencies to an equivalent set that has the same closure as the original set, but with fewer dependencies.
- This is done by removing redundant dependencies and combining dependencies that have common attributes on the left-hand side.

Procedure of canonical cover

2.Elimination: The second step is to eliminate any extraneous attributes from the left-hand side of the dependencies. An attribute is considered extraneous if it can be removed from the left-hand side without changing the closure of the dependencies.

3.Minimization: The final step is to minimize the number of dependencies by removing any dependencies that are implied by other dependencies in the set.

1. R(W X Y Z)

X->W

WZ->XY

Y->WXZ

Find out canonical / minimal cover

Decomposition

1. $X \to W$ ✓
2. $WZ \to X$ ✗
3. $WZ \to Y$
4. $Y \to W$
5. $Y \to X$
6. $Y \to Z$

① $X^+ = \{X, W\}$

Ignore $X \to W$

$X^+ = \{X\}$

② $WZ^+ = \{W, Z, X, Y\}$

Ignore $WZ \to X$

$WZ^+ = \{W, Z, Y, X\}$

this will be Removed

① $X \to W$ ✓
② $WZ \to Y$ ✓
③ $Y \to W$
④ $Y \to X$
⑤ $Y \to Z$

$WZ^+ = \{W, Z, Y, X\}$

Ignore $WZ \to Y$

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

$Y^+ = \{Y, W, X, Z\}$

Ignore $Y \to W$

$Y^+ = \{Y, X, Z, W\}$ this will be ignored

① $X \to W$
② $WZ \to Y$
③ $Y \to X$ ✓
⑤ $Y \to Z$ ✓

$Y^+ = \{Y, X, Z, W\}$

Ignore $Y \to X$

$Y^+ = \{Y, Z\}$

$Y^+ = \{Y, Z, X, W\}$

Ignore $Y \to Z$ $= \{Y, X, W\}$

$X \to W$

$WZ \to Y$

$Y \to X$

$Y \to Z$

$WZ^+ = \{W, Z, Y, X\}$

$W^+ = \{W\}$

$Z^+ = \{Z\}$

$$\boxed{\begin{array}{l} X \to W \\ WZ \to Y \\ Y \to XZ \end{array}} \to \text{Minimal cover}$$

2. R(A B C D)

A->B

C->B

D->ABC

AC->D

Find canocial cover

Decomposition

1. $A \to B$ ✓
2. $C \to B$ ✓
3. $D \to A$ ✓
4. $D \to B$ ✗
5. $D \to C$
6. $AC \to D$

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

Ignore $A \to B$

$A^+ = \{A\}$

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

Ignore $C \to B$

$C^+ = \{C\}$

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

Ignore $D \to A$

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

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

Ignore $D \to B$

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

1. $A \to B$
2. $C \to B$
3. $D \to A$
4. $D \to C$ ✓
5. $AC \to D$ ✓

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

Ignore $D \to C$

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

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

Ignore $AC \to D$

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

$A \to B$

$C \to B$

$D \to A$

$D \to C$

$AC \to D$

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

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

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

$$\left.\begin{array}{l} A \to B \\ C \to B \\ D \to AC \\ AC \to D \end{array}\right\} \text{Minimal cover}$$

3. R(V W X Y Z)

V->W

VW->X

Y->VXZ

1. $V \to W$ ✓
2. $VW \to X$ ✓
3. $Y \to V$ ✓
4. $Y \to X$ X
5. $Y \to Z$

$V^+ = \{V, W, X\}$

Ignore $V \to W$

$V^+ = \{V\}$

$VW^+ = \{V, W, X\}$

Ignore $VW \to X$

$VW^+ = \{V, W\}$

$Y^+ = \{Y, V, X, Z, W\}$

Ignore $Y \to V$

$Y^+ = \{Y, X, Z\}$

$Y^+ = \{Y, V, X, Z, W\}$

Ignore $Y \to X$

$Y^+ = \{Y, V, W, X, Z\}$

1. $V \to W$
2. $VW \to X$
3. $Y \to V$
4. $Y \to Z$

$Y^+ = \{Y, V, Z, W, X\}$

Ignore $Y \to Z$

$Y^+ = \{Y, V, W, X\}$

$V \to W$

$VW \to X$

$Y \to V$

$Y \to Z$

$VW^+ = \{V, W, X\}$

$V^+ = \{V, W, X\}$

$W^+ = \{W\}$

$V \to W$

$V \to X$

$Y \to V$

$Y \to Z$

$$\boxed{\begin{array}{c} V \to WX \\ Y \to VZ \end{array}}$$

(Minimal Cover)

4. R(A B C)

A->BC

B->C

A->B

AB->C

Decomposition

$A \to B$ ✓
$A \to C$ ✗
$B \to C$
$A \to B$
$AB \to C$

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

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

① $A \to B$
② $B \to C$
③ $A \to B$
④ $AB \to C$

$B^+ = \{B, C\}$
Ignore $B \to C$
$B^+ = \{B\}$

$A^+ = \{A, B, C\}$
③ $A \to B$ Ignore
$A^+ = \{A, B, C\}$

① $A \to B$
② $B \to C$
③ $AB \to C$ ✗

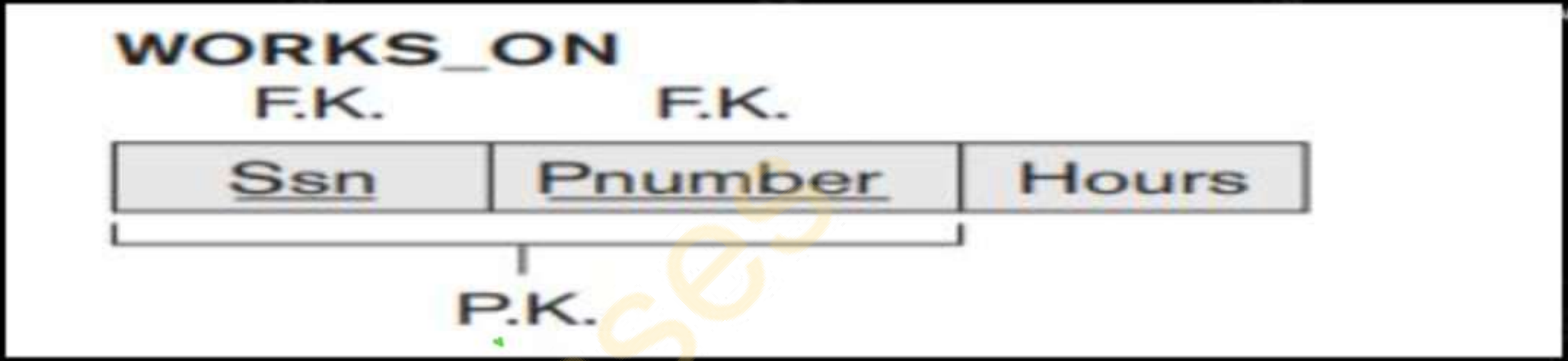
$AB^+ = \{A B C\}$
Ignore $AB \to C$
$AB^+ = \{A B, C\}$

$A \to B$
$B \to C$
↓
minimal cover

$A \to C$ ✓

➢ An attribute of relation schema R is called **a prime attribute of R** if it is a member of some candidate key of R.

➢ An attribute is called nonprime if it is not a prime attribute—that is, if it is not a member of any candidate key.



➢ Both Ssn and Pnumber are prime attribute

➢ Hours are non-prime attribute

P.K = SSN, Pnumber

Work on
(SSN, Pnumber, Hours)

SSN, Pnumber → Prime attribute

Hours → Non-prime attribute

R(ABCDEFG)

{AB C.K1

{CDE C.K.2 $\Rightarrow$ {Non-prime F, G}

A, B, C, D, E → Prime attributes

AKTU QUESTION

Q.1	Explain the procedure of calculating of the canonical cover of a given functional dependency with example	AKTU 2021-22
Q.2	Find the canonical cover/minimal cover X->W WZ->XY Y->WXZ	AKTU 2022-21
Q.3	A set of FDs for the relation R{A, B, C, D, E, F} is AB →C, C → A, BC → D,ACD → B, BE → C, EC → FA, CF → BD, D→ E. Find a minimum cover forth is set of FDs	AKTU 2022-23

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-4

Today's Target

- Find candidate key
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

Find the candidate keys

1. $R(A\ B\ C\ D)$

A->BC

AD essential attribute

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

$(Key)^+ = R$

2. $R(A\ B\ C\ D)$

ABC->D

AB->D

A->BD

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

C.K – P.K.

3. $R(A\ B\ C\ D)$

AB->C

C->BD

D->A

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

✓ $AB^+ = \{A, B, C, D\}$
× $AD^+ = \{A, D\}$
✓ $BD^+ = \{B, D, A, C\}$

C, AB, BD – C.K.

4. $R(A\ B\ C\ D)$

B-> ACD

ACD->B

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

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

$\{B, ACD\}$ C.K

Find the candidate keys

5. R(A B C D)

D essential

A->B

B->C

C ->A

$D^+ = \{ D \}$

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

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

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

AD BD CD – C.K.

