19CSE202 - Database Management Systems
Normalization - 1NF, 2NF

Ms. Vidhya. S
Assistant Professor(Sr.Gr.)
Department of Computer Science and Engineering
Amrita School of Engineering
Amrita Vishwa Vidyapeetham
Coimbatore

Normalization

- Formal process of decomposing relations with anomalies to produce smaller, well- structured and stable relations
- Primarily a tool to validate and improve a logical design so that it satisfies certain constraints that *avoid unnecessary duplication of data.*
- Normalization deals with "*re-organising*" a relational database by, generally, breaking up tables (relations) to remove various anomalies.
- A normal form applies to a table/relation, not to the database

First Normal Form (1NF)

- A relation R is said to be in first normal form if the domains of all the attributes of R are atomic.

- ⋆ There's no top-to-bottom ordering to the rows.
- ⋆ There's no left-to-right ordering to the columns.
- ⋆ There are no duplicate rows.
- ⋆ Every row-and-column intersection contains exactly one value from the applicable domain (and nothing else).
- ⋆ All columns are regular [i.e. rows have no hidden components such as row IDs, object IDs, or hidden timestamps].

Atomicity of Domain

- Ambiguity
  - Most values can be decomposed by functions supported by DBMS
- Example:
  - Character strings can be decomposed into substrings
  - ISBN contains language and publisher identifiers
  - Fixed point floating number can be decomposed into integral and fractional parts
- The notion is **not** absolute
- A value may be atomic in some purpose and nonatomic in some other

Atomicity of Domain

Atomicity specific to application

- First name, last name etc are atomic
  - ⭑ We won't ask to break them down
  - ⭑ Full name may not be atomic if the application requires first name or last name separately for any purpose
- Phone number is atomic
  - Set of phone numbers is not
- IDs are usually atomic
- Address is not atomic if parts (such as zip code) are required separately
- Non-atomic values increase redundancy of data

1 NF Example

- Small fragment of a database dealing with a university.
- Consider students first names as a primary key (i.e., Whenever we see particular first name more than once such as Fang or Allan, this will always refer to the same person: there is only one Fang in the university)
- It has the following columns
  - S, which is a Student
  - B, which is the Birth Year of the Student
  - C, which is a Course that the student took
  - T, which is the Teacher who taught the Course the Student took
  - F, which is the Fee that the Student paid the Teacher for taking the course

1NF Example

	S	B	C	T	F
	Fang	1990	DB	Zvi	1
			OS	Allan	2
	John	1980	OS	Allan	2
			PL	Marsha	4
	Mary	1990	PL	Vijay	1

Student Fang is registered for two courses

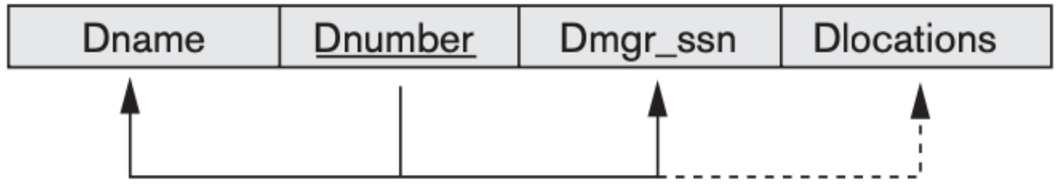
R	S	B	C	T	F
	Fang	1990	DB	Zvi	1
	John	1980	OS	Allan	2
	Mary	1990	PL	Vijay	1
	Fang	1990	OS	Allan	2
	John	1980	PL	Marsha	4

We must use two different rows for storing

1NF Example

(a)

DEPARTMENT



(b)

DEPARTMENT

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

(c)

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

First Normal Form (1NF)

- Techniques to achieve first normal form
  - ⋆ Remove attribute and place in separate relation
  - ⋆ Use several atomic attributes.
- Does not allow nested relations.
  - ⋆ Each tuple can have a relation within it.
- ***<u>To change to 1NF</u>***<u>:</u>
  - ⋆ Remove nested relation attributes into a new relation
  - ⋆ Propagate the primary key into it
  - ⋆ Unnest relation into a set of 1NF relations

1NF

(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)

