Functional Dependencies and Single Valued Normalization (Up to BCNF)

Harsh Srivastava[1], Jyotiraditya Tripathi[2] , Preeti Bala[3]
[1 & 2] M.Tech. Student, Centre for Computer Sci. & Tech. Central University of Punjab, Bathinda, India
[3] Visiting faculty IMT-CDL Ghaziabad, India
Email - [1]harshmagic20@gmail.com [2]shandilyajyotiraditya@gmail.com [3]preetripathi38@gmail.com

Abstract: *This paper represents the core concept of functional dependencies and normalization approach. Removing the redundancy from database is the main aim of normalization because it arises many other problems in database like insertion anomaly, updation anomaly and it also increases the volume of data.*

Key Words: *Keys, normalization, Functional dependencies and redundancy.*

1. INTRODUCTION:

In relational database theory, a functional dependency is a constraint between two sets of attributes in a relation from a database. In other words, functional dependency is a constraint that describes the relationship between attributes in a relation. Normalization is process which is used to reduce the redundancy from database. There is no any approach that can remove redundancy from database completely but it can be reduced.

2. NORMALIZATION:

It is a process for eliminating or reducing redundancy.
Redundancy occurs if two or more independent relations stored in single relation.
Student- Course- Enroll

Sid	Sname	Age	Cid	Cname	Duration	Fee
S1	A	20	C1	DB	$\cancel{15}$ 16	-
S1	A	20	C2	DS	14	-
S1	A	20	C3	DM	18	-
S2	**A**	**20**	**C4**	**C**	**10**	**-**
S3	B	22	C1	DB	$\cancel{15}$ 16	-
S4	B	23	C1	DB	15	-
XX	XX	XX	C5	CN	15	-

[Dummy data]

Problems Because of Redundancy (Anomaly):

Updation Anomaly: If failed to update any redundant copy inconsistency occurs.

Deletion Anomaly: Because of deletion of some data forced to delete other independent data.

Insertion Anomaly: Because of insertion of some data forced to insert other dummy data.

Decomposition of relation: Splitting relation into two or more sub relations to reduce redundancy.

Sid	Sname	Age
S1	A	20
S2	A	20
S3	B	22
S4	B	23

Sid	Cid	Fee
S1	C1	-
S1	C2	-
S1	C3	-
S2	C4	-
S3	C1	-
S4	C1	-

Cid	Cname	Duration
C1	DB	15
C2	-	-
C3	-	-
C4	-	-

Functional Dependency [FD]

Sid	Sname	Cid
S1	A	C1
S1	A	C2
S2	B	C1
S3	B	C2
S2	B	C3

"Sid $\rightarrow$ Sname"
X, Y some set attributes over relation R. X $\rightarrow$ Y implied in R only if t1, t2 any tuples such that if t1x = t2x then t1y= t2y.

X	Y	Z
X1	Y1	Z1
X1	Y1	Z2
X1	Y1	Z3
X2	Y2	Z4
X3	Y2	Z5

Suppose Z is super key.
X → Y, Implied
Y → X, Not implied

$$\left.\begin{array}{l} Z \rightarrow X \\ Z \rightarrow Y \end{array}\right\}\ \text{Implied}$$

Trivial Functional Dependency:

X, Y are some set of attributes over R if X ≥ Y
Then X → Y
Hence this process is called as Trivial Dependency.
Sid → Sid
Sid Sname → Sid Sname

$$\begin{tikzcd}
\text{Sid Sname} \arrow[r] \arrow[d] & \text{Sname} \arrow[d] \\
X & Y \\
\text{Sname} \arrow[r] \arrow[u] & \text{Sname} \arrow[u]
\end{tikzcd}$$

Every possible trivial functional dependency implied in relation.

Non-Trivial Functional Dependency:

X, Y some non-empty set of attributes over R.

X → Y non trivial FD if X∩Y= ϕ
(No common attribute in X, Y sets)
Sid → Sname

$$\begin{tikzcd}
\text{Sid} \arrow[d] & \text{Cid} \arrow[r] \arrow[d] & \text{fee} \\
X \cap & Y = \phi &
\end{tikzcd}$$

Sname→Sid
(Not implied)
Some relations may not have any non – trivial functional dependency.

Semi Non-Trivial Functional Dependency:

Combination of trivial and non-trivial.

$$\begin{tikzcd}
 & \text{Sid} \rightarrow \text{Sid Sname} \quad \text{T/F} \arrow[dl] \arrow[dr] & \\
\text{Sid} \rightarrow \text{Sid [trivial] T} & & \text{Sid} \rightarrow \text{Sname [Non-trivial] T/F}
\end{tikzcd}$$

If no two values of X is same then X→ Y is implied.
Note: If there is no any non-trivial functional dependency in table, then all attribute forms only one candidate key.