6. R(A B C D)

AB->CD

D ->A

$B^+ = \{B\}$

$AB^+ = \{A, B, C, D\}$ ✓

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

$DB^+ = \{D, A, B, C\}$ ✓

AB, DB – C.K.

7. R (A B C D E F)

AB->C

C->D

B->AE

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

C.K.

8. R(A B C D)

AB->CD

C->A

D->B

$A^+ = \{A\}$ ×

$B^+ = \{B\}$ ×

$C^+ = \{C, A\}$ ×

$D^+ = \{D, B\}$ ×

$AB^+ = \{A, B, C, D\}$ ✓

$AC^+ = \{A, C\}$ ×

$AD^+ = \{A, D, B, C\}$ ✓

$BC^+ = \{B, C, A, D\}$ ✓

$BD^+ = \{B, D\}$ ×

$CD^+ = \{C, D, B, A\}$ ✓

AB, AD, BC, CD

C.K.

9. R(A B C D E)

AB->CD

D->A

BC->DE

$B^+ = \{B\}$

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

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

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

$EB^+ = \{E, B\}$ ×

AB, CB, DB C.k.

10. R (A B C D E F)

AB->C

DC->AE

E->F

$BD^+ = \{B, D\}$ ×

$ABD^+ = \{A, B, D, C, E, F\}$ ✓

$CBD^+ = \{C, B, D, A, E, F\}$ ✓

$EBD^+ = \{E, B, D, F\}$ ×

$FBD^+ = \{F, B, D\}$ ×

$FEBD^+ = \{F, E, B, D\}$ ×

ABD, CBD

11. R(W X Y Z)

Z->W

Y->XZ

XW->Y

$X^+ = \{X\}$

$Y^+ = \{Y, X, Z, W\}$

$W^+ = \{W\}$

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

Y, XW, XZ – C.k.

XW^+ $\{X, W, Y, Z\}$

$XZ^+ = \{X, Z, W, Y\}$

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

12. R(A B C D E)

CE->D

D->B

C->A

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

C.k. P.k.

AKTU QUESTION

Q.1	ANY candidate key question	AKTU 2018-19 AKTU 2019-20 AKTU 2021-22

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-5

Today's Target

- Candidate key and canonical cover
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

Find the candidate key

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

AB->C

AD->GH

BD->EF

A ->I

H->J

$(ABD)^+ = \{A, B, D, C, G, H, E, F, I, J\}$

C.K.

2 R(A B C D E)

A->B

BC->E

DE->A

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

$ACD^+ = \{A, C, D, B, E\}$ ✓

$BCD^+ = \{B, C, D, E, A\}$ ✓

$ECD^+ = \{E, C, D, A, B\}$ ✓

ACD, BCD, ECD C.K.

3 R(A B C D E)

BC-> ADE

D->B

$C^+ = \{C\}$

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

$BC^+ = \{B, C, A, D, E\}$ ✓

$DC^+ = \{D, B, C, A, E\}$ ✓

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

$\underbrace{BC, DC}_{C.K.}$

AC EC ✗

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

4. R(A B C $\boxed{D E}$ F)

AB->C

C->B

D->BF

E->F

F->A

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

C.K.

5. R(A B C D E F G H)

CH->G

A->BC

B->CFH

E->A

F->EG

$D^+ = \{D\}$

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

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

$CD^+ = \{C, D\}$ ×

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

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

$GD^+ = \{G, D\}$ ×

$HD^+ = \{H, D\}$ ×

CD, GD, HD

$CGD^+ = \{C, G, D\}$ ×

$CHD^+ = \{C, H, G, D\}$ ×

$CHGD^+ = \{C, H, G, D\}$ ×

$\left.\begin{array}{l} AD, BD, ED, FD \end{array}\right\}$ C.K.

6. Find the canonical cover/minimal cover

R(ABC) (AKTU 2018-19)

A->B

B->C

A->C

AB->B

AB->C

AC->B

① Decomposition
② Extra functional

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

ignore $A \to B$

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

✓ $B \to C$
✓ $A \to C$
× $AB \to B$
$AB \to C$
$AC \to B$

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

ignore $B \to C$

$B^+ = \{B\}$

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

ignore $A \to C$

$A^+ = \{A\}$

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

ignore $AB \to B$

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

$B \to C$
$A \to C$
$AB \to C$ ×
$AC \to B$

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

ignore $AB \to C$

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

$$\boxed{\begin{array}{l} B \to C \\ A \to C \\ AC \to B \end{array}}$$

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

ignore $AC \to B$

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

$B \to C$
$A \to C$
$AC \to B$

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

$$\boxed{\begin{array}{l} B \to C \\ A \to C \\ A \to B \end{array}}$$

$$\boxed{\begin{array}{l} B \to C \\ A \to BC \end{array}} \to \text{Canonical cover or Minimal cover.}$$

Find the canonical cover/minimal cover (AKTU 2017-18)

R(X Y Z)

F = {

X→YZ

Y→Z

X→Y

XY→Z

}

$X^+ = \{X, Y, Z\}$

$Y^+ \{Y, Z\}$

$Z^+ = \{Z\}$

Decomposition

1. X→Y X
2. X→Z
3. Y→Z
4. X→Y
5. XY→Z

$X^+ = \{X, Y, Z\}$

Ignore eq(1) X→Y

$X^+ = \{X, Z, Y\}$

1. X→Z $X^+ = \{X, Z, Y\}$
2. Y→Z Ignore X→Z
3. X→Y $X^+ = \{X, Y, Z\}$
4. XY→Z

✓ 1. Y→Z $Y^+ = \{Y, Z\}$

✓ 2. X→Y Ignore Y→Z

X 3. XY→Z $Y^+ = \{Y\}$

$X^+ = \{X, Y, Z\}$

Ignore X→Y

$X^+ = \{X\}$

$XY^+ = \{X, Y, Z\}$

Ignore XY→Z

$XY^+ = \{X, Y, Z\}$

$\boxed{\begin{matrix} Y \to Z \\ X \to Y \end{matrix}}$ Canonical Cover or Minimal Cover.

$X^+ = \{X, Y, Z\}$

$Y^+ = \{Y, Z\}$

$Z^+ = \{Z\}$

Rechecking

Consider another set of functional dependencies:

F={

B->A

AD->BC

C->ABD

}

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

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

Decomposition

✓ ① $B \to A$
× ② $AD \to B$
③ $AD \to C$
④ $C \to A$
⑤ $C \to B$
⑥ $C \to D$

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

$AD^+ = \{A, D, B, C\}$
② Ignore $AD \to B$
$AD^+ = \{A, D, C, B\}$

✓ ① $B \to A$
✓ ② $AD \to C$
× ③ $C \to A$
④ $C \to B$
⑤ $C \to D$

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

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

✓ $B \to A$
✓ $AD \to C$
✓ $C \to B$
✓ $C \to D$

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

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

$B \to A$
$AD \to C$
$C \to B$
$C \to D$

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

$B \to A$
$AD \to C$
$C \to B$
$C \to D$

{Canocial Cover}

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

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-6

Today's Target

- Normalization
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

Normalization

- Normalization in DBMS is the process of organizing data in a database to reduce redundancy and improve data integrity.
- It involves dividing a database into smaller tables and defining relationships between them to ensure data is stored efficiently and logically.

Need for normalization:

Normalization is essential to:

- Reduce Data Redundancy: Avoid storing duplicate data to save space.
- Improve Data Integrity: Maintain consistency across the database by eliminating anomalies.
- Simplify Maintenance: Make updates, deletions, and insertions error-free.

Employee

Emp_ID	Emp_ Name	Dept_ID	Dept-name	Dept_location
1	Alice	D01	HR	New York
2	Bob	D01	HR	New York
3	Charlie	D03	IT	San Francisco

- **Redundancy:** "HR" and "New York" are repeated for every employee in the same department.
- **Update Anomaly:** Updating the location of "HR" requires multiple changes.
- **Insertion Anomaly:** Cannot add a new department without at least one employee.
- **Deletion Anomaly:** Deleting all employees in "HR" removes department info.

Employee

Emp_ID	Demp_Name	FK Dept_ID
1	Alice	D01
2	Bob	D01
3	Charlie	D02

Department

Dept_ID	Dept_name	Dept_location
D01	HR	New York
D02	IT	San Francisco

Abhishek

- **No Redundancy:** Department info stored once.
- **Easy Updates:** Update "HR" location in one place.
- **No Anomalies:** Adding or deleting info is straightforward.

Normal form

1NF
2NF
3NF
BCNF → Numericals + theoretical

4NF
5NF → theoretical

1 NORMAL FORM

- It states that the **domain of an attribute** must include **only atomic** (simple, indivisible) **values** and that the value of any attribute in a tuple must be a **single value** from the domain of that attribute.