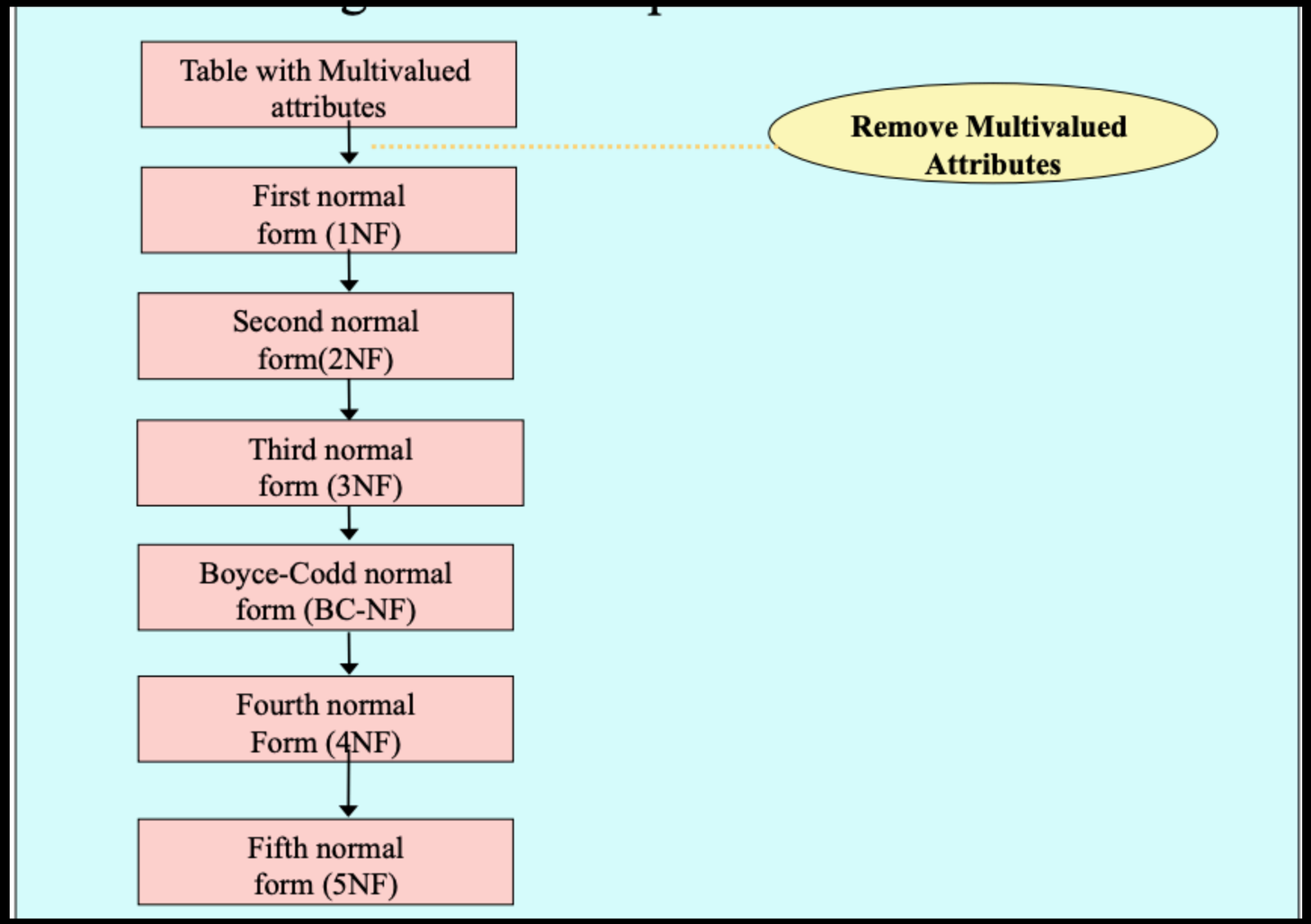
EMP_PROJ1

Ssn	Ename

EMP_PROJ2

Ssn	Pnumber	Hours

Steps in Normalization



1NF

- Only atomic attributes (simple, single-value)
- All the columns have unique names.
- A primary key has been identified