Armstrong Rules over Functional Dependency:

1. **Reflexivity:**

   X →X "Trivial Functional Dependency"

2. **Transitivity:**

   If X → Y
   And Y → Z
   Then X → Z [Always true]

3. **Augmentation:**

   If X → Y then X Z → Y Z

4. **Split Rule:**

   If X → Y Z, then
   X → Y
   X → Z

5. **Union Rule:**

   If X → Y and X → Z, then
   X → Y Z

Attribute Closure:

Set of attributes determine by X.
F= {AB → C, B → D, D →E, AE → F}

$$\begin{tikzcd}
(AB+) = \{A, B, C, D, E, F\} & \\
AB \rightarrow ABCDEF \arrow[r, dash] & {} \arrow[lu]
\end{tikzcd}$$

If B → D then AB → AD
If AB → AD then AB → A
AB → D

(B+) = {B, D, E}

Note: Attributes belong to candidate key is prime attributes and remaining are non-prime.
If X → Y non-trivial functional dependency in R with Y is prime attribute, then R has at least two Candidate keys.

Membership Test:

F = {-------}E X → Y
F = {AB → C, BC → D} K AB → D

1. AB → C then AB → BC

   $\uparrow$ $\uparrow$ [Augmentation rule]
   B B

   AB → BC, BC → D, then
   AB → D

2. (AB)+ = ABCD
AB → D implied in relation function
X → Y implied in function.

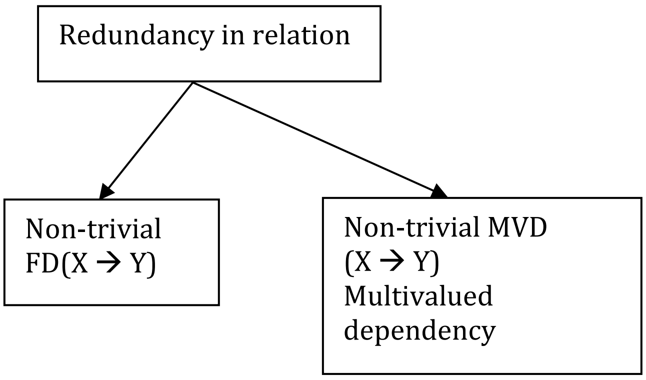
Equality between Functional Dependency sets:

F and G are functional dependencies, F and G is equal if,

(i) **F covers G:** Every functional dependency of G implied in F.
(ii) **G covers F:** Every functional dependency of F should implied in G.

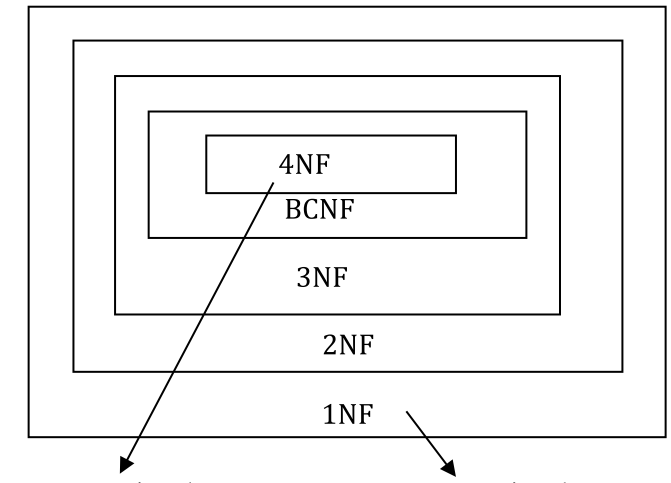
I. Normal Forms

Used to eliminate or reduce redundancy.



1NF Based on functional dependency [Single valued dependency]
2NF
3NF
BCNF → No redundancy due to FD's → Not free from redundancy due to MVD's
4NF] → Based on MVD's
[Eliminates redundancy even because of MVD's]

Expressive Power Set of Relations Accepted by NF:



[More restricted] [Less restricted]

First Normal Form (1NF):

Relational schema R is in 1NF if every attribute of R is single valued (Atomic).
[No multi-valued attribute in R]

Sid	**Sname**	**Cn**
S1	A	C1/C2
S2	B	C2/C3
S3	B	C2

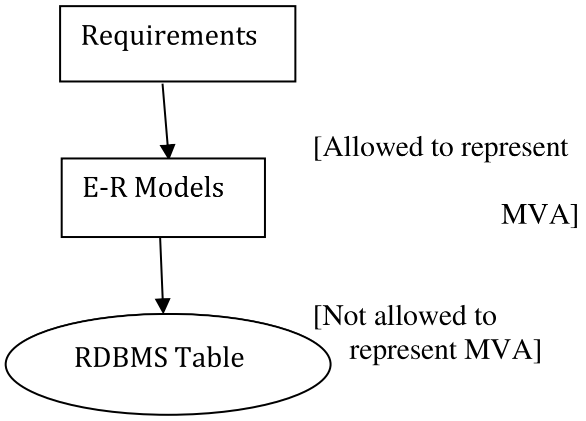
Multi valued

[Not in NF]

