Rule of Transitivity

3. **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. $A \rightarrow B$ is called as A functionally that determines B. If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

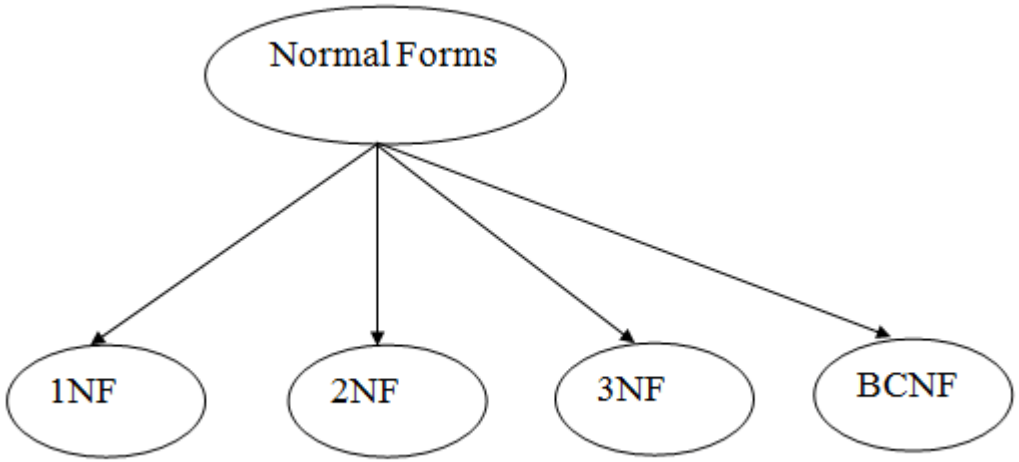
Normalization

Geetha.S

Normalization

- Normalization is the process of organizing the data in the database.
- Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.
- Normalization divides the larger table into the smaller table and links them using relationship.
- The normal form is used to reduce redundancy from the database table.

Types of Normal Forms
There are the four types of normal forms:



Normal Form	Description
1NF	A relation is in 1NF if it contains an atomic value.
2NF	A relation will be in 2NF if it is in 1NF and all non-key attributes are fully functional dependent on the primary key.
3NF	A relation will be in 3NF if it is in 2NF and no transition dependency exists.
4NF	A relation will be in 4NF if it is in Boyce Codd normal form and has no multi-valued dependency.
5NF	A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.

Anomalies

- Normalization is used to eliminate the undesirable characteristics like Insertion, Update and Deletion Anomalies.

Anamoly

Anomalies are the problem that can occur in poorly planned, Un-normalized database where all the data is stored in one table .

Three types:

1)Insertion

2)Deletion

3)Updation

- INSERT ANOMALY: This refers to the situation when it is impossible to insert certain types of data into the database.

- DELETE ANOMALY: The deletion of data leads to unintended loss of additional data, data that we had wished to preserve.

- UPDATE ANOMALY: This refers to the situation where updating the value of a column leads to database inconsistencies (i.e., different rows on the table have different values).

Insertion anomaly

An **Insert Anomaly** occurs when certain attributes cannot be inserted into the database without the presence of other attributes.

For example this is the converse of delete anomaly - we can't add a new course unless we have at least one student enrolled on the course.

StudentNum	CourseNum	Student Name	Address	Course
S21	9201	Jones	Edinburgh	Accounts
S21	9267	Jones	Edinburgh	Accounts
S24	9267	Smith	Glasgow	physics
S30	9201	Richards	Manchester	Computing
S30	9322	Richards	Manchester	Maths

Example 2:

Consider a relation emp_dept with attributes:
E#
Ename
Adderss
D
Dname
Dmgr

Let us assume that new department has been started by the organization but initially there is no employee appointed for that department,then touple for this department cannot be inserted into the table as E# will havë “NULL”which is not allowed as E# is primary key.

Delete Anomaly

- **Delete Anomaly** : A **Delete Anomaly** exists when certain attributes are lost because of the deletion of other attributes.

- Example: When a project is deleted, it will result in deleting all the employees who work on that project. Alternately, if an employee is the sole employee on a project, deleting that employee would result in deleting the corresponding project.

For example, consider what happens if Student S30 is the last student to leave the course - All information about the course is lost.