- **Not in 1NF**

ID	Name	Subjects
1	Alice	{Math,Science}

- **1NF**

ID	Name	Subjects
1	Alice	Math
1	Alice	Science

- 1NF disallows having a **set of values, a tuple of values,** or a **combination of both as an attribute value for a single tuple**. In other words, 1NF disallows relations within relations or relations as attribute values within tuples.

Relation within Relation, Not 1NF):

ID	Name	Address
1	Alice	Street: "Main St", City: "NYC")

1NF

iD	Name	Street	City
1	Alice	Main St	NYC

- 1NF are single atomic (or indivisible) values.

(a)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations

(b)

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

This violates the 1NF

There are three main techniques to achieve first normal form for such a relation

1.Remove the attribute Dlocations that violates 1NF and place it in a separate relation DEPT_LOCATIONS along with the primary key Dnumber of DEPARTMENT

DEPARTMENT

Dname	Dnumber	Dmgr_ssn
Research	5	333445555
Administration	4	987654321
Headquarters	1	888665555

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

2. Expand the key so that there will be a separate tuple in the original DEPARTMENT relation for each location of a DEPARTMENT.

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

This solution has the disadvantage of introducing redundancy in the relation and hence is rarely adopted.

3. If a **maximum number of values** is known for the attribute—for example, if it is known that at most three locations can exist for a department—replace the Dlocations attribute by three atomic attributes: Dlocation1, Dlocation2, and Dlocation3

DNAME	DNUMBER	DMGR_SSN	Dlocation1	DLOCATION2	Dloaction3
Research	5	33344555	Bellaire	SUGARLAND	HOUSTON
Administration	4	987654321	STAFFORD	NULL	NULL
Headquarter	1	888665555	HOUSTON	NULL	NULL

This solution has the **disadvantage of introducing NULL** values **if most departments have fewer than three locations.**

NOTE:

- Of the three solutions above, the first is generally considered best because it does not suffer from redundancy and it is completely general.
- It places no maximum limit on the number of values. In fact, if we choose the second solution.
- It will be decomposed further during subsequent normalization steps into the first solution.

First normal form also disallows multivalued attributes that are themselves composite. These are called nested relations because each tuple can have a relation within it.

EMP_PROJ(Ssn, Ename, {PROJS(Pnumber, Hours)})

(a)

EMP_PROJ

		Projs	
Ssn	Ename	Pnumber	Hours

(b)

EMP_PROJ

Ssn	Ename	Pnumber	Hours
123456789	Smith, John B.	1	32.5
		2	7.5
666884444	Narayan, Ramesh K.	3	40.0
453453453	English, Joyce A.	1	20.0
		2	20.0
333445555	Wong, Franklin T.	2	10.0
		3	10.0
		10	10.0
		20	10.0
999887777	Zelaya, Alicia J.	30	30.0
		10	10.0
987987987	Jabbar, Ahmad V.	10	35.0
		30	5.0
987654321	Wallace, Jennifer S.	30	20.0
		20	15.0
888665555	Borg, James E.	20	NULL

(c)

EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

Note:

- Every relation is in 1NF.
- This is because formal definition of a relation states that value of all the attributes must be atomic.

GW Question 1: Check whether it is in 1NF

CANDIDATE (Ssn, Name, {JOB_HIST (Company, Highest_position, {SAL_HIST (Year, Max_sal)})})

internal partial keys Company and Year

CANDIDATE (SSN, Name)

CANDIDATE_Jobhist (SSN, Company, Highest_position)

CANDIDATE_Jobhist_Salhist (SSN, Company, Year, Max_sal)

1NF

2 NF (2 NORMAL FORM)

A given relation is called in Second Normal Form (2NF) if and only if-

1. Relation already exists in 1NF.
2. No partial dependency exists in the relation.

Partial Dependency

A partial dependency is a dependency where a portion of the candidate key or incomplete candidate key determines non-prime attribute(s).

$A \rightarrow B$ is called a partial dependency if and only if-

- A is a subset of some candidate key
- B is a non-prime attribute.

If any one condition fails, then it will not be a partial dependency.

R(A B C D)

AB->D

B->C X

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

$\downarrow$

C.K

Prime attribute = {A, B}

Non-prime attribute = {C, D}

1NF

2NF X

$R_1(\underline{AB}D)$ $AB \rightarrow D$

$R_2(\underline{B}C)$ $B \rightarrow C$

$\alpha \to \beta$

$(ABC)^+$

$R(ABCDE)$

Prime → ABC

Non-prime DE

$AB C \to D$ ✓ 2NF ✓

$\underline{B} \to E$ 2NF ✗

$B \to C \to$ 2NF

$A \to C \to$ 2NF

$R_1(\underline{ABC}D)$ $ABC \to D$

$B \to C$

$A \to C$

$R_2(\underline{B}\,E)$ $B \to E$

AKTU QUESTIONS

Define Normal form. Explain 1Nf, 2 NF OR Any numerical to check whether table in 2NF	AKTU 2016-17 AKTU 2017-18 AKTU 2020-21 AKTU 2019-20

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3
Database Design and Normalization
Lecture-7

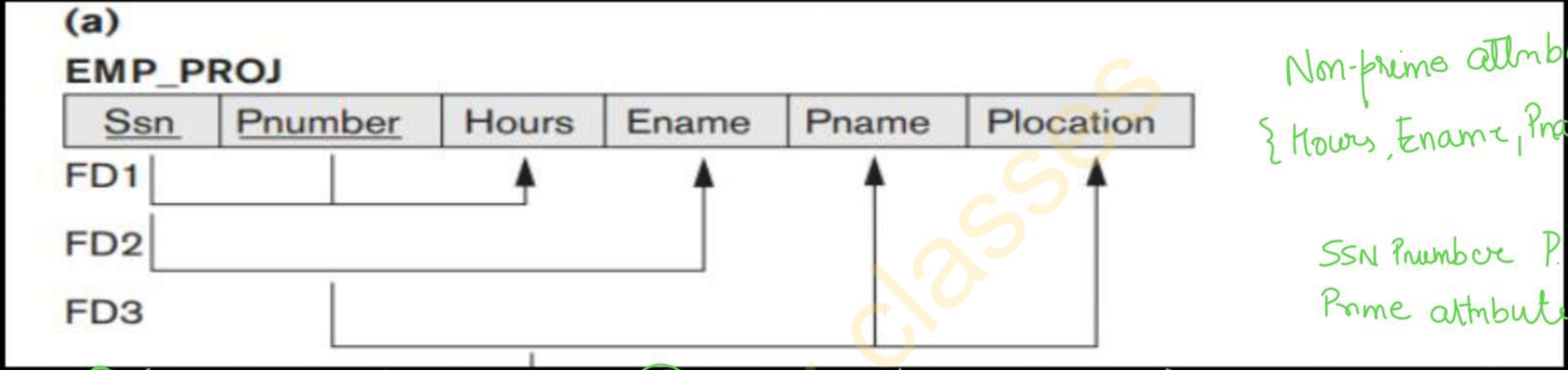
Today's Target

- 2NF NUMERICALS
- 3NF AND BCNF
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

2NF



Non-prime attribute
{Hours, Ename, Pname, Plocation}

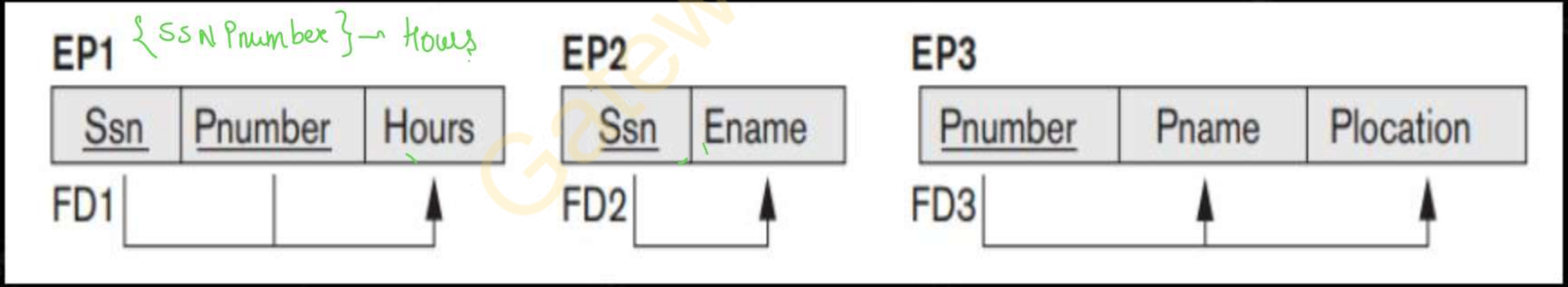
SSN Pnumber P.K, C.K
Prime attribute → SSN, Pnumber

1NF

✓ ① {SSN Pnumber} → Hours

X ② {SSN} → {Ename}

③ Pnumber → {Pname, Plocation}
X



GW

GATEWAY CLASSES

1. R(A B C D E)

AB->C

D->E

CHECK whether it is 2 NF if not convert it into.