Multivalued attribute

Shirt_Info		
Shirt_ID	Design	Size
100	Mickey Mouse	XXL,XL,M
101	Flowers	M
102	Good Vibes	M,S
103	Triangles	XL
Primary Key: Shirt_ID		

Design_Info	
Shirt_ID	Design
100	Mickey Mouse
101	Flowers
102	Good Vibes
103	Triangles
Primary Key: Shirt_ID	

Size_Info	
Shirt_ID	Size
100	XXL
100	XL
100	M
101	M
102	M
102	S
103	XL
Primary Key: Shirt_ID, Size	

Second Normal Form (2NF)

- A table is supposed to be in second normal form if,
  - ⋆ It is in the 1st normal form.
  - ⋆ It does not have any partial dependency
- Eg: In `Customer_Info` table, `Store_Name` depends on `Store_ID` and not on `Cust_ID`. This is a partial dependency. Hence, `Customer_Info` is not in second normal form (though it satisfies 1NF). It can be decomposed into `Customer_Data` and `Store_Data`.

Customer_Info			
Cust_ID	Store_ID	Customer_Name	Store_Name
568	74896	Liam	JPM_1
574	74896	Charles	JPM_1
568	25614	Liam	JPM_2
235	25614	Cyrus	JPM_2
Primary Key: Cust_ID, Store_ID			

Second Normal Form (2NF)

Store_ name depends on Store_ID

Partial Dependency

Customer_Info			
Cust_ID	Store_ID	Customer_Name	Store_Name
568	74896	Liam	JPM_1
574	74896	Charles	JPM_1
568	25614	Liam	JPM_2
235	25614	Cyrus	JPM_2
Primary Key: Cust_ID, Store_ID			

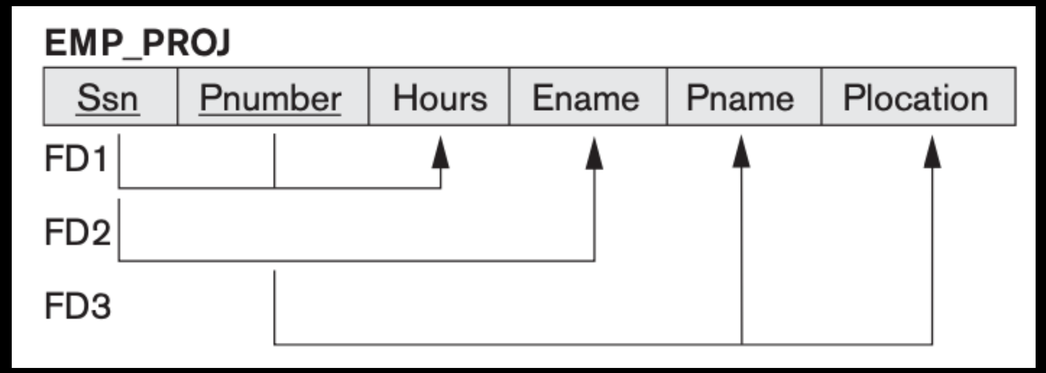
Customer_Data	
Cust_ID	Customer Name
568	Liam
574	Charles
145	Marcus
235	Cyrus
Primary Key: Cust_ID	

Store_Data	
Store_ID	Store_Name
74896	JPM_1
25614	JPM_2
Primary Key: Store_ID	

Full vs Partial Functional Dependency

- ***Full Functional Dependency:*** A functional dependency $X \rightarrow Y$ is a **full functional dependency** if removal of any attribute A from X means that the dependency does not hold any more; that is, for any attribute $A \varepsilon X$, $(X - \{A\})$ does ***not*** functionally determine Y.

- ***Partial Functional Dependency:*** A functional dependency $X \rightarrow Y$ is a **partial dependency** if some attribute $A \varepsilon X$ can be removed from X and the dependency still holds; that is, for some $A \varepsilon X$, $(X - \{A\}) \rightarrow Y$.