StudentNum	CourseNum	Student Name	Address	Course
S21	9201	Jones	Edinburgh	Accounts
S21	9267	Jones	Edinburgh	Accounts
S24	9267	Smith	Glasgow	physics
S30	9201	Richards	Manchester	Computing
S30	9322	Richards	Manchester	Maths

Example: Delete Anomaly

Consider a relation emp_dept with attributes:
E#
Ename
Adderss
D#
Dname
Dmgr#

Now consider there is only one employee in some department and that employee leaves the organization then touple of that employee has to be deleted from the table, but in addition to that the information of the department also will get deleted.

Update Anomaly

EXAMPLE OF AN UPDATE ANOMALY

Update Anomaly:

An **Update Anomaly** exists when one or more instances of duplicated data is updated, but not all.

the relation:

Suppose the manager of the department has changed, this requires that DMGR in all tuples corresponding to that department must be changed to reflect the new status. If we fail to update all the tuples of the given department , then two different record of employee working in the same department might show different DMGR leading to inconsistency in the database.

An **Update Anomaly** exists when one or more instances of duplicated data is updated, but not all. For example, consider Jones moving address - you need to update all instances of Jones's address.

StudentNum	CourseNum	Student Name	Address	Course
S21	9201	Jones	Edinburgh	Accounts
S21	9267	Jones	Edinburgh	Accounts
S24	9267	Smith	Glasgow	physics
S30	9201	Richards	Manchester	Computing
S30	9322	Richards	Manchester	Maths

1 NF

Decomposing the relation into First Normal Form

First Normal Form (1NF)

1. A relation will be 1NF if it contains an atomic value.
2. It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.
3. First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

(a)

(b)

DEPARTMENT

DNAME	DNUMBER	DMGRSSN	DLOCATIONS
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

(c)

DEPARTMENT

DNAME	DNUMBER	DMGRSSN	DLOCATION
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

(a) **EMP_PROJ**

SSN	ENAME	PROJS	
		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

1st Normal Form Definition

- A database is in first normal form if it satisfies the following conditions:
- Contains only atomic values
- There are no repeating groups
- An atomic value is a value that cannot be divided.

- For example, in the table shown below, the values in the [Color] column in the first row can be divided into "red" and "green", hence [TABLE_PRODUCT] is not in 1NF.

TABLE_PRODUCT

Product ID	Color	Price
1	red, green	15.99
2	yellow	23.99
3	green	17.50
4	yellow, blue	9.99
5	red	29.99

- This table is not in first normal form because the [Color] column can contain multiple values. For example, the first row includes values "red" and "green.“ To bring this table to first normal form, we split the table into two tables and now we have the resulting tables:
- Now first normal form is satisfied, as the columns on each table all hold just one value.

TABLE_PRODUCT_PRICE

Product ID	Price
1	15.99
2	23.99
3	17.50
4	9.99
5	29.99

TABLE_PRODUCT_COLOR

Product ID	Color
1	red
1	green
2	yellow
3	green
4	yellow
4	blue
5	red

next example

Example: Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

EMPLOYEE table: The decomposition of the EMPLOYEE table into 1NF has been shown below:

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385, 9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389, 8589830302	Punjab

EMP_ID	EMP_NAME	EMP_PHONE	EMP_STATE
14	John	7272826385	UP
14	John	9064738238	UP
20	Harry	8574783832	Bihar
12	Sam	7390372389	Punjab
12	Sam	8589830302	Punjab

2nd NF

Second Normal Form(2)

- A table should be in 1NF
- There must not be any partial dependency of any column on primary key.
- A table which has composite primary key, each column in the table that is not part of the primary key (non prime attribute) must depend upon the entire composite key for its existence. A non key columns should be dependent on primary key.
- If any column depends only on one part of the composite key, then the table fails **Second Normal Form**

Process From 1NF to 2NF

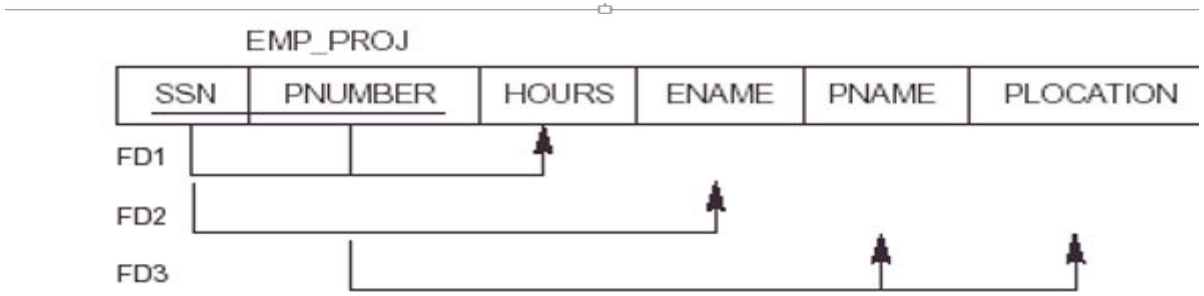
- Take each non prime attribute in turn and ask the question

"Is this attribute dependent on one part of the key?"