$(ABD)^+ = \{A, B, D, C, E\}$ C.K

1NF ✓

2NF

Prime attribute = {A, B, D}

Non-prime attribute - {C, E}

A B → C (partial Dependency)

D → E (partial Dependency)

R_1 (ABD)

R_2 (ABC) AB → C

R_3 (DE) D → E

NOTE

$\boxed{P \rightarrow NP}$ X

NP → NP {Not our concern in 2NF}

2. R(A B C D E)

A->B

B->E

C->D

CHECK whether it is 2 NF if not convert it into

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

C.K

Prime attribute - {A, C}

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

1NF → Every relation is in 1NF

2NF A → B → Partial Depency X 2NF

B → E → Non-partial dependency

C → D → Partial Dependecny.

2NF R_1 (AC)

R_2 (ABE) A → B B → E

R_3 (CD) C → D

3. R(AB C D E F G H I J)

AB->C

AD->GH

BD->EF

A->I

H->J

CHECK whether it is 2 NF if not convert it into

$(ABD)^+ = \{A, B, C, D, G, H, E, F, I, J\}$

(find out the Key of a relation)

Check whether it in 1 NF, 2 NF if Not convert

$(ABD)^+$ c.k $\{A, B, D\}$ → Prime attribute

$\{C, E, F, G, H, I, J\}$ Non-prime attribute

1NF ✓

AB → C (P.D)

AD → GH (P.D)

BD → EF (P.D)

A → I (P.D)

H → J (NOT Partial Dependency)

R_1 (ABD)

R_2 (ABC) AB → C

R_3 (ADGHJ) AD → GH, H → J

R_4 (BDEF) BD → EF

R_5 (AI) A → I

Third Normal Form-

A given relation is called in Third Normal Form (3NF) if and only if-

- Relation already exists in 2NF. P → NP ✗ NP → NP ✓ P → P ✓
- No transitive dependency exists for non-prime attributes. NP → NP ✗ P → P ✓

Transitive Dependency

A → B is called a transitive dependency if and only if-

A is not a super key.

B is a non-prime attribute.

If any one condition fails, then it is not a transitive dependency.

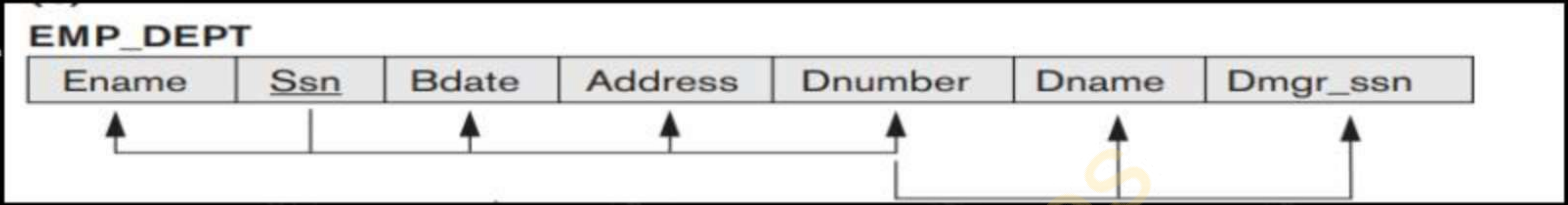
$\alpha \rightarrow \beta$ Sup ✗ (Non-prime)

Prove 3NF → 2NF

P → NP ✗ | P → NP ✗ (PD)
NP → NP ✗ | NP → NP ✓
P → P | P → P ✓

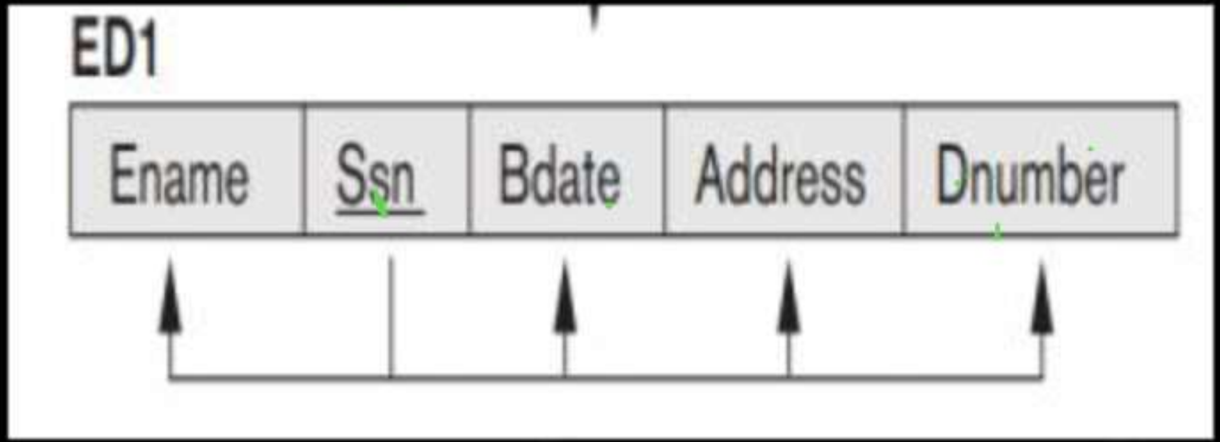
NOTE-

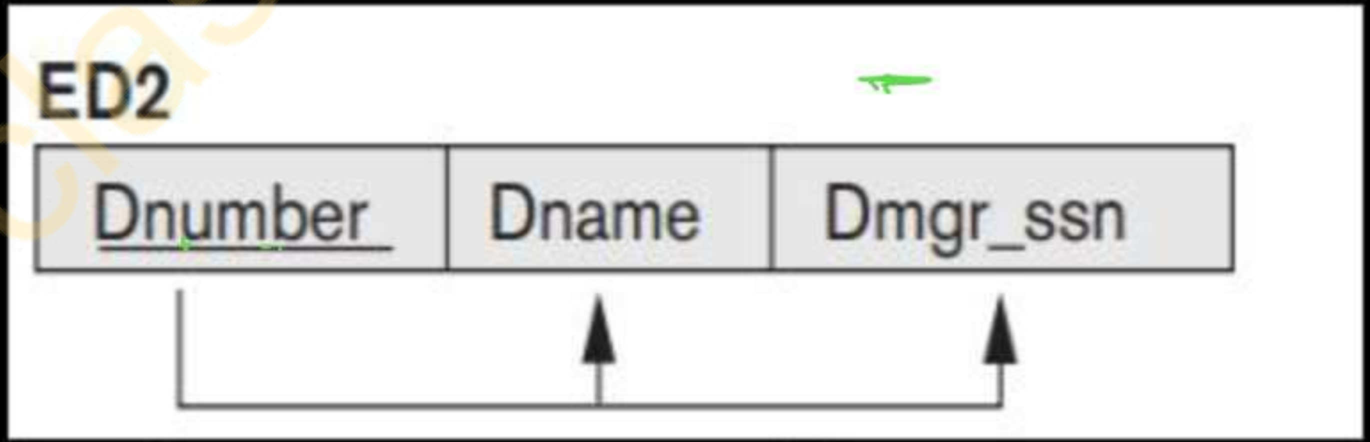
- Transitive dependency must not exist for non-prime attributes.
- However, transitive dependency can exist for prime attributes.



1NF ✓
2NF ✓

X3NF





Definition. According to Codd's original definition, a relation schema R is in 3NF if it satisfies 2NF and no nonprime attribute of R is transitively dependent on the primary key.

SUMMARY

NORMAL FORM	TEST	REMEDY
First (1NF)	Relation should have no multivalued attributes or nested relations	Form new relations for each multivalued attribute or nested relation.
Second (2NF)	For relations where primary key contains multiple attributes, no non key attribute should be functionally dependent on a part of the primary key	Decompose and set up a new relation for each partial key with its dependent attribute(s). Make sure to keep a relation with the original primary key and any attributes that are fully functionally dependent on it

SUMMARY

NORMAL FORM	TEST	REMEDY
THIRD (3NF)	Relation should not have a non key attribute functionally determined by another non-key attribute (or by a set of non-key attributes). That is, there should be no transitive dependency of a non-key attribute on the primary key.	Decompose and set up a relation that includes the non-key attribute(s) that functionally determine(s) other non-key attribute(s)non-key attribute(s)

1. R(A B C)

A->B

B->C

CHECK whether it is 3 NF if not convert it into

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

Prime attribute = {A}

Non-prime = {B, C}

R(ABC)

A→B (No PD) 1NF ✓

B→C (No P.D) 2NF ✓

3NF

A→B (No transitive Dependency)

B→C (transitive dependency) 2NF

R_1(A B) A→B (3NF)

R_2(B C) B→C

2.R(A B C D E)

A->B

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

C->D C.K

Prime attribute = {A, C}

Non-prime attribute = {B, D, E}

Check whether it is 3rd normal form or not if not convert it

A→B (Partial Dependency) 1NF ✓

B→E (No P.D) 2NF

C→D (No P.D)

R_1(A C)

R_2(A B E) A→B, B→E

R_3(C D) C→D

R_1(A C) 3NF

R_{21}(A B) A→B

R_{22}(B E) B→E

R_3(C D) C→D

GW GATEWAY CLASSES

3. R(A B C D E F G H I J)

AB->C

A->DE

B->F

F->GH

D->IJ

$(AB)^+ = \{A, B, C, D, E, F, G, H, I, J\}$

C.K. P.K.