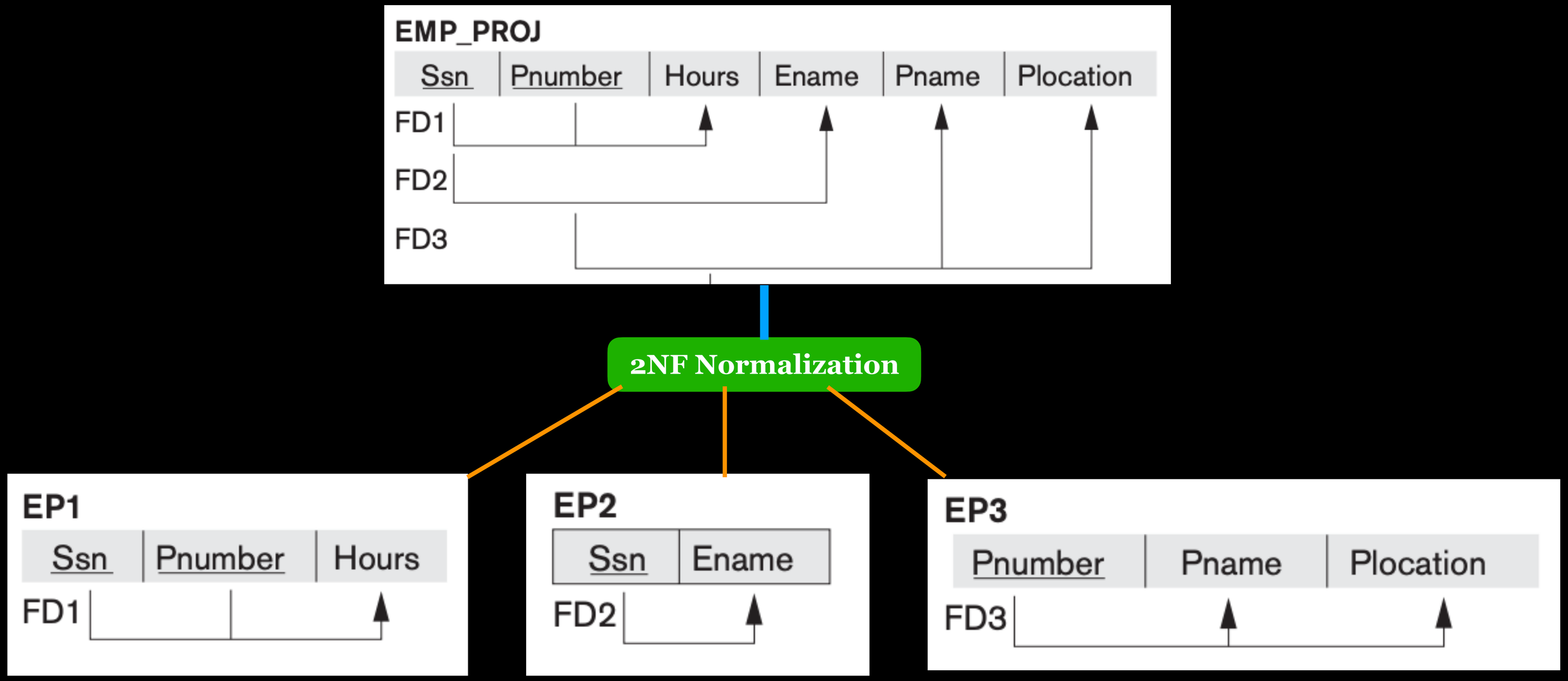
{Ssn, Pnumber} → Hours is a full dependency
(neither Ssn → Hours nor Pnumber → Hours holds)

{Ssn, Pnumber} → Ename is partial dependency
because Ssn → Ename holds.

2NF

- The test for 2NF involves testing for functional dependencies whose left-hand side attributes are part of the primary key. If the primary key contains a single attribute, the test need not be applied at all.