- If yes, remove attribute to new table with a copy of the part of the key it is dependent upon.
- If no, check against other part of the key and repeat above process
- if still no, i.e. not dependent on either part of key , keep attribute in current table.

Does that non key column describe what primary key identifies?

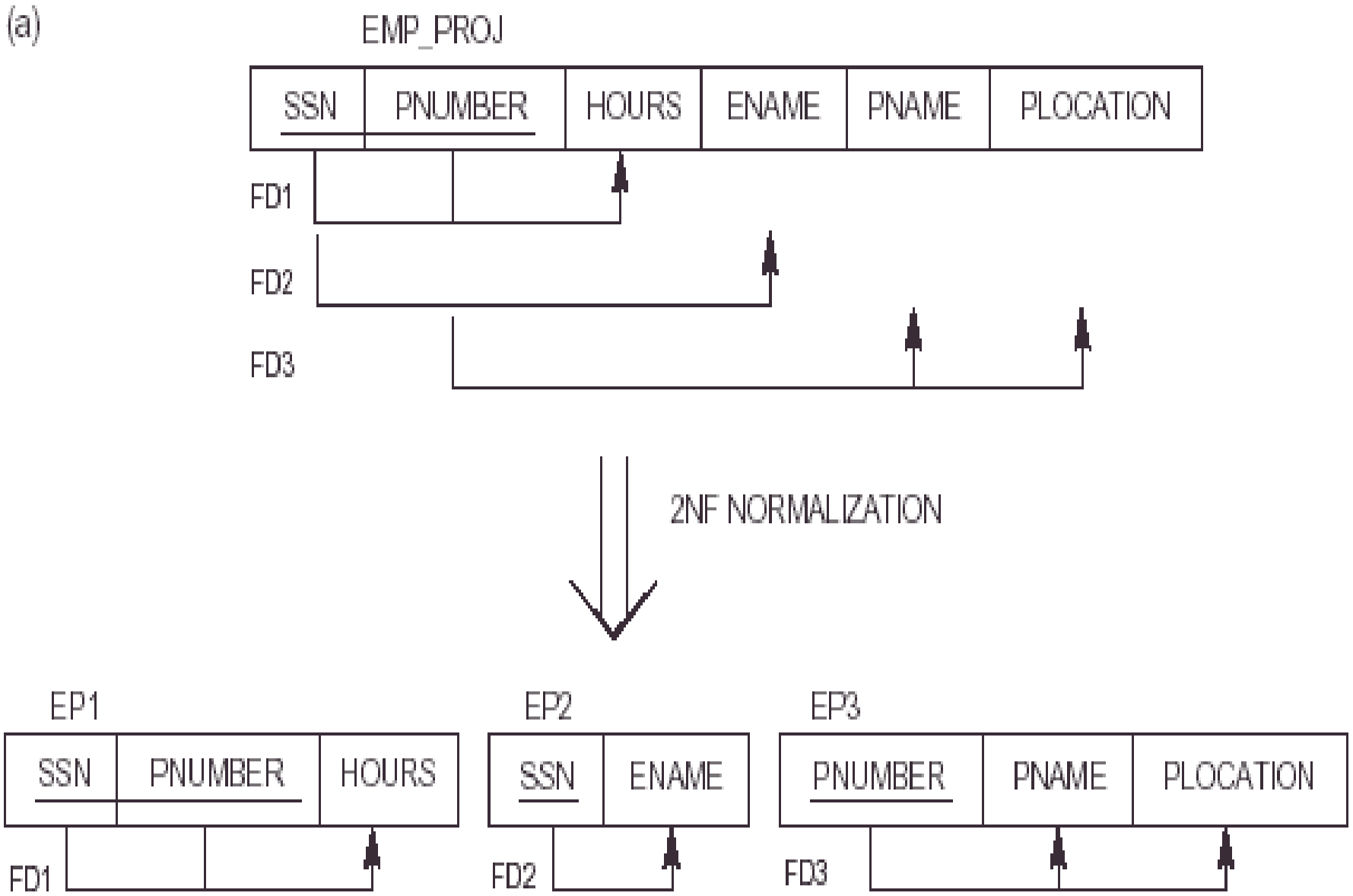
Examples
Second Normal Form



- {SSN, PNUMBER} → HOURS is a full FD since neither SSN → HOURS nor PNUMBER → HOURS hold
- {SSN, PNUMBER} → ENAME is *not* a full FD (it is called a *partial dependency*) since SSN → ENAME also holds

