

# Third Normal Form and Boyce-Codd Normal Form

**Dr. Seema Gupta Bhol**

# Third Normal Form (3NF)

This form dictates that all **non-key** attributes of a table must be functionally dependent on a candidate key i.e. there can be no interdependencies among non-key attributes.

For a table to be in 3NF, there are two requirements

- The table should be second normal form
- No attribute is transitively dependent on the primary key

# Example 1

Consider a relation : sports(*rollno*, *game*, *feestructure*)

Rollno	Game	Feestructure
1	Basketball	500
2	Basketball	500
3	Basketball	500
4	Cricket	600
5	Cricket	600
6	Cricket	600
7	Tennis	400

FD = {rollno  $\rightarrow$  game,  
rollno  $\rightarrow$  feestructure,  
game  $\rightarrow$  feestructure}

Thus, rollno is primary key (from first two functional dependencies).

Also, there is transitive dependency: **game  $\rightarrow$  feestructure.**

- The sports table is in 1NF because there are no multivalued attributes.
- Student table is also in 2NF because all non-key attributes are fully functionally dependent on the primary key (rollno).
- But the table is not in 3NF because there is transitive dependency i.e.  
game  $\rightarrow$  feestructure.
- Feestructure has transitive/indirect dependency on rollno via game.

# Anomalies

- The above student table is also suffering from all three anomalies –
- **Insertion anomaly** – A new game can't be inserted into the table unless we get a student to play that game.
- **Deletion anomaly** – If *rollno 7* is deleted from the table we also lost the complete information regarding tennis.
- **Updation anomaly** –To change the fee structure for basketball we need to make changes in more than one place.
- So, now to convert the above student table into 3NF first we need to decompose the table .

# Third Normal Form (3NF)

- A relation that is in First and Second Normal Form and in which no non-primary-key attribute is transitively dependent on the primary key, then it is in Third Normal Form (3NF).
- A relation is in **3NF** when it is in **2NF** and there is no transitive dependency.

# Decomposition for 3NF

To overcome these anomalies, the student table should be divided into smaller tables.

If  $X \rightarrow Y$  is transitive dependency then divide R into  $R1(X^+)$  and  $R2(R - Y^+)$ .

Game  $\rightarrow$  feestructure is a transitive dependency  
[since neither game is a key nor fee is a key attribute]

$R1 = \text{game}^+ = (\text{game}, \text{feestructure})$

$R2 = (\text{student-feestructure}^+) = (\text{rollno}, \text{game})$

So divide the student table into

$R1(\underline{\text{game}}, \text{feestructure})$  and

$R2(\underline{\text{rollno}}, \text{game})$ .

## R1(game, feestructure)

Rollno	Game	Rollno
1	Basketball	1
2	Basketball	2
3	Basketball	3
4	Cricket	4
5	Cricket	5
6	Cricket	6
7	tennis	7

## R2 (rollno, game)

Game	Feestructure
Basketball	500
Cricket	600
Tennis	400

The above two tables are free from all three anomalies.



## Example 2

STUD_NO	STUD_NAME	STUD_STATE	STUD_COUNTRY	STUD_AGE
1	RAM	HARYANA	INDIA	20
2	RAM	PUNJAB	INDIA	19
3	SURESH	PUNJAB	INDIA	21

FD set: {STUD\_NO  $\rightarrow$  STUD\_NAME,  
STUD\_NO  $\rightarrow$  STUD\_STATE,  
STUD\_NO  $\rightarrow$  STUD\_AGE,  
STUD\_STATE  $\rightarrow$  STUD\_COUNTRY}

Candidate Key: ????  
{STUD\_NO}

For this relation :

STUD\_NO  $\rightarrow$  STUD\_STATE and

STUD\_STATE  $\rightarrow$  STUD\_COUNTRY are true.

So STUD\_COUNTRY is transitively dependent on STUD\_NO.

It violates the third normal form.