2NF Normalization



2NF

STUD_INSTRUCTOR

Location	Branch	Department	Batch	Section	Roll	Name	Class Advisor	HOD

DEPT_HEAD

Location	Branch	Department	HOD

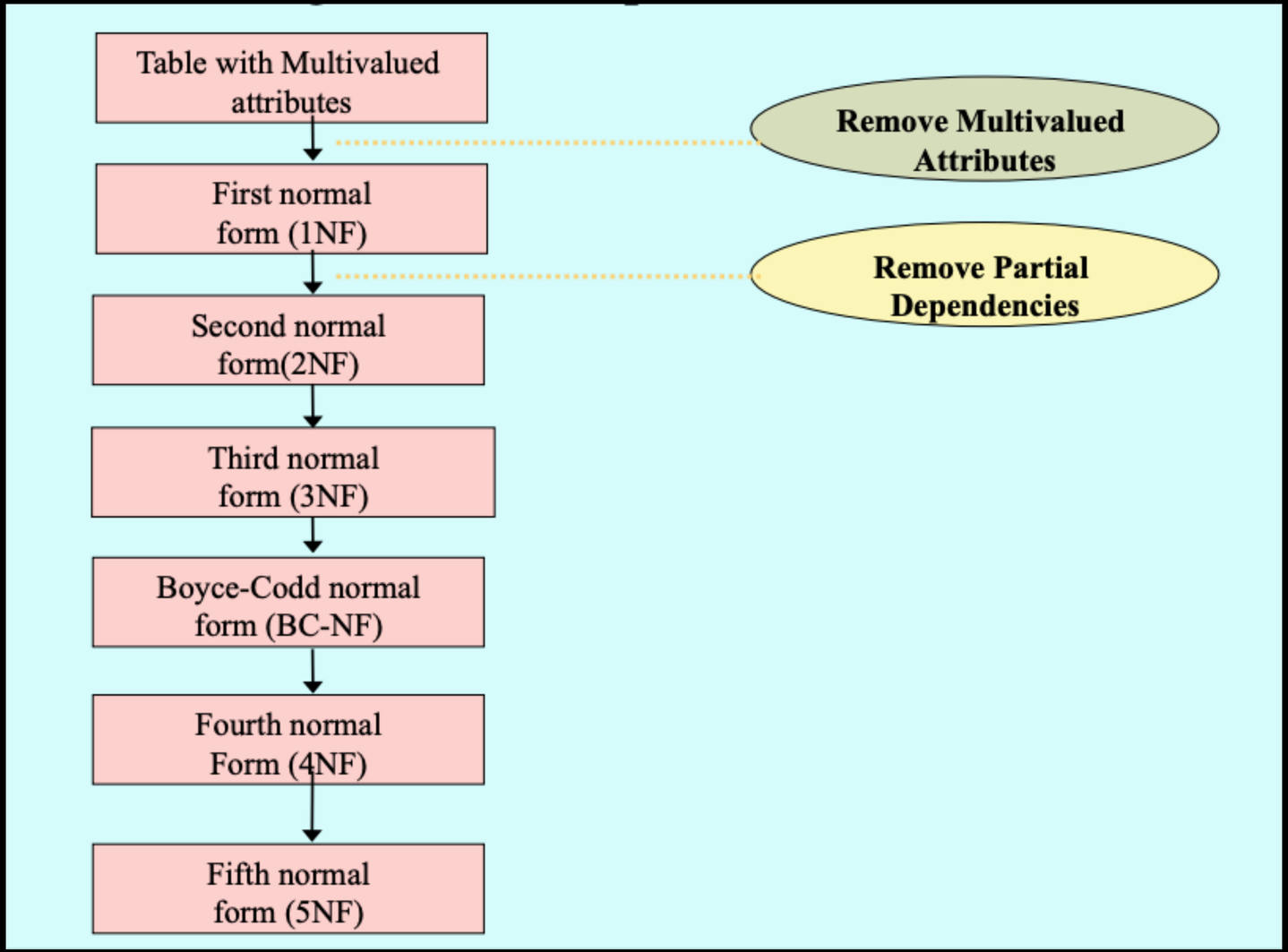
SECTION_ADVISOR

Location	Branch	Department	Batch	Section	Class Advisor

STUDENT

Location	Branch	Department	Batch	Section	Roll	Name

Second Normal Form (2NF)



