STA261
MANAJEMEN DATA RELASIONAL

Functional Dependencies and Normalization

DEPARTEMEN STATISTIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
INSTITUT PERTANIAN BOGOR

SEMESTER GANJIL 2021/2022

Functional Dependency

- A formal tool for analysis of relational schemas that enables us to detect and describe some of the above-mentioned **problems** in precise terms.
- The single most important concept in relational schema design theory is a **functional dependency**.

Definition:

Let A,B be *sets* of attributes.

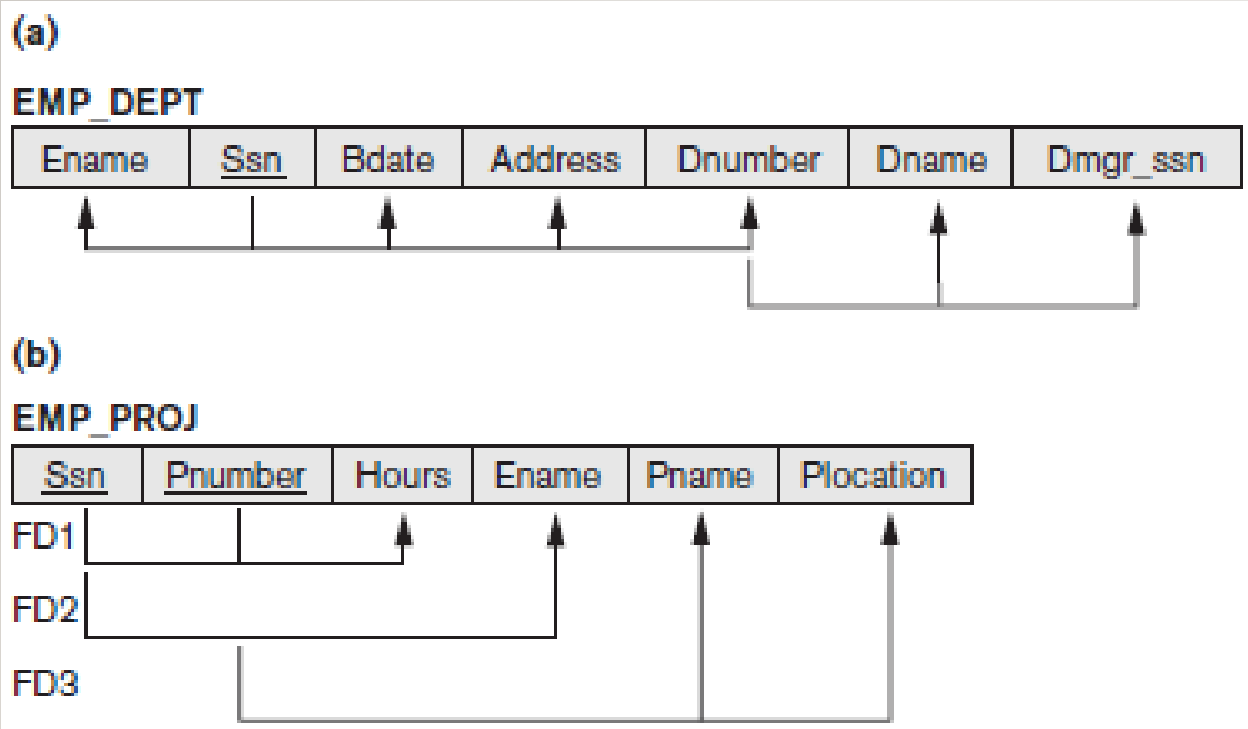
We write A → B or say A ***functionally determines*** B if, for any tuples t_1 and t_2:

$t_1[A] = t_2[A]$ implies $t_1[B] = t_2[B]$

and we call A → B a **functional dependency**

A->B means that
"whenever two tuples agree on A then they agree on B."