To convert it in third normal form, we will decompose the relation STUDENT as:

STUDENT (STUD\_NO, STUD\_NAME, STUD\_STATE, STUD\_AGE)

And

STATE\_COUNTRY (STATE, COUNTRY)

## Example 3 (Not in 3NF)

Scheme  $\rightarrow$  {Title, PubID, PageCount, Price }

- Key  $\rightarrow$  {Title, PubId}
  - {Title, PubId}  $\rightarrow$  {PageCount, Price}
  - {PageCount}  $\rightarrow$  {Price}
- 
- Is this relation in first normal form?
  - Yes, as all fields contains atomic values .
  - Is this relation in Second normal form?
  - Yes. Both Price and PageCount depend on a key hence 2NF.
  - Is this relation in Third normal form?
  - Transitively {Title, PubID}  $\rightarrow$  {Price} hence not in 3NF.

# 3NF - Decomposition

1. Move all items involved in transitive dependencies to a new entity.
2. Identify a primary key for the new entity.
3. Place the primary key for the new entity as a foreign key on the original entity.

## Example 3 (Convert to 3NF)

Old Scheme → { Title, PubID, PageCount, Price }

New Scheme → R1 { PageCount, Price }

R2 { Title, PubID, PageCount }

# Example 4

## Example 4 (Not in 3NF)

Scheme  $\rightarrow \{\underline{\text{Studio}}, \text{StudioCity}, \text{CityTemp}\}$

- Primary Key  $\rightarrow \{\text{Studio}\}$
- $\{\text{Studio}\} \rightarrow \{\text{StudioCity}\}$
- $\{\text{StudioCity}\} \rightarrow \{\text{CityTemp}\}$
- $\{\text{Studio}\} \rightarrow \{\text{CityTemp}\}$
- Is this relation in Second normal form?
- Both StudioCity and CityTemp depend on the entire key hence 2NF.
- Is this relation in Third normal form?
- CityTemp transitively depends on Studio hence violates 3NF .

# Example 4 - Decomposition

## Example 4 (Convert to 3NF)

Old Scheme  $\rightarrow$  {Studio, StudioCity, CityTemp}

New Scheme  $\rightarrow$  {Studio, StudioCity}

New Scheme  $\rightarrow$  {StudioCity, CityTemp}

# Example 5

## Example 5 (Not in 3NF)

Scheme  $\rightarrow$  {BuildingID, Contractor, Fee}

- Primary Key  $\rightarrow$  {BuildingID}
- {BuildingID}  $\rightarrow$  {Contractor}
- {Contractor}  $\rightarrow$  {Fee}
- {BuildingID}  $\rightarrow$  {Fee}

BuildingID	Contractor	Fee
100	Randolph	1200
150	Ingersoll	1100
200	Randolph	1200
250	Pitkin	1100
300	Randolph	1200

- Is this relation in Second normal form?
- Both Contractor and Fee depend on the entire key hence 2NF.
- Is this relation in Second normal form?
- Fee transitively depends on the BuildingID. Thus not in 3 NF.

# Example 5 - Decomposition

## Example 5 (Convert to 3NF)

Old Scheme  $\rightarrow$  {BuildingID, Contractor, Fee}

New Scheme  $\rightarrow$  {BuildingID, Contractor}

New Scheme  $\rightarrow$  {Contractor, Fee}

BuildingID	Contractor
100	Randolph
150	Ingersoll
200	Randolph
250	Pitkin
300	Randolph

Contractor	Fee
Randolph	1200
Ingersoll	1100
Pitkin	1100

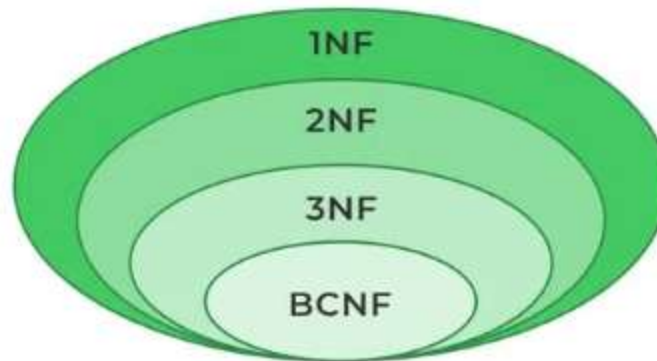


# 3rd Normal Form / Boyce-Codd Normal Form

- 3rd Normal Form: no transitive dependencies

*Transitive dependency means that a non-key attribute depends on another non-key attribute(s).*