3NF

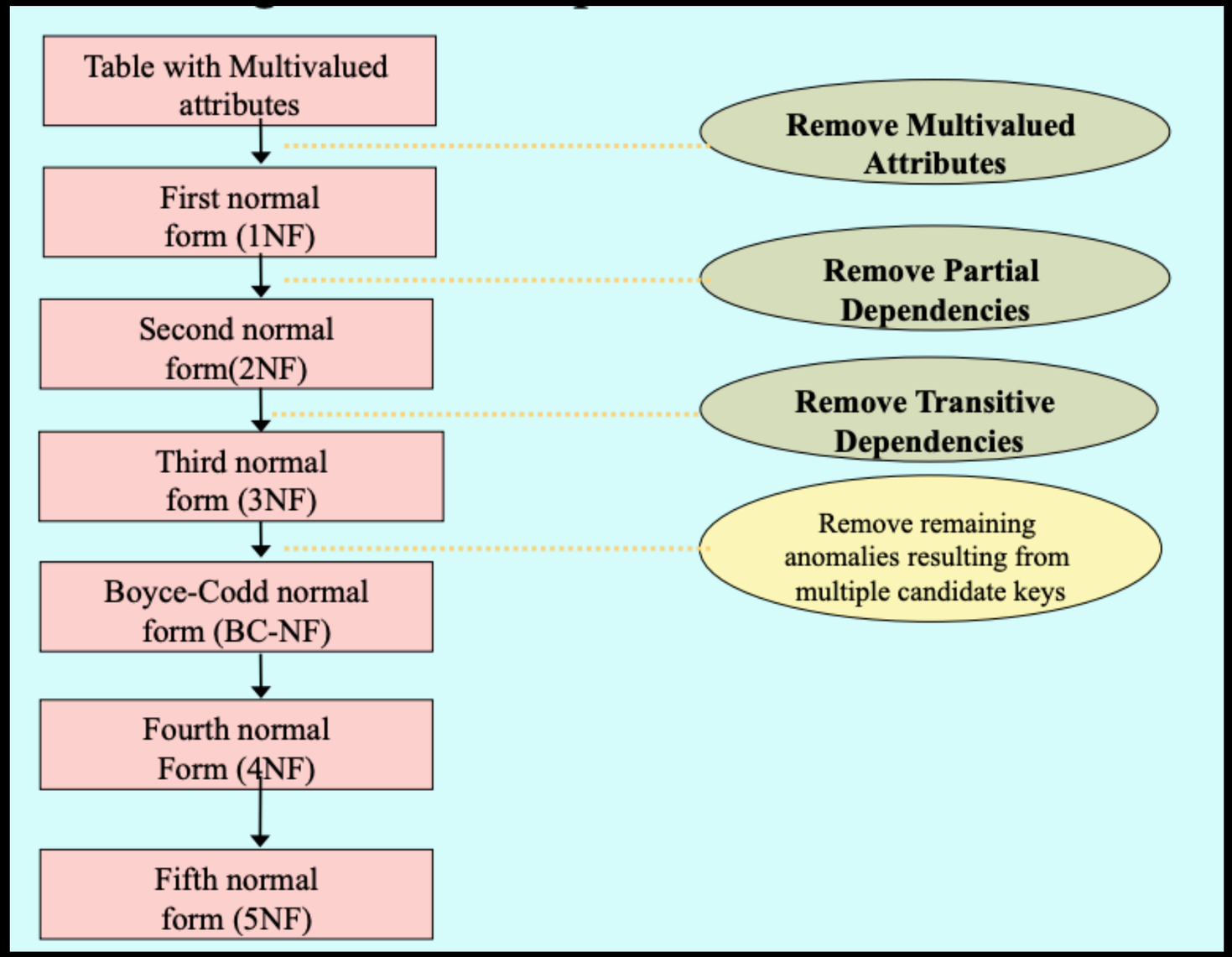


Table with Multivalued attributes
First normal form (1NF)
Second normal form(2NF)
Third normal form (3NF)
Boyce-Codd normal form (BC-NF)
Fourth normal Form (4NF)
Fifth normal form (5NF)
Remove Multivalued Attributes
Remove Partial Dependencies
Remove Transitive Dependencies
Remove remaining anomalies resulting from multiple candidate keys