Prime attribute = {A, B}

Non-prime attribute = {C, D, E, F, G, H, I, J}

1NF → Every Relation is in 1NF

2NF

AB→C {No P.D}

A→DE {P.D}

B→F {P.D}

F→GH {No P.D}

D→IJ {No P.D}

R_1(<u>AB</u>C) AB→C

R_2(<u>A</u>DEIJ)
A→DE, D→IJ

R_3(<u>B</u>FGH)
B→F, F→GH

3NF

R_1(<u>AB</u>C) AB→C

R_{21}(<u>A</u>DE) A→DE

R_{22}(<u>D</u>IJ) D→IJ

R_{31}(<u>B</u>F) B→F

R_{32}(<u>F</u>GH) F→GH

4. R(A B C D E)

AB->C

B->D

D->E

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

C.K. P.K

Check whether it is 3rd normal form or not if not convert it

Prime attribute = {A, B}

Non-prime attribute = {C, D, E}

1NF → Every Relation is in 1NF.

2NF

AB → C {No P.D}

B → D {P.D}

D → E {No P.D}

R_1(ABC) AB → C

R_2(BDE) B → D, D → E — 2NF

transitive dependency

R_1(ABC) AB → C ✓

R_{21}(BD) B → D ✓

R_{22}(DE) D → E ✓

5. R(A B C D E F G H I J)

AB->C (P.D) C.K.

AD->GH (P.D) $(ABD)^+ = \{A, B, C, D, G, H, E, F, I, J\}$

BD->EF (P.D) Prime = {A, B, D}

A->I (P.D) Non-prime = {C, E, F, G, H, I, J}

H->J (No P.D) 1NF ✓

Check whether it is 3rd normal form or not if not convert it

2NF

R_1(ABD)
R_2(ABC) AB→C
R_3(ADGHJ) AD→GH H→J
R_4(AI) A→I
R_5(BDEF) BD→EF

3NF (2NF) × No Transitive dependency

R_1(ABD)
R_2(ABC) AB→C
R_{31}(ADGH) AD→GH
R_{32}(HJ) H→J
R_4(AI) A→I
R_5(BDEF) BD→EF

Boyce-Codd Normal Form-

A given relation is called in BCNF if and only if-

- Relation already exists in 3NF.
- For each non-trivial functional dependency $A \rightarrow B$, A is a super key of the relation.

R(A B C)

AB->C

C->B

List all prime and non-prime attributes In Relation R(A,B,C,D,E) with FD set F = {AB→C, B→E, C→D}.

	Full Functional Dependency	Partial Functional Dependency
1.	A functional dependency X->Y is a fully functional dependency if Y is functionally dependent on X and Y is not functionally dependent on any proper subset of X.	A functional dependency X->Y is a partial dependency if Y is functionally dependent on X and Y can be determined by any proper subset of X.
2.	In full functional dependency, the non-prime attribute is functionally dependent on the candidate key.	In partial functional dependency, the non-prime attribute is functionally dependent on part of a candidate key.
3.	n fully functional dependency, if we remove any attribute of X, then the dependency will not exist anymore.	In partial functional dependency, if we remove any attribute of X, then the dependency will still exist.

	Full Functional Dependency	Partial Functional Dependency
4.	Full Functional Dependency equates to the normalization standard of Second Normal Form.	Partial Functional Dependency does not equate to the normalization standard of Second Normal Form. Rather, 2NF eliminates the Partial Dependency.
5.	An attribute A is fully functional dependent on another attribute B if it is functionally dependent on that attribute, and not on any part (subset) of it.	An attribute A is partially functional dependent on other attribute B if it is functionally dependent on any part (subset) of that attribute.

Consider the universal relation R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies F = { {A, B } → {C}, {A}→{D, E}, {B}→{F}, {F}→{G, H}, {D}→{I, J} }. What is the key for R? Decompose R into 2NF and then 3NF relations.(AKTU 2022-23)

AKTU PYQS

1.	Explain BCNF with example	AKTU 2017-18
2.	What is second normal form	AKTU 2017-18
3.	List all prime and non-prime attributes In Relation R(A,B,C,D,E) with FD set F = {AB→C, B→E, C→D}.	AKTU 2023-24
4	Consider the universal relation R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies F = { {A, B } → {C}, {A}→{D, E}, {B}→{F}, {F}→{G, H}, {D}→{I, J} }. What is the key for R? Decompose R into 2NF and then 3NF relations.	AKTU 2022-23

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

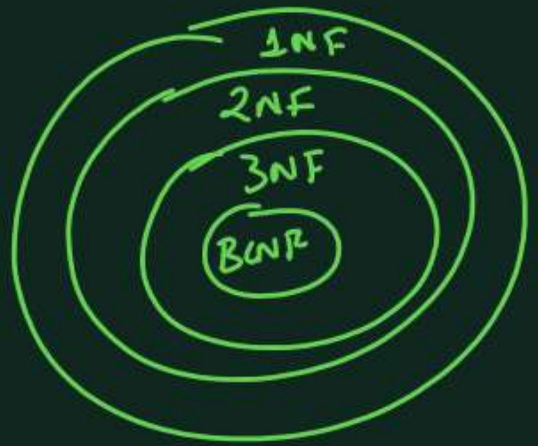
Lecture-8

Today's Target

- Numericals on functional dependency / Normal form
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)



either α is S.k or
β is prime

1NF – Multivalued ✗
2NF → No partial dependency $P \rightarrow NP$ ✓
3NF → 2NF + No transitive dependency $NP \rightarrow NP$
BCNF $P/NP \rightarrow P$ + 3NF
+ 2NF

$\alpha \rightarrow \beta$ — Prime
S.k
3NF   P

$P \rightarrow NP$
$NP \rightarrow NP$

3NF
2NF + No transitive

$P \rightarrow NP$ ✗
$NP \rightarrow NP$ ✗
$P+NP \rightarrow P$ ✗

α — β — BCNF
S.k

Given functional dependency it satisfies which normal form

1. R(A B C D E F G H)

AB->C

A->DE

B->F

F->GH

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

Prime attribute = {A, B}

Non-prime {C, D, E, F, G, H}

BCNF $\alpha \to \beta$ X

S.k

2NF X $P \to NP$

3NF $\alpha \to \beta$

S.k / Prim

1NF ✓

2. R(A B C D E)

CE->D

D->B

C->A

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

Prime attribute = {C, E}

Non-prime = {A, B, D}

BCNF → $\alpha \to \beta$ X

S.k.

3NF $\alpha \to \beta$ X

S.k Prim

2NF No partial $P \to NP$

$C \to A$ X

1NF

3. R(A B C D E F)

AB->C

AC->AE

E->F

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

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

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

$EBD = \{E, F, B, D\}$ x

$FBD = \{F, B, D\}$ x

$CEBD^+$ $CFBD^+$

{C, E, B, D, F} {F B, D}

(ABD)

Prime = {A, B, D}

Non-prime = {C, E, F}

BCNF $\alpha \to \beta$

S.k.

3NF $\alpha \to \beta$

S.k Prim

2NF → 1NF + No partial

1NF → ✓

GATEWAY CLASSES

Given functional dependency it satisfies which normal form

4. R(A B C D E F G H I)

AB->C

BD->EF

AD->GH

A->I

$(ABD)^+ = \{A, B, C, E, F, D, G, H, I\}$

Prime = {A, B, D}

Non-prime = {C, E, F, G, H, I}

BCNF $\alpha \to \beta$ ✗ (α: S.k.)

3NF $\alpha \to \beta$ (α: S.k., β: Prim)

2NF → P → NP ✗

1NF ✓

5. R(A B C D E)

AB->CD

D->A

BC->DE

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

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

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

C.K.

$EB^+ = \{E, B\}$ ✗

Prime = {A, B, C, D}

Non prim = {E}

$\alpha \to \beta$ BCNF (α: S.k.)

3NF $\alpha \to \beta$ (α: S.k., β: Prime)

6 R(A B C D E) (3NF)

BC->ADE

D->B

$C^+ = \{C\}$

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

C.k. $BC^+ = \{B, C, A, D, E\}$ ✓

$DC^+ = \{D, C, B, A, D, E\}$ ✓

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

Prime = {B, D, C}

Non-Prim = {A, E}

BCNF $\alpha \to \beta$ (α: S.k.)