Functional Dependency



Ssn → {Ename, Bdate, Address, Dnumber}
Dnumber → {Dname, Dmgr_ssn}

{Ssn, Pnumber} → Hours
Ssn → Ename
Pnumber → {Pname, Plocation}

Figure 14.3 a **diagrammatic notation** for displaying FDs

Functional Dependency

a. {Ssn, Pnumber} → Hours
b. Ssn → Ename
c. Pnumber → {Pname, Plocation}

These functional dependencies specify that:

(a) a combination of **Ssn** and **Pnumber** values **uniquely** determines the number of hours the employee currently works on the project per week (**Hours**)

(b) The value of an employee's Social Security number (**Ssn) uniquely** determines the employee name (**Ename**)

(c) The value of a project's number (**Pnumber**) **uniquely** determines the project name (**Pname**) and location (**Plocation**)

Functional Dependency

Exercise:

TEACH

Teacher	Course	Text
Smith	Data Structures	Bartram
Smith	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

List all FDs and non-FDs for single attribs.

TEXT → COURSE. ✓

TEACHER → COURSE ×

TEXT → TEACHER ×

COURSE → TEXT ×

Functional Dependency

Exercise:

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

List all FDs and non-FDs for single attribs.

B → C ✓
C → B ✓
{A, B} → C ✓
{A, B} → D ✓
{C, D} → B ✓

A → B (tuples 1 and 2) ×
B → A (tuples 2 and 3) ×
D → C (tuples 3 and 4) ×

Functional Dependency

Movies	*Title*	*Director*	*Actor*
	The Birds	Hitchcock	Hedren
	The Birds	Hitchcock	Taylor
	Bladerunner	Scott	Hannah
	Apocalypse Now	Coppola	Brando

Movies : Title -> Director

Showings	*Theater*	*Screen*	*Title*	*Snack*
	Rex	1	The Birds	coffee
	Rex	1	The Birds	popcorn
	Rex	2	Bladerunner	coffee
	Rex	2	Bladerunner	popcorn
	Le Champo	1	The Birds	tea
	Le Champo	1	The Birds	popcorn
	Cinoche	1	The Birds	Coke
	Cinoche	1	The Birds	wine
	Cinoche	2	Bladerunner	Coke
	Cinoche	2	Bladerunner	wine
	Action Christine	1	The Birds	tea
	Action Christine	1	The Birds	popcorn

Showings : {Theather, Screen} -> Title

Functional Dependency

Name	Color	Category	Dep	Price
Gizmo	Green	Gadget	Toys	49
Widget	Black	Gadget	Toys	59
Gizmo	Green	Whatsit	Garden	99

{Name} → {Color}
{Category} → {Department}
{Color, Category} → {Price}
{Name, Category} → {Price}

Normalization

Normalization

- Decomposing a poorly designed (or unnormalized) relation into several relations.

Normal Form

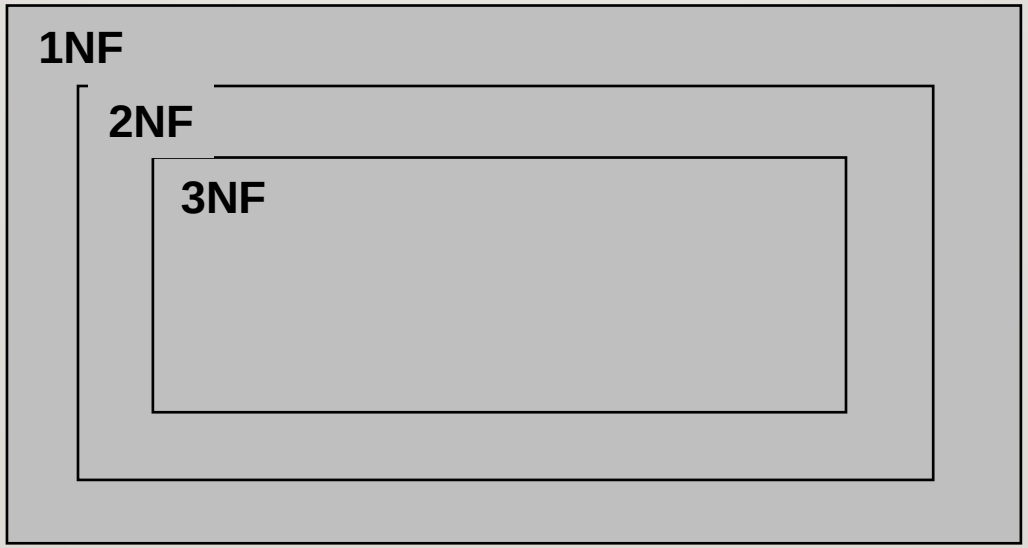
- A condition (based on functional dependencies and key properties) to determine whether a relational schema is normal/not.