*This definition says nothing about dependencies that involve the key.*



# 3rd Normal Form / Boyce-Codd Normal Form

**BCNF: Every determinant is a candidate key.**

*Determinant: any attribute(s) that functionally determine another attribute.*

*BCNF means that there are no “transitive” dependencies involving key or non-key attributes.*

# Example 1

Consider a relation R with attributes (student, subject, teacher).

Student	Teacher	Subject
Jhansi	P.Naresh	Database
jhansi	K.Das	C
subbu	P.Naresh	Database
subbu	R.Prasad	C

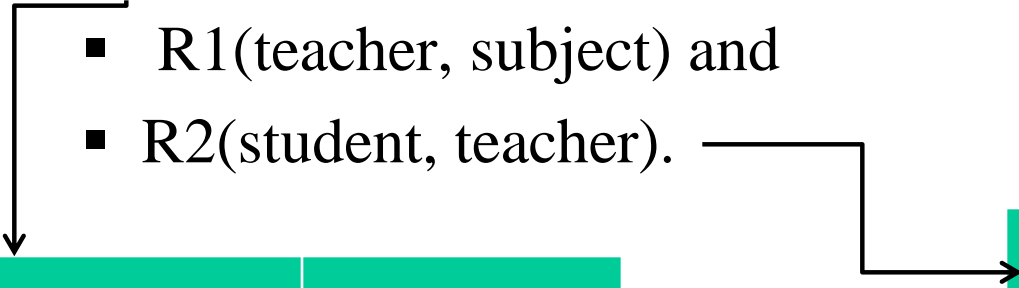
F: { (student, Teacher)  $\rightarrow$  subject, (student, subject)  $\rightarrow$  Teacher, Teacher  $\rightarrow$  subject }

**Candidate keys** are (student, teacher) and (student, subject).

- The above relation is in 3NF [since there is no transitive dependency].
- The above relation is not in BCNF, because in the FD (teacher- $\rightarrow$ subject), teacher is not a key.
- This relation suffers with anomalies –
  - For example, if we try to delete the student Subbu, we will lose the information that R. Prasad teaches C.
- These difficulties are caused by the fact the teacher is determinant but not a candidate key.

# Decomposition for BCNF

- Teacher  $\rightarrow$  subject ,violates BCNF [since teacher is not a candidate key].
- If  $X \rightarrow Y$  violates BCNF then divide R into  $R_1(X, Y)$  and  $R_2(R-Y)$ .
- So R is divided into two relations:
  - $R_1(\text{teacher, subject})$  and
  - $R_2(\text{student, teacher})$ .



Teacher	Subject
P.Naresh	database
K.DAS	C
R.Prasad	C

Student	Teacher
Jhansi	P.Naresh
Jhansi	K.Das
Subbu	P.Naresh
Subbu	R.Prasad

# Example 2

## Example 2 - Movie (Not in BCNF)

Scheme  $\rightarrow$  {MovieTitle, MovieID, PersonName, Role, Payment }

1. Key1  $\rightarrow$  {MovieTitle, PersonName}
  2. Key2  $\rightarrow$  {MovieID, PersonName}
  3. {MovieID}  $\rightarrow$  {MovieTitle}
- Both role and payment functionally depend on both candidate keys . No transitivity , thus 3NF.
  - Dependency between MovieID & MovieTitle Violates BCNF.

# BCNF - Decomposition

## Example 2 (Convert to BCNF)

Old Scheme  $\rightarrow$  {MovieTitle, MovieID, PersonName, Role, Payment }

New Scheme  $\rightarrow$  R1 { MovieID, PersonName, Role, Payment }  
 $\rightarrow$  R2 { MovieID, MovieTitle }

## Example 3

Let's assume there is a company where employees work in more than one department.