3NF $\alpha \to \beta$ (α: S.k., β: Pri)

7.R(V,W X Y Z)

X->YV

Y->Z

Z->Y

VW->Y

WX = {W, X, Y, V, Z}

Prime = {W X}

Non-prime = {Y, V, Z}

BCNF $\alpha \to \beta$ (α: S.k.)

3NF $\alpha \to \beta$ (α: S.k., β: Pr)

2NF ✓ P → NP

1NF

Gateway classes

Given functional dependency it satisfies which normal form

8. R(A B C D E F)

ABC->D

ABD->E

CD->F

CDF->B

BF->D

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

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

$(EAC)^+ = \{E,A,C\}$ ✗

$(FAC)^+ = \{F,A,C\}$ ✗

$(EFAC)^+ = \{E,F,A,C\}$ ✗

(BAC, DAC) → C.k.

Prime = {A,B,C,D}

Non-prime = {E,F}

BCNF $\alpha \to \beta$, α: S.k.

3NF $\alpha \to \beta$, α: S.k, β: P

2NF P→NP ✗

1NF ✓

9. R(A B C)

A->B

B->C

C->A

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

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

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

$\alpha \to \beta$, α: S.k

BCNF ✓

10. R(A B C D E F)

A->BCDEF

BC->ADEF

DEF->ABC

A, BC, DEF: C.k.

$\alpha \to \beta$, α: S.k

BCNF

11. R(A B C)

AB->C

C->A

AB = {A,B,C}

CB = {C,B,A}

Prime = {A,B,C}

$\alpha \to \beta$, α: S.k ✗ BCNF

$\alpha \to \beta$, α: S.k, β: Prime 3NF ✓

12. R(A B C D E)

A->B

BC->E

DE->A

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

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

ECD = {E,C,D}

(ACD, BCD) C.k

$\alpha \to \beta$, α: S.k. BCNF ✗

$\alpha \to \beta$, α: SK, β: Prime 3NF ✗

P→NP 2NF ✗

1NF

Given functional dependency it satisfies which normal form

1.R(A B C D E)

AB->CD

D->A

BC->DE

C.k $\{AB^+ = \{A, B, C, D, E\}$, $CB^+ = \{B, C, D, E, A\}$, $DB^+ = \{D, A, B, C, E\}$, $EB^+ = \{E, B\}$

Prm: {A, B, C, D}

Nm-prm: {E}

BCNF $\alpha \to \beta$ x
S.k

3NF ✓ $\alpha \to \beta$
s.k Prm

2. R(W X Y Z)

Y, XW, XZ
C.k

Z->W

Y->XZ

XW->Y

$X^+ = \{X\}$

$Y^+ = \{Y, X, Z, W\}$ ✓

$W^+ = \{W\}$

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

$XW^+ = \{X, W, Y, Z\}$

$XZ^+ = \{X, Z, W, Y\}$

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

S.k
$\alpha \to \beta$ BCNF x

3NF $\alpha \to \beta$
S.k Pri 3NF ✓

3. R(A B C D E)

A->B

B->E

C->D

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

Prime = {A, C}

Non-prime = {B, D, E}

$\alpha \to \beta$ BCNF x
S.k.

$\alpha \to \beta$ 3NF x
S.k Prime

2NF P → NP X 1NF

4. R(A B C D E F)

AB->C

DC->AE

E->F

HW

5.R(V W X Y Z)

Z->Y

Y->Z

Y->YV

VW->X

6.R(A B C D E F)

ABC->D

ABD->E

CD->F

CDF->B

BF->D

Given functional dependency it satisfies which normal form

HW Pratice Questn 3

1. R(A B C D E F)

C->F

E->A

EC->D

A->B

2. R(A B C D E H)

A->B

BC->D

E->C

D->A

3. R(A B C D E P G)

AB->CD

DE->P

C->E

P->C

B->G

4. R(A B C D E F G H)

CH->G

A->BC

B->CFH

E->A

F->EG

Given functional dependency it satisfies which normal form

5 R(A B C D)

A->B

B->C

C->BD

6. R(A B C D E F)

AB->CD

CD->EF

BC->DEF

D->B

CE->F

	Full Functional Dependency	Partial Functional Dependency
1.	A functional dependency X->Y is a fully functional dependency if Y is functionally dependent on X and Y is not functionally dependent on any proper subset of X.	A functional dependency X->Y is a partial dependency if Y is functionally dependent on X and Y can be determined by any proper subset of X.
2.	In full functional dependency, the non-prime attribute is functionally dependent on the candidate key.	In partial functional dependency, the non-prime attribute is functionally dependent on part of a candidate key.
3.	n fully functional dependency, if we remove any attribute of X, then the dependency will not exist anymore.	In partial functional dependency, if we remove any attribute of X, then the dependency will still exist.

	Full Functional Dependency	Partial Functional Dependency
4.	Full Functional Dependency equates to the normalization standard of Second Normal Form.	Partial Functional Dependency does not equate to the normalization standard of Second Normal Form. Rather, 2NF eliminates the Partial Dependency.
5.	An attribute A is fully functional dependent on another attribute B if it is functionally dependent on that attribute, and not on any part (subset) of it.	An attribute A is partially functional dependent on other attribute B if it is functionally dependent on any part (subset) of that attribute.

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-9

Today's Target

- 4 normal form
- Lossless decomposition
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

Fourth normal form (4NF)

- A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency.

Condition 1:(MULTIVALUED DEPENDENCY)

For a dependency A →→B, if for a single value of A, multiple values of B exists, then the relation will be a multi-valued dependency.

Condition 2:

A table should have at least 3 column(a,b,c) to have multi value dependency.

Condition 3:

For a table with column A->B is a multivalued dependency then b and c should independent of each other.

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

The EMP relation with two MVDs: Ename →→ Pname and Ename →→ Dname.

EMP_PROJECTS

Ename	Pname
Smith	X
Smith	Y

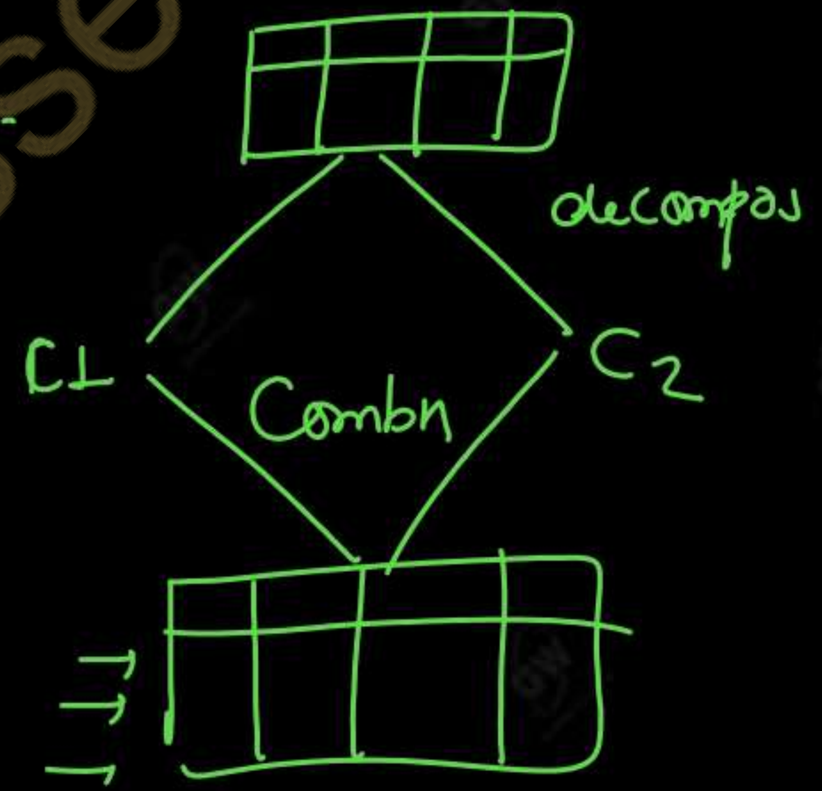
EMP_DEPENDENTS

Ename	Dname
Smith	John
Smith	Anna

Decomposing the EMP relation into two 4NF relations EMP_PROJECTS and EMP_DEPENDENTS

An MVD X →→ Y in R is called a trivial MVD if (a) Y is a subset of X, or (b) X ∪ Y = R

- The relation EMP_PROJECTS has the trivial MVD Ename →→ Pname and the relation EMP_DEPENDENTS has the trivial MVD Ename →→ Dname.
- An MVD that satisfies neither (a) nor (b) is called a nontrivial MVD. A trivial MVD will hold in any relation state r of R; it is called trivial because it does not specify any significant or meaningful constraint on R.