Normalization

Normal Form

- 1st Normal Form (1NF) = All tables are flat
- 2nd Normal Form (2NF)
- 3rd Normal Form (3NF)

DB designs based on *functional dependencies*, intended to prevent data *anomalies*



Keys

Superkey of R:

- A set of attributes, SK, of R such that no two tuples in any valid relational instance, r(R), will have the same value for SK.
- Therefore, for any two distinct tuples, t_1 and t_2 in r(R), $t_1[SK] \neq t_2[SK]$.

Key of R:

- A minimal superkey. That is, a superkey, K, of R such that the removal of ANY attribute from K makes K **not** superkey.

If a relation has more than one keys (= **candidate keys**), we can select any one (arbitrarily) to be the **primary key**.

Keys

Key Attributes

Superkey: a set of attributes that make the tuple unique

Key K: minimal superkey

Candidate keys: several keys, one is chosen to be primary key

Prime attribute: Attribute that is part of candidate key

Non prime attribute: Attribute that is not part of any candidate key

Keys

Example:

CAR(State, LicensePlateNo, VechicleID, Model,Year, Manufacturer)

This schema has two keys:

K1 = {State, LicensePlateNo}

K2 = {VachicleID}

Both K1 and K2 are superkeys

K3 = {VechicleID, Manufacturer} is a superkey, but not a key

Normalization

1NF: First Normal Form

- Disallows
  - composite attributes
  - multivalued attributes
  - **nested relations**; attributes whose values for an *individual tuple* are non-atomic
- Considered to be part of the definition of relation

Normalization

1NF: First Normal Form (from multivalued)

(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}

(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

Figure 14.9 Normalization into 1NF.
(a) A relation schema that is **not** in 1NF.
(b) Sample state of relation DEPARTMENT.
(c) 1NF version of the same relation with redundancy.

Normalization

1NF: First Normal Form (from nested relations)

(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

Figure 14.10 Normalization into 1NF.
(a) Schema of the EMP_PROJ relation with a *nested relation* attribute PROJS.
(b) Sample extension of the EMP_PROJ relation showing nested relations within each tuple.
(c) Decomposition of EMP_PROJ into relations: EMP_PROJ1 and EMP_PROJ2 by propagating the primary key.

Normalization

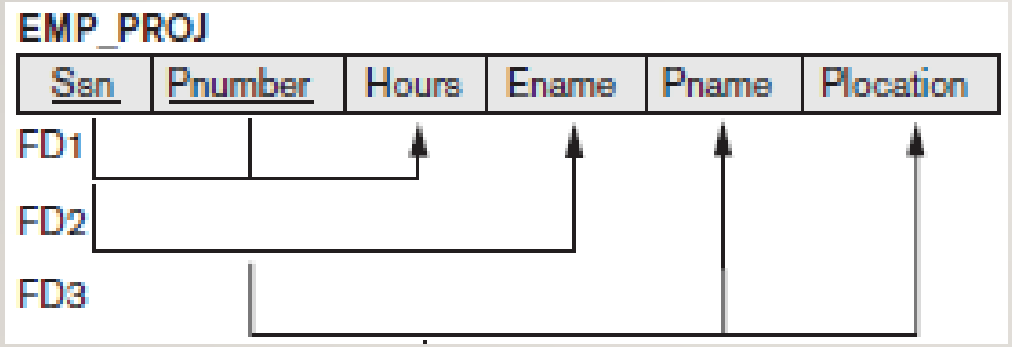
2NF: Second Normal Form

- Uses the concepts of **FDs** and **primary key**
- Definition
  - **Full functional dependency:**
    FD Y -> Z where removal of any attribute from Y means the FD does not hold any more
- Examples:
  - {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