EMPLOYEE table:

EMP_ID	EMP_COUNTRY	EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
264	India	Designing	D394	283
264	India	Testing	D394	300
364	UK	Stores	D283	232
364	UK	Developing	D283	549

In the above table Functional dependencies are as follows:

$EMP\_ID \rightarrow EMP\_COUNTRY$

$EMP\_DEPT \rightarrow \{DEPT\_TYPE, EMP\_DEPT\_NO\}$

**Candidate key: {EMP-ID, EMP-DEPT}**



The table is not in BCNF because neither EMP\_DEPT nor EMP\_ID alone are keys.

To convert the given table into BCNF, we decompose it into three tables:

**EMP\_COUNTRY table:**

EMP_ID	EMP_COUNTRY
264	India
264	India

**EMP\_DEPT table:**

EMP_DEPT	DEPT_TYPE	EMP_DEPT_NO
Designing	D394	283
Testing	D394	300
Stores	D283	232
Developing	D283	549

**EMP\_DEPT\_MAPPING table:**

EMP_ID	EMP_DEPT
D394	283
D394	300
D283	232
D283	549

# Example

Let us consider the student table.

Stu_ID	Stu_Branch	Stu_Course	Branch_Number	Stu_Course_No
101	Computer Science & Engineering	DBMS	B_001	201
101	Computer Science & Engineering	Computer Networks	B_001	202
102	Electronics & Communication Engineering	VLSI Technology	B_003	401
102	Electronics & Communication Engineering	Mobile Communication	B_003	402

Functional Dependency of the above is as mentioned:

$\text{Stu\_ID} \rightarrow \text{Stu\_Branch}$

$\text{Stu\_Course} \rightarrow \{\text{Branch\_Number}, \text{Stu\_Course\_No}\}$

Candidate Keys of the above table are:  **$\{\text{Stu\_ID}, \text{Stu\_Course}\}$**

# Students Table in BCNF?

The table present above is not in BCNF, because as we can see that neither Stu\_ID nor Stu\_Course is a candidate Key.

Stu_ID	Stu_Branch
101	Computer Science & Engineering
102	Electronics & Communication Engineering

Candidate Key : Stu\_ID.

Stu\_ID to Stu\_Course\_No Table

Stu_ID	Stu_Course_No
101	201
101	202
102	401
102	402

Candidate : {Stu\_ID, Stu\_Course\_No}

Stu_Course	Branch_Number	Stu_Course_No
DBMS	B_001	201
Computer Networks	B_001	202
VLSI Technology	B_003	401
Mobile Communication	B_003	402

Candidate Key: Stu\_Course

# Decomposition – Loss of Information

1. If decomposition does not cause any loss of information it is called a **lossless** decomposition.
2. If a decomposition does not cause any dependencies to be lost it is called a **dependency-preserving** decomposition.
3. Any table scheme can be decomposed in a lossless way into a collection of smaller schemas that are in BCNF form. However the dependency preservation is not guaranteed.
4. Any table can be decomposed in a lossless way into 3<sup>rd</sup> normal form that also preserves the dependencies.
  - 3NF may be better than BCNF in some cases

# Lossless Join Decomposition

If we decompose a relation R into relations R1 and R2, Decomposition is lossy if:  $R1 \bowtie R2 \supset R$

Decomposition is lossless if  $R1 \bowtie R2 = R$

To check for lossless join decomposition using the FD set, the following conditions must hold:

1. The Union of Attributes of R1 and R2 must be equal to the attribute of R. Each attribute of R must be either in R1 or in R2.