4 NF

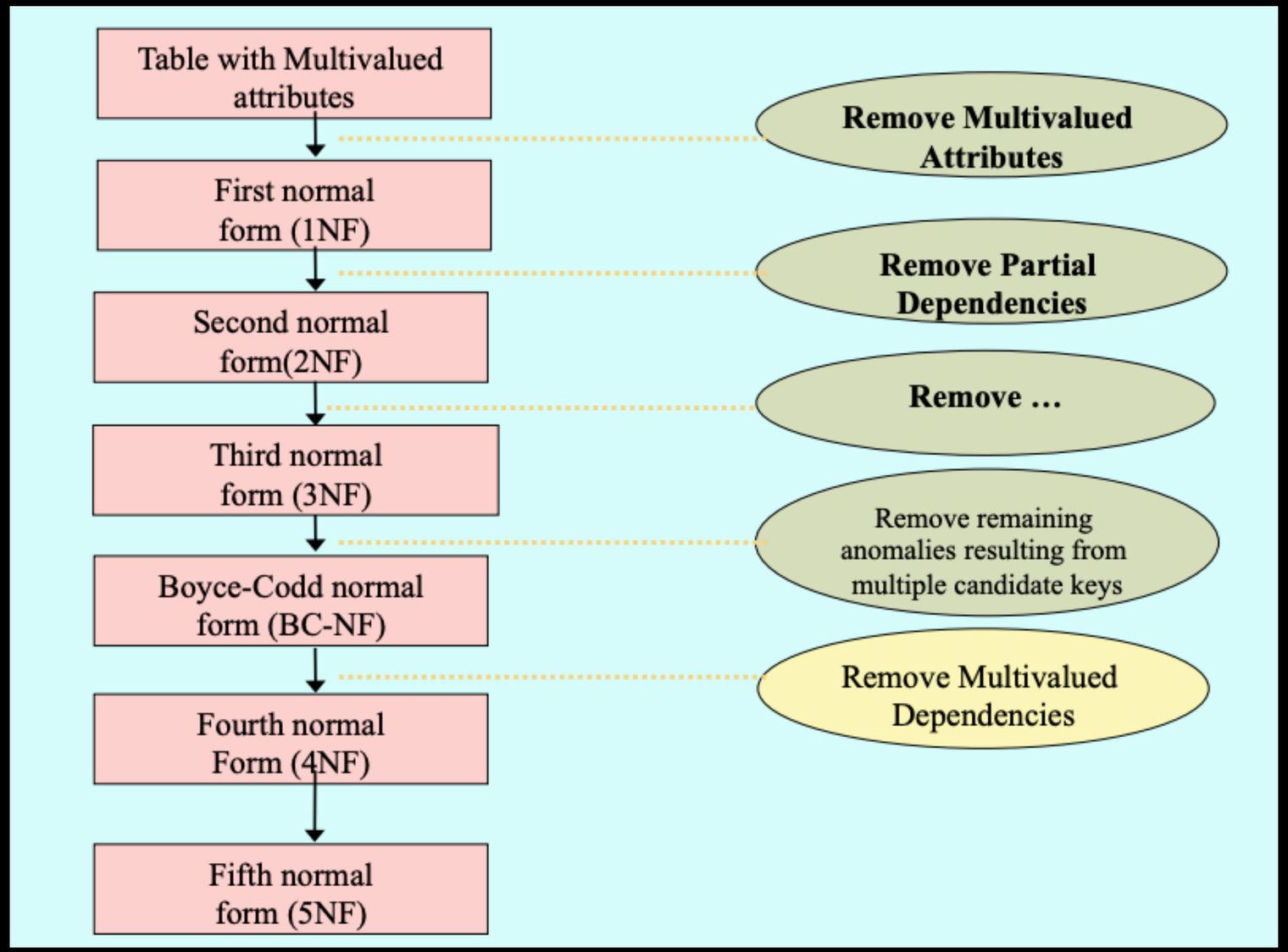


Table with Multivalued attributes
First normal form (1NF)
Second normal form(2NF)
Third normal form (3NF)
Boyce-Codd normal form (BC-NF)
Fourth normal Form (4NF)
Fifth normal form (5NF)
Remove Multivalued Attributes
Remove Partial Dependencies
Remove …
Remove remaining anomalies resulting from multiple candidate keys
Remove Multivalued Dependencies

Thank You