Normalization

2NF: Second Normal Form

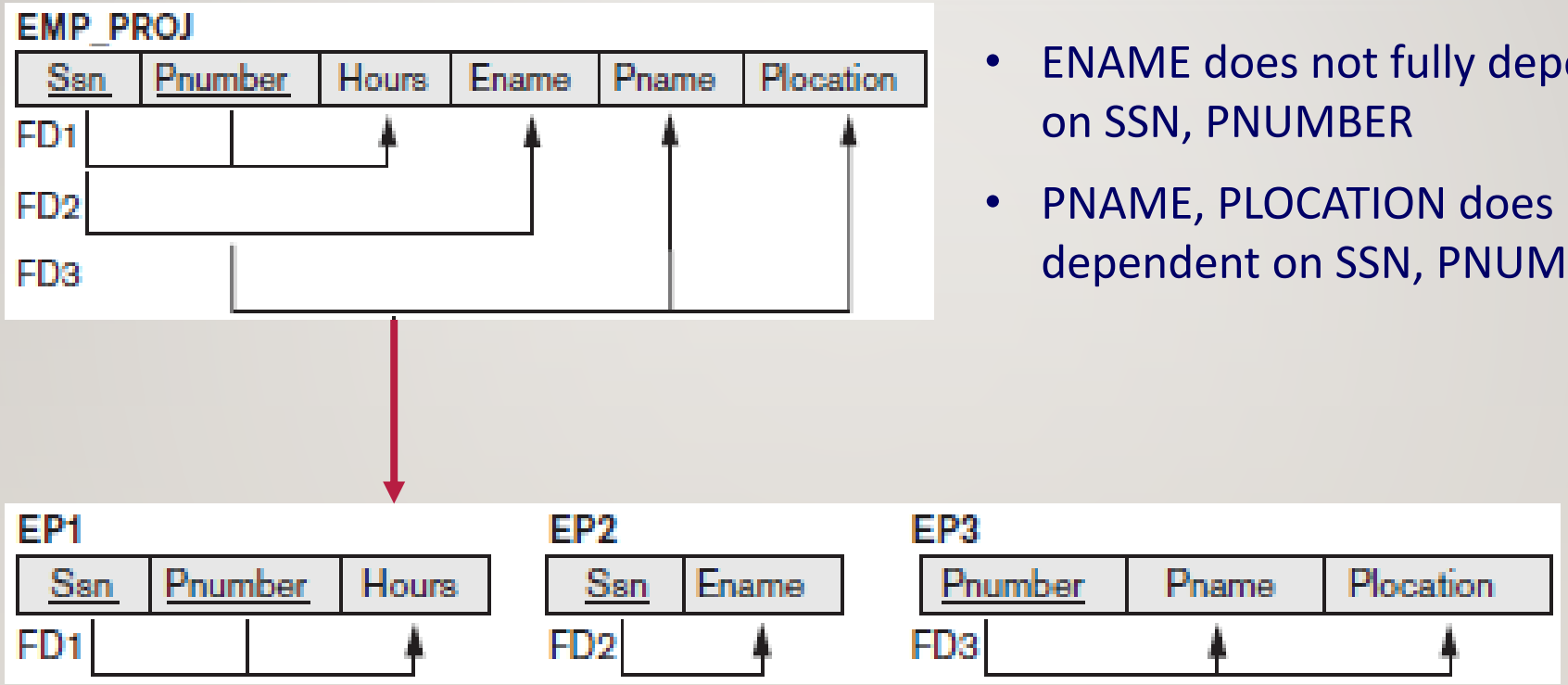
- A relation schema R is in **2NF** if every non-prime attribute A in R is fully dependent on the primary key
- R can be decomposed into 2NF relations via the process of 2NF normalization



- ENAME does **not** fully dependent on SSN, PNUMBER
- PNAME, PLOCATION does **not** fully dependent on SSN, PNUMBER

Normalization

2NF: Second Normal Form



- ENAME does not fully dependent on SSN, PNUMBER
- PNAME, PLOCATION does not fully dependent on SSN, PNUMBER

Figure 14.11 (a) Normalizing EMP_PROJ into 2NF relations.