Sid	Cname	Sname
S1	C1	A
S1	C2	A
S2	C2	B
S2	C3	B
S3	C2	B

R (Sid Sname Cname)
F = {Sid → Sname}
Candidate key {Sid Cname}

R(ABCDE)
{A→B, B→C, C→A}
Candidate keys= {ADE, BDE, CDE}
{In 1 NF}



[Default 1NF]

Non-trivial functional dependency X → Y with X is not super key in relation R then X → Y forms redundancy.

X → Y
Not Super key
R Z

[Redundancy]

X	Y
X1	Y1
X1	Y1
X1	Y1

X2	Y3
X2	Y3

X -> Y non-trivial functional dependency with X is super-key then,

X → Y doesn't form redundancy.

R(Sid Sname Cname)

Sid → Sname

↑

Not Super Key

Not Super Key:

1. Proper subset of candidate key, or
2. Non-prime attribute

   R(ABCDE)

   Candidate key {AB, BC}

{AB → C, C → A, C → D, D → E

Not Super key

[Forms Redundancy]

Functional Dependency (X→Y) forms redundancy:

(i) Proper Subset of Candidate key(X) → Non-prime attribute (Y) [Partial Dependency]

(ii) Non-prime attribute(X) → Non-prime attribute (Y)

(iii) Proper subset of candidate key(X)(Prime) → Proper subset of other candidate key (Y) (Prime)

(iv) Not possible case → Non prime attribute(X) → Proper subset of candidate key(Y)

	1NF	2NF	3NF	BCNF
1	Allowed	Not allowed	Not allowed	Not allowed
2	Allowed	Allowed	Not allowed	Not allowed
3	Allowed	Allowed	Allowed	Not allowed

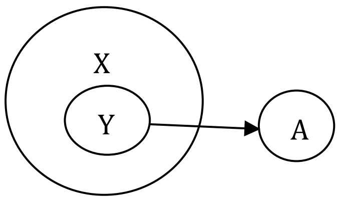
Degree of Redundancy:

1NF > 2NF > 3NF > BCNF

Second Normal Form (2NF):

Relational schema R is in 2NF if no partial dependencies in R.

Partial dependencies



X: Any candidate key

Y: Proper subset of candidate key

A: Non-prime

Y → A [Partial dependency]

[Proper subset of candidate key X]

Third Normal Form (3NF):

Relational schema R is in 3NF if every non-trivial functional dependency x → Y in relation R with

1. X is super key, or
2. Y is prime attribute.

X → Y

⇧ or ⇧

Super key Prime attribute

1. Proper subset of → Non-prime Candidate key [Not Applicable]
2. Non-Prime(X) → Non-prime(Y) [Not Applicable]
3. Proper subset of → Proper subset of candidate key (Y)

Boyce-Codd Normal Form (BCNF):

Relation R is in BCNF if every non-trivial functional dependency X → Y in R with X is super key. BCNF is fully free from single valued redundancy.

3. CONCLUSION:

From above explanation it is clear that the redundancy cannot be removed from database completely. For example if any relation is in BCNF then we can say that this particular relation is free from single valued dependencies but there still exists multi valued dependencies. For single valued dependencies, BCNF is the most restrictive normal form and 1nf is least restrictive.

4. REFERENCES:

1. Database System Concepts', *Sixth Edition* Avi Silberschatz Henry F. Korth S. Sudarshan.

2. C. Batini, M. Lenzerini, and S. B. Navathe. A comparative analysis of methodologies for database schema integration. In ACM Computing Surveys, pages 323–364, 1986.

3. M. J. Cafarella, A. Halevy, D. Zhe Wang, Y. Zhang, and E. Wu. Webtables: Exploring the power of tables on the web. In Proc. of VLDB, 2008.

4. Yao, H., H.J. Hamilton and C.J. Butz, 2002.‖FD_MINE: Discovering Functional Dependencies in a Database Using Equivalences‖ 2002. IEEE International Conference on Data Mining (ICDM02), IEEE Computer Society, Maebashi City, Japan, ISBN 0-7695-1754-4,Dec. 9-12, 2002, pp. 729-732.

5. Huhtala, Y., Kärkkäinen, J., Porkka, P., and Toivonen, H., ―TANE: An Efficient Algorithm for Discovering Functional and Approximate Dependencies.‖ The Computing Journal, 42(2):100-111 (1999).