Lossless decomposition /non-additive

- This property gurantees that the extra or loss tuplen problem does occur afte rdecomposition.
- It Is mandatory that this property always hold good.
- I f a realtion R is decomposed into realtion r1 and r2 then it is lossless if and only if

1.att(r1) U attr(r2)=attr(R)

2. att(r1) ∩ attr(r2) not equal to Ø

3. att(r1) ∩ attr(r2) -> att(r1)/attr(r2)

R

R(ABCD)

A	B	C	D
1	a	p	x
2	b	q	y

r1

A	B
1	a
2	b

r2

C
p
q

r1(AB)U

r2(C)=R

ABC ≠ ABCD

$r1$

A	B
1	a
2	b

$r2$

C	D
p	x
q	y

$r1 \times r2$

A	B	C	D
1	a	p	x
1	a	q	y
2	b	p	x
2	b	q	y

Extra tuple

$\{AB\} \cap \{C, D\}$

ϕ

$attr(r1) \cap attr(r2) \neq \phi$

1. $attr(r1) \cup attr(r2) = attr(R)$

$\{AB\} \cup \{CD\} = R(ABCD)$

$ABCD = ABCD$

R

A	B	C
1	a	p
2	b	q
3	a	r

C.K.

r_1

A	B
1	a
2	b
3	a

r_2

B	C
a	p
b	q
a	r

$r_1 \bowtie r_2$

A	B	C
1	a	p
~~1~~	~~a~~	~~r~~
2	b	q
~~3~~	~~a~~	~~p~~
3	a	r

$attr(r_1) \cap attr(r_2) \rightarrow attr(r_1)$

$attr(r_2)$

$B \rightarrow attr(r_1)$

R

A	B	c	D	E
a	12	1	P	W
b	234	2	Q	X
C	568	1	R	Y
d	68	3	S	Z

1 R1(AB) R2(CD)

2 R1(ABC) R2(DE)

3 R1(ABC) R2(CDE)

4 R1(ABCD) R2(CDE)

5R1(ABCD)R2(DE)

6 R1(ABC) R2(BCD) R3(DE)

① R1(AB) R2(CD)

cndn.

attr(R1) ∪ attr(R2) = R

{AB}∪{CD} = ABCDE

ABCD ≠ ABCDE

② {AB}∩{CD} ≠ ϕ

ϕ F

Lossy decomposition

③ No c.k.

Which of the following are lossless decomposition (AKTU 2014-15/2016-17)

②

R1 (ABC) R2(DE)

ABC ∪ DE = R

{ABCDE} = {ABCDE}

① ✓

{ABC} ∩ {DE} ≠ ϕ

ϕ false

Lossy decomposition

R1(ABC) R2 (CDE)

① ABC ∪ CDE = R

{ABCDE} = {ABCDE}

② ABC ∩ CDE ≠ ϕ

{C}

C is not c.k as it contain duplicate Value Lossy

④

R1(ABCD) R2(CDE)

① ABCD ∪ CDE = R

{ABCDE} = {ABCDE}

True

② ABCD ∩ CDE ≠ ϕ

CD

③ C D. is C.k.

So it is Lossless decomposition

R1(ABCD) R2 (DE)

{ABCD} ∪ {DE} = R

{ABCDE} = {ABCDE}

2 ABCD ∩ DE ≠ ϕ

D

3. D is C.k.

Lossless Decomp.

⑤

R1(ABC) R2(BCD) R(DE)

R12(ABCD)

BC - com. R1 R2

R12 combine R3

ABCD ∪ DE

ABCDE ✓

ABCD ∩ DE ≠ ϕ

D → C.k.

Lossless Decomp.

R(v, w, x, y, z)

z->y

y->z

x->yv

vw->x

VW^+
$XW^+ = \{X, Y, V, W, Z\}$
YW^+
ZW^+

Lossy decomposition

1.r1(vwx) r2(xyz)

① $\{VWX\} \cup \{XYZ\} = VWXYZ$

② $\{VWX\} \cap \{XYZ\} = X \longrightarrow X^+ = \{X, Y, V, Z\}$

C.K X

2.r1(vw) r2(yz)

$VW \cup YZ = VWYZ$ ① ✗ ② ✗ Lossy decomposition

3.r1(vwx) r2(yz)

① VWXYZ ② $VWX \cap YZ = \phi$

Lossy decomposition

4.r1(vw) r2(wxyz)

$r1(VW) \cup r2(WXYZ) = VWXYZ$

$r1(VW) \cap r2(WXYZ) \neq \phi$

W

$W^+ = \{W\}$ — C.K. ✓

Lossy decomposition

AKTU PYQS

Q.1	What is Functional Dependency? Explain the procedure of calculating the Canonical Cover of a given Functional Dependency Set with suitable example.	AKTU 2020-21
Q.2	(i) Consider the relation R(a,b,c,d) with Set F={a→c,b→d}. Decompose this relation in 2 NF. (ii) Explain the Loss Less Decomposition with example	AKTU 2020-21
Q.3	Explain the multivalued dependency ? Explain with suitable example that how functional dependency can be used to show that decompositions are loss-less?	AKTU 2019-20/2018-19

AKTU

B.Tech III-Year

5th Semester

CS IT & CS Allied

GW GATEWAY CLASSES

DBMS: Database Management System

UNIT 3

Database Design and Normalization

Lecture-10

Today's Target

- ...Lossless decomposition
- 5NF
- AKTU PYQs

By PRAGYA RAJVANSHI

B.Tech, M.Tech(C.S.E.)

1. R(v,w x,y z)

z->y

y->z

x->yv

vw->x

1.r1(vwx) r2(xyz)

2.r1(vw) r2(yz)

3.r1(vwx) r2(yz)

4.r1(vw) r2(wxyz)

Lossless
& Decomposition

1
$r_1(VWX)\ r_2(XYZ)$

$r_1 \cup r_2 = R$

$(VWX) \cup (XYZ) = VWXYZ$

$VWXYZ = VWXYZ$ — 1

$att\ r_1 \cap att\ r_2 \neq \phi$

$(VWX) \cap (XYZ) \neq \phi$

X

$r_1 \to$ (VWX) X not candidate X

$X^+ = \{X, Y, V, Z\}$

r_2 (XYZ) $X^+ = \{X, Y, V, Z\}$

r_2 X is candidate key

4 $r_1(VW)\ r_2(WXYZ)$

$att\ r_1 \cup att\ r_2 = R$

$VWXYZ = VWXYZ$

$att(r_1) \cap att(r_2) \neq \phi$

W

W → Candidat ?

$r_1(WV)$

$r_2(WXYZ)$

$W^+ = \{W\}$ Losy decomposition

2. Given the following set of FDs on schema R (V,W,X,Y,Z) {Z→V, W→Y, XY→Z, V→WX}

State whether the following decomposition are loss-less-join decompositions or not. (i) (i) R1=(V,W,X) , R2=(V,Y,Z)

(ii) R1=(V,W,X), R2=(X,Y,Z) .

$Z \to V$
$W \to Y$
$XY \to Z$
$V \to WX$

$R_1 = (V, W, X)$ $R_2 = (V, Y, Z)$

1 $att(R_1) \cup att(R_2) = R$
$(VWX) \cup (VYZ) = VWXYZ$
$VWXYZ$

2 $att(R_1) \cap att(R_2) \neq \phi$
$(VWX) \cap (VYZ) = V$

$R_1(V, W, X)$
$R_2(V, Y, Z)$ $V^+ = \{V, W, X, Y, Z\}$

Lossless decomposition

$R_1(VWX)$ $R_2(XYZ)$

1 $R = VWXYZ$ — 1

$att(R_1) \cap att(R_2) \neq \phi$

X — 2

$X^+ = \{X\}$ $R_1 = V, W, X$
$R_2 = XYZ$

Lossy decomposition.

3. Consider a relation schema R (A , B , C , D) with the following functional dependencies-

$A \rightarrow B$

$B \rightarrow C$

$C \rightarrow D$

$D \rightarrow B$

Determine whether the decomposition of R into R_1 (A , B) , R_2 (B , C) and R_3 (B , D) is lossless or lossy.