Normalization

3NF: Third Normal Form

- A relation R is in 3NF if R is in 2NF and there is **no nonprime attribute** A in R having a **transitive dependency** with respect to primary key
- Notes:
  - On FD X -> Y dan Y -> Z, with X as primary key, we consider this a problem only if Y is **not** a candidate key.
  - Example:

    EMP (SSN, Emp#, Salary) does not violate 3NF, why?

    SSN -> Emp# and Emp# -> Salary then SSN -> Salary (transitive)

    Emp# is actually a candidate key

Normalization

3NF: Third Normal Form

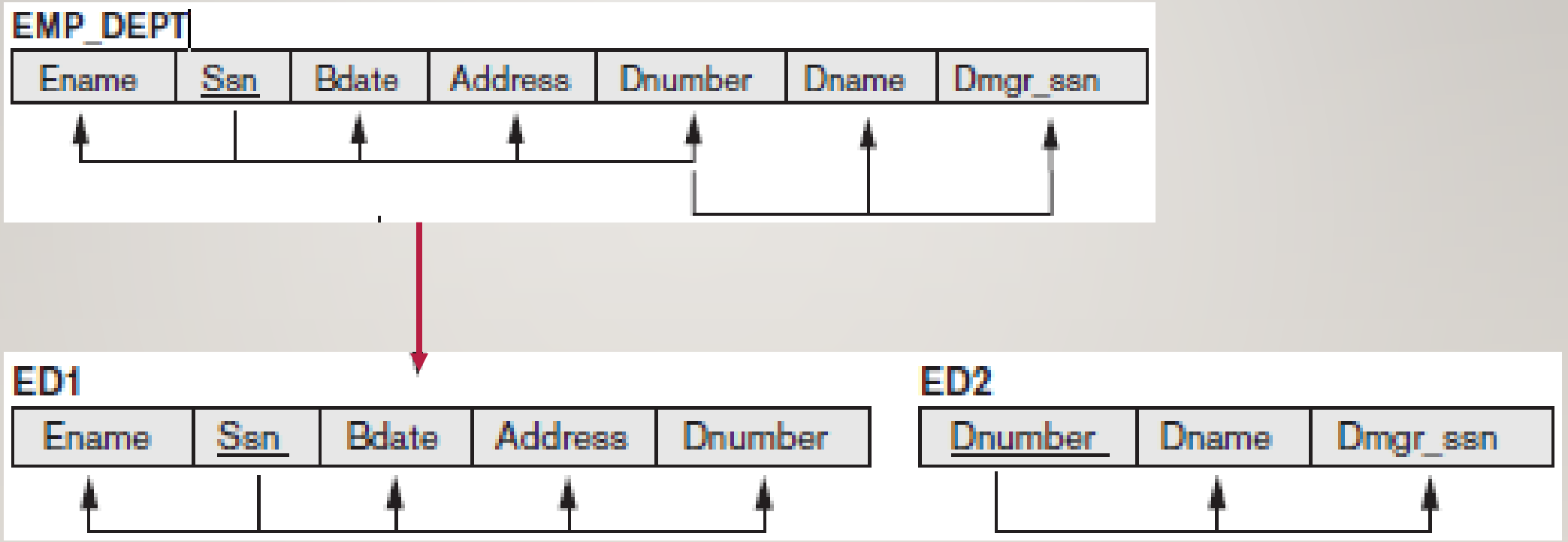


Figure 14.11 (b) Normalizing EMP_DEPT into 3NF relations.

Normalization

Normal Form	Test	Remedy (Normalization)
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 nonkey 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.
Third (3NF)	Relation should not have a nonkey attribute functionally determined by another nonkey attribute (or by a set of nonkey attributes). That is, there should be no transitive dependency of a nonkey attribute on the primary key.	Decompose and set up a relation that includes the nonkey attribute(s) that functionally determine(s) other nonkey attribute(s).

Functional Dependencies and Normalization