Summary:

- A relation schema R is in **second normal form** (**2NF**) if every non-prime attribute A in R is fully functionally dependent on the primary key


(a)
EMP_PROJ
SSN
PNUMBER
HOURS
ENAME
PNAME
PLOCATION
FD1
FD2
FD3
2NF NORMALIZATION
EP1
SSN
PNUMBER
HOURS
FD1
EP2
SSN
ENAME
FD2
EP3
PNUMBER
PNAME
PLOCATION
FD3

2nd Normal Form Definition

- A database is in second normal form if it satisfies the following conditions:
- It is in first normal form
- All non-key attributes are fully functional dependent on the primary key

- In a table, if attribute B is functionally dependent on A, but is not functionally dependent on a proper subset of A, then B is considered fully functional dependent on A.
- Hence, in a 2NF table, all non-key attributes cannot be dependent on a subset of the primary key.
- Note that if the primary key is not a composite key, all non-key attributes are always fully functional dependent on the primary key.
- A table that is in 1st normal form and contains only a single key as the primary key is automatically in 2nd normal form.

- 2nd Normal Form Example
- Consider the following example:

TABLE_PURCHASE_DETAIL

Customer ID	Store ID	Purchase Location
1	1	Los Angeles
1	3	San Francisco
2	1	Los Angeles
3	2	New York
4	3	San Francisco

- This table has a composite primary key [Customer ID, Store ID]. The non-key attribute is [Purchase Location].
- In this case, [Purchase Location] only depends on [Store ID], which is only part of the primary key.
- Therefore, this table does not satisfy second normal form.

- To bring this table to second normal form, we break the table into two tables, and now we have the following:

TABLE_PURCHASE

Customer ID	Store ID
1	1
1	3
2	1
3	2
4	3

TABLE_STORE

Store ID	Purchase Location
1	Los Angeles
2	New York
3	San Francisco

3rd NF

- 3rd Normal Form (3NF)
- Definition: A table is in 3NF if it is in 2NF and if it has no transitive dependencies.

Third Normal Form

- Definition
  - **Transitive functional dependency** – if there a set of attribute Z that are neither a primary nor candidate key and both X → Z and Y → Z holds.
- Examples:
  - SSN → DMGRSSN is a transitive FD since

  SSN → DNUMBER and DNUMBER → DMGRSSN hold
  - SSN → ENAME is *non-transitive* since there is no set of

  attributes X where SSN → X and X → ENAME

3rd Normal Form Definition

- A database is in third normal form if it satisfies the following conditions:
- It is in second normal form
- There is no transitive functional dependency
- By transitive functional dependency, we mean we have the following relationships in the table: A is functionally dependent on B, and B is functionally dependent on C. In this case, C is transitively dependent on A via B.