R(ABCD)

R'(ABC) R_3 (B,D)

R_1(A,B) R_2(B,C)

R'(A B C)

R_1 (A,B) R_2(B,C)

att(R_1) ∪ att(R_2) = R'

(A,B) ∪ (B,C) = ABC

ABC

att(R_1) ∩ att(R_2) ≠ ϕ

(AB) ∩ (B,C)

B

att(R_1) ∩ att(R) → att(R_1) / att(R_2)

R_1(A,B)

R_2(B,C)

B^+ = {B, C, D}

R_2 B C.K

Lossless decomposition

R(ABCD)

R'(ABC) R_3(B,D)

① att(R') ∪ att(R_3) = R

ABC ∪ B D = R

ABCD

② att(R') ∩ att(R_3) ≠ ϕ

(ABC) ∩ (B D)

B

att(R') ∩ att(R_3) → att(R') / att(R_3)

R'(ABC)

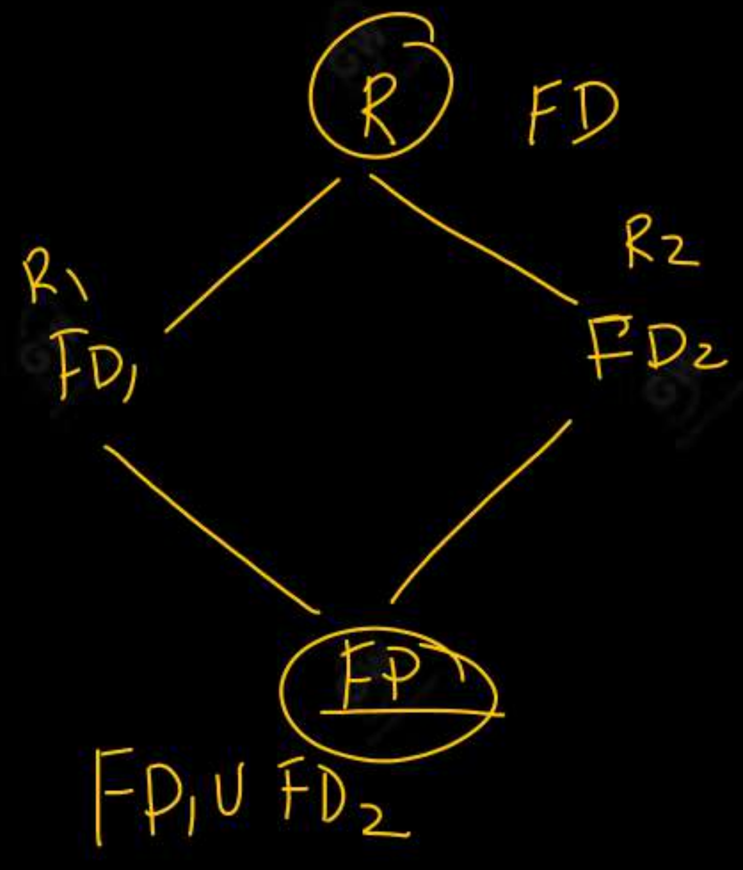
R_3(BD)

B^+ = {B, C, D}

B.C.K R_3

Overall Decomposition is lossless

Dependency preserving decomposition



1. R(A,B,C)

A->B

B->C

C->A

R1(A,B) , R2(B,C) → Dependency preserving decomposition

Whether this decomposition is dependency preserving or not

trivial dependency

$\begin{cases} A \to A \\ B \to B \end{cases}$ ←

R1(A,B)	R2(B,C)
$A^+ = \{A, B, C\}$	$B^+ = \{B, C, A\}$
$B^+ = \{B, C, A\}$	$C^+ = \{C, A, B\}$
$\left.\begin{matrix} A \to B \\ B \to A \end{matrix}\right\} F_1$	$\left.\begin{matrix} B \to C \\ C \to B \end{matrix}\right\} F_2$

$F_1 \cup F_2$ $\quad C \to B\,(F_2)$, $B \to A\,(F_1)$ → $C \to A$

R(A B C D)

AB->CD

D->A

$R_1(A, D)$ $R_2(BCD)$

Whether this decomposition is dependency preserving or not.

$R_1(A, D)$

$A^+ = \{A\}$

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

$D \to A$ $\{F_1$

$R_2(BCD)$

$B^+ = \{B\}$ $B \to B$ ✗

$C^+ = \{C\}$ $C \to C$ ✗

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

$BC = \{B, C\}$ $BC \to BC$ ✗

$BD = \{B, D, A\}$ $BD \to A$ ✗

$CD = \{C, D, A\}$ $CD \to A$ ✗

empty $\{F_2$

$F_1 \cup F_2$

$AB \to CD$ (lost)

It is not a Dependency preserving decomposition

What is transitive dependencies? what are problem arise when transitive dependences present in the database.

A *transitive dependency* occurs in a database when a non-prime attribute (not part of the candidate key) depends indirectly on a candidate key through another non-prime attribute.

For example, in a relation R(A,B,C)

R(ABC)

A->B

B->C — transitive dependency

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

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

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

$R_2(\underline{B}C)$ $B \to C$

Student

Student_ID	Student_Name	Department_ID	Department_Name
1	Alice	D01	Computer science
2	Bob	D02	Maths
3	Charlie	D01	Computer science

functional dependecies

Student_ID → Department_ID

Department_ID → Department_Name

Student_id → Student_name

$(\text{Student_id})^+ = \{$Departmend_id, Student_id, Department_name, Student_name$\}$

C.k

Problems with Transitive Dependency:

Data Redundancy: The Department_Name is repeated for every student in the same department.

Update Anomaly: If "Computer Science" changes to "CS", all rows for D01 must be updated.

Insertion Anomaly: Cannot add a new department without adding a student.

Deletion Anomaly: If the last student in a department is deleted, department information is also lost.

R1

Student_id	Student_Name	Department_id
1	Alice	D01
2	Bob	D02
3	Charlie	D02

Student_id → Student_name

Student_id → Department_id

R2

Department_id	Department_Name
D01	Computer Science
D02	Maths

Department_id → Department_name

Join dependency

- A join dependency is a constraint in a database that specifies how a table can be split into multiple smaller tables and then joined back without any loss of information.
- A relation has a join dependency if it can be decomposed into two or more sub-relations R1,R2,…,Rn, such that the natural join of these sub-relations exactly recreates the original relation R.
- A join dependency that is not implied by candidate keys leads to a violation of 5th Normal Form (5NF).

X

Company	Product	Agent
C1	TV	Aman
C1	AC	Aman
C2	Refrigerator	Mohan
C2	TV	Mohit

Table: R1

Company	Product
C1	TV
C1	AC
C2	Refrigerator
C2	TV

Table: R2

Product	Agent
TV	Aman
AC	Aman
Refrigerator	Mohan
TV	Mohit

$R_1 \bowtie R_2$

Here, we can see that we got two additional tuples after performing join i.e. (C1, TV, Mohan) & (C2, TV, Aman) these tuples are known as Spurious Tuple, which is not the property of Join Dependency. Therefore, we will create another relation R3 and perform its natural join with (R1 ⋈ R2)

AR2

Company	Product	Agent
C1	TV	Aman
~~C1~~	~~TV~~	~~Mohan~~
C1	AC	Aman
C2	Refrigerator	Mohan
~~C2~~	~~TV~~	~~Aman~~
C2	TV	Mohit

Table: R3

Company	Agent
C1	Aman
C2	Mohan
C2	Mohit

$R = R_1 \bowtie R_2 \bowtie R_3$

Now on doing natural join of (R1 ⋈ R2) ⋈ R3, we get

Company	Product	Agent
C1	TV	Aman
C1	AC	Aman
C2	Refrigerator	Mohan
C2	TV	Mohit

5 Normal Form

- Fifth Normal Form (5NF), also known as Projection-Join Normal Form (PJNF), is a level of database normalization aimed at eliminating redundancy and anomalies due to join dependencies.
- A relation is in 5NF if it is in 4NF and every non-trivial join dependency in the relation is a consequence of the candidate keys.

  X Join dependencies
  4NF

- In simpler terms, 5NF ensures that a table can be decomposed into smaller tables without losing any information and without introducing any redundancy. All decompositions should be lossless and should only occur when the decomposition is based on candidate keys and join dependencies.

(c) SUPPLY

Sname	Part_name	Proj_name
Smith	Bolt	ProjX
Smith	Nut	ProjY
Adamsky	Bolt	ProjY
Walton	Nut	ProjZ
Adamsky	Nail	ProjX
Adamsky	Bolt	ProjX
Smith	Bolt	ProjY

The relation SUPPLY with no MVDs is in 4NF but not in 5NF if it has the JD(R1, R2, R3)

(d) R_1

Sname	Part_name
Smith	Bolt
Smith	Nut
Adamsky	Bolt
Walton	Nut
Adamsky	Nail

R_2

Sname	Proj_name
Smith	ProjX
Smith	ProjY
Adamsky	ProjY
Walton	ProjZ
Adamsky	ProjX

R_3

Part_name	Proj_name
Bolt	ProjX
Nut	ProjY
Bolt	ProjY
Nut	ProjZ
Nail	ProjX

Decomposing the relation SUPPLY into the 5NF relations R1, R2, R3

AKTU PYQS

Q.1	Given the following set of FDs on schema R (V,W,X,Y,Z) {Z→V, W→Y, XY→Z, V→WX}State whether the following decomposition are loss-less-join decompositions or not. (i) R1=(V,W,X) , R2=(V,Y,Z) (ii) R1=(V,W,X), R2=(X,Y,Z) .	AKTU 2022-23
Q.2	What is transitive dependencies? what are problem arise when transitive dependences present in the database	AKTU 2021-22

AKTU PYQS

Q Normalisation Advantages

Q.3	(i) What is highest normal form of the Relation R(W,X,Y,Z) with the set F= { WY → XZ, X → Y } (ii) Consider a relation R(A,B,C,D,E) with set F= { A→CD, C→B, B→AE} What are the prime attributes of this Relation and Decompose the given relation in 3NF.	AKTU 2020-21
Q.4	**What is transitive dependencies? what are problem arise when transitive dependences present in the database**	AKTU 2021-22

Thank you