$$\text{Att}(R1) \cup \text{Att}(R2) = \text{Att}(R)$$

2. The intersection of Attributes of R1 and R2 must not be NULL.

$$\text{Att}(R1) \cap \text{Att}(R2) \neq \Phi$$

3. The common attribute must be a key for at least one relation (R1 or R2)

$$\text{Att}(R1) \cap \text{Att}(R2) \rightarrow \text{Att}(R1) \quad \text{or}$$

$$\text{Att}(R1) \cap \text{Att}(R2) \rightarrow \text{Att}(R2)$$

# Example 1

- Consider R (A, B, C, D) with FD set{ A->BC} is decomposed into R1(ABC) and R2(AD) which is a lossless join decomposition as:
- First condition holds true as  $Att(R1) \cup Att(R2)$   
 $= (ABC) \cup (AD) = (ABCD) = Att(R)$ .
- Second condition holds true as  $Att(R1) \cap Att(R2)$   
 $= (ABC) \cap (AD) = A \neq \Phi$
- The third condition holds as  $Att(R1) \cap Att(R2) = A$  is a key of R1(ABC) because A->BC is given.

# Dependency Preserving Decomposition

If we decompose a relation  $R$  into relations  $R_1$  and  $R_2$ , All dependencies of  $R$  either must be a part of  $R_1$  or  $R_2$  or must be derivable from a combination of functional dependency of  $R_1$  and  $R_2$ .

For Example, A relation  $R(A, B, C, D)$  with FD set  $\{A \rightarrow BC\}$  is decomposed into  $R_1(ABC)$  and  $R_2(AD)$  which is dependency preserving because FD  $A \rightarrow BC$  is a part of  $R_1(ABC)$ .

# Example 1

- Consider a schema  $R(A, B, C, D)$  and functional dependencies  $A \rightarrow B$  and  $C \rightarrow D$ .
- Then the decomposition of  $R$  into  $R_1(AB)$  and  $R_2(CD)$ .
- Is it loss less dependency preserving decomposition?
- For lossless join decomposition, these three conditions must hold:
- $\text{Att}(R_1) \cup \text{Att}(R_2) = ABCD = \text{Att}(R)$
- $\text{Att}(R_1) \cap \text{Att}(R_2) = \Phi$ , which violates the condition of lossless join decomposition. **Hence the decomposition is not lossless.**
- For dependency preserving decomposition,  $A \rightarrow B$  can be ensured in  $R_1(AB)$  and  $C \rightarrow D$  can be ensured in  $R_2(CD)$ . **Hence it is dependency preserving decomposition.**



# Example2: EMPLOYEE\_DEPARTMENT

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY	DEPT_ID	DEPT_NAME
22	Denim	28	Mumbai	827	Sales
33	Alina	25	Delhi	438	Marketing
46	Stephan	30	Bangalore	869	Finance
52	Katherine	36	Mumbai	575	Production
60	Jack	40	Noida	678	Testing

The above relation is decomposed into two relations EMPLOYEE and DEPARTMENT

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY
22	Denim	28	Mumbai
33	Alina	25	Delhi
46	Stephan	30	Bangalore
52	Katherine	36	Mumbai
60	Jack	40	Noida

DEPT_ID	EMP_ID	DEPT_NAME
827	22	Sales
438	33	Marketing
869	46	Finance
575	52	Production
678	60	Testing

# Is it Lossless?

- First condition  $\text{Att}(R1) \cup \text{Att}(R2) = \text{Att}(R)$  ?
- Second condition  $\text{Att}(R1) \cap \text{Att}(R2) \neq \Phi$  ?
- The third condition  $\text{Att}(R1) \cap \text{Att}(R2) = A$  is a key of  $R1$  or  $R2$ ?

Third Normal Form (3NF) is considered adequate for normal relational database design because most of the 3NF tables are free of insertion, update, and deletion anomalies.

Moreover, 3NF always ensures functional dependency preserving and lossless.

# Comparison of BCNF and 3 NF

- BCNF has a comparatively much lower redundancy.
- In the case of 3NF, preservation occurs for all the functional dependencies.
- In the case of BCNF, there is no preservation for all the functional dependencies.
- Lossless decomposition is comparatively much easier to achieve in the case of 3NF.