Third Normal Form

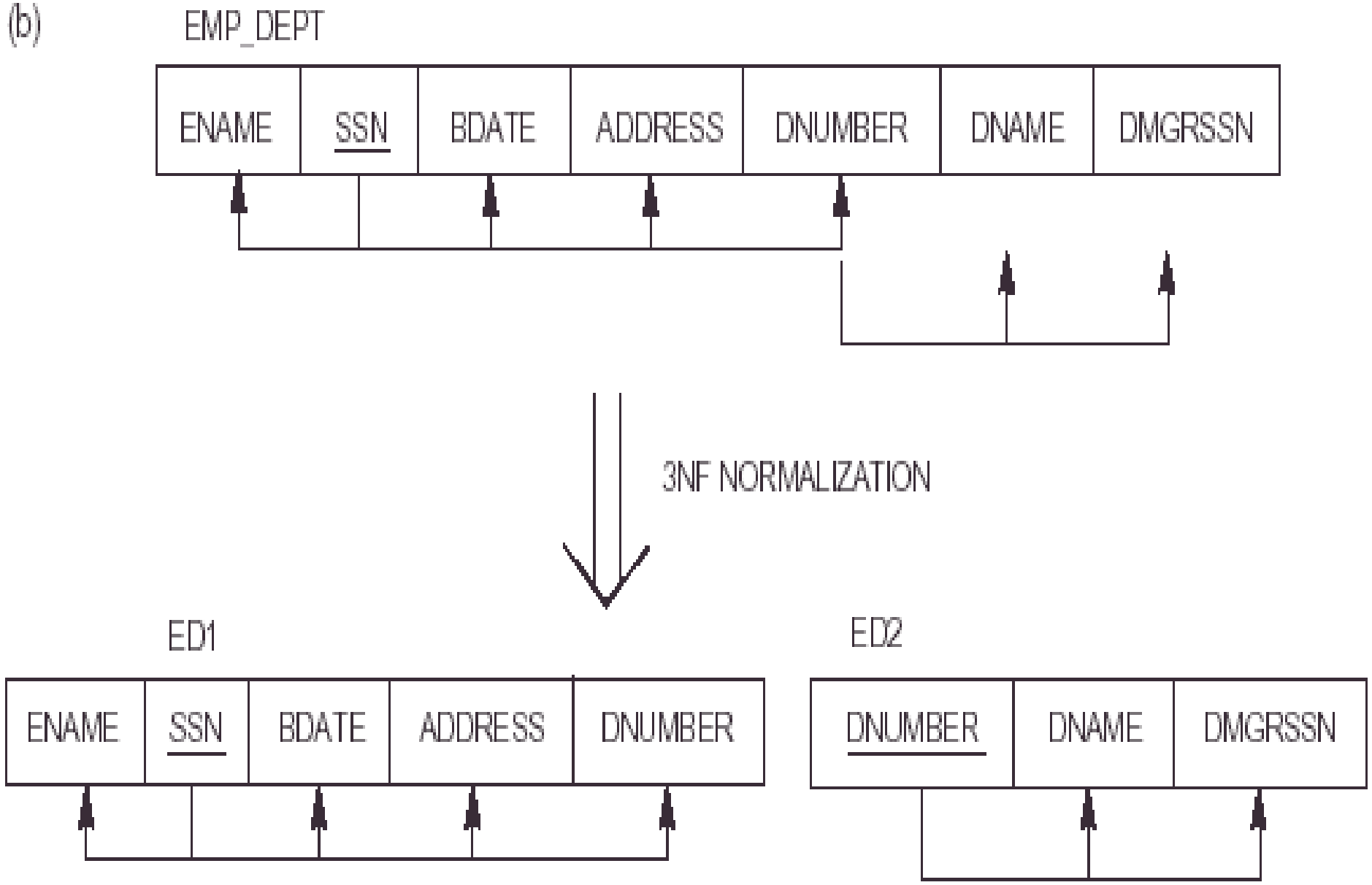
- Let’s take an example to understand it better:

Book	Author	Author_age
Game of Thrones	George R. R. Martin	66
Harry Potter	J. K. Rowling	49
Dying of the Light	George R. R. Martin	66

{Book} ->{Author} (if we know the book, we knows the author name)
{Author} does not ->{Book}
{Author} -> {Author_age}
Therefore as per the rule of **transitive dependency**: {Book} -> {Author_age} should hold

Split the above table into two

- Book Author Table1
- Author AuthorAge Table2


(b)
EMP_DEPT
ENAME
SSN
BDATE
ADDRESS
DNUMBER
DNAME
DMGRSSN
3NF NORMALIZATION
ED1
ENAME
SSN
BDATE
ADDRESS
DNUMBER
ED2
DNUMBER
DNAME
DMGRSSN

TABLE_BOOK_DETAIL

Book ID	Genre ID	Genre Type	Price
1	1	Gardening	25.99
2	2	Sports	14.99
3	1	Gardening	10.00
4	3	Travel	12.99
5	2	Sports	17.99

In the table able, [Book ID] determines [Genre ID], and [Genre ID] determines [Genre Type].
Therefore, [Book ID] determines [Genre Type] via [Genre ID] and we have transitive functional dependency, and this structure does not satisfy third normal form.

To bring this table to third normal form, we split the table into two as follows:

TABLE_BOOK

Book ID	Genre ID	Price
1	1	25.99
2	2	14.99
3	1	10.00
4	3	12.99
5	2	17.99

TABLE_GENRE

Genre ID	Genre Type
1	Gardening
2	Sports
3	Travel

Now all non-key attributes are fully functional dependent only on the primary key. In [TABLE_BOOK], both [Genre ID] and [Price] are only dependent on [Book ID]. In [TABLE_GENRE], [Genre Type] is only dependent on [Genre ID].