Database Management Systems (DBMS)

Lec 15: Relational database design (Contd.)

Ramesh K. Jallu

IIIT Raichur

Date: 23/03/21

Recap

- Normal forms based on primary key and corresponding normalization
  - **1st normal form**
    - Relation should not have multivalued attributes or nested relations
  - **2nd normal form**
    - No nonkey attribute should functionally dependent on a part of the primary key
  - **3rd normal form**
    - No transitive dependency of a nonkey attribute on the primary key

Today's overview

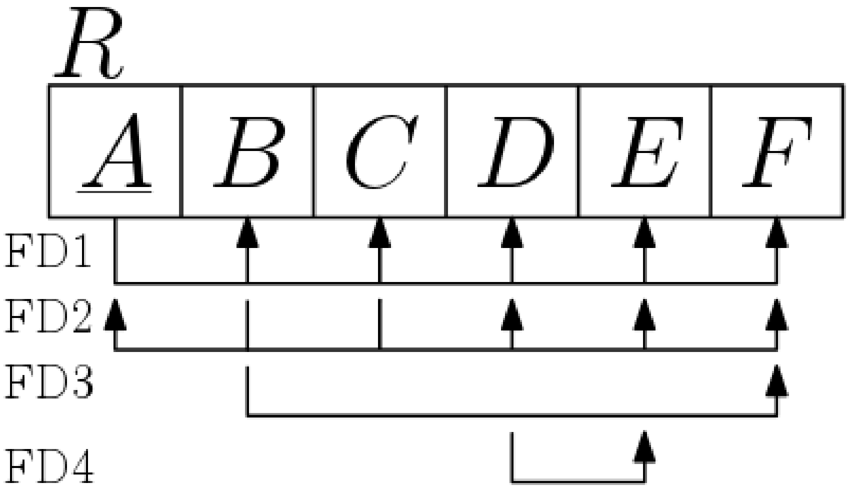
- General definitions of 2NF and 3NF
- Properties of normalization
- Boyce-Codd normal form

General definitions of 2NF and 3NF

- The discussed 2NF and 3NF are for a given database where primary keys have already been defined
- They do not take other candidate keys of a relation, if any, into account
- we now give the more general definitions of 2NF and 3NF that take all candidate keys of a relation into account
- The PD, FD, and TD are now considered w.r.t all candidate keys

1. General definition of 2NF

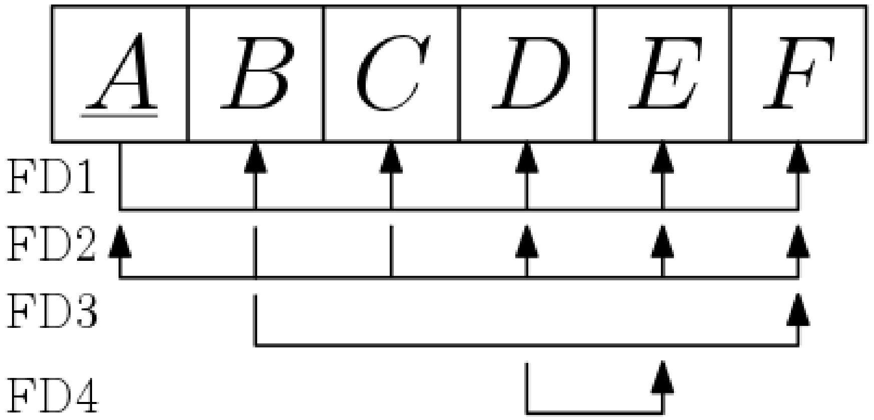
- A relation schema ***R*** is in 2NF if every nonprime attribute in ***R*** is fully functionally dependent on every key of ***R***



Candidate key $\{B, C\}$

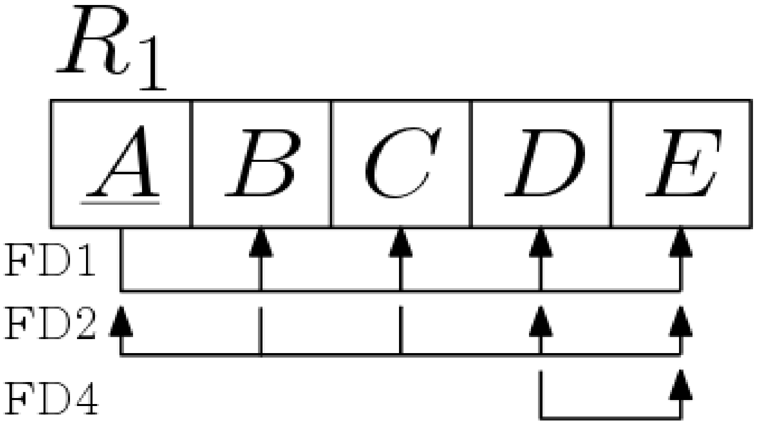
$$A \rightarrow \{B, C, D, E, F\}$$
$$\{B, C\} \rightarrow \{A, D, E, F\}$$
$$B \rightarrow F$$
$$D \rightarrow E$$

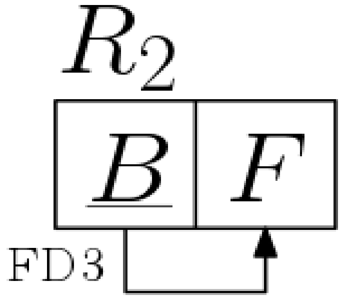
1. General definition of 2NF (Contd.)



Candidate key $\{B, C\}$

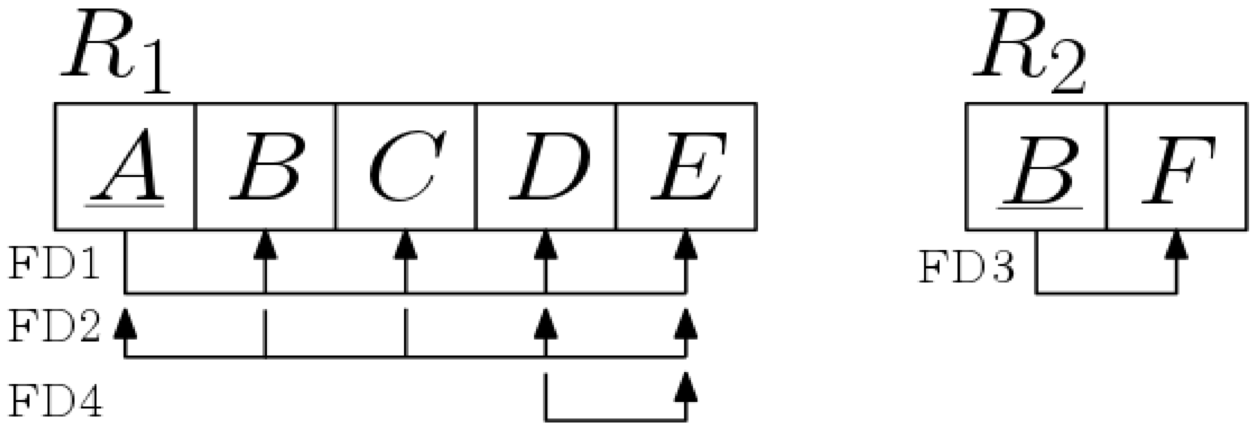
$$A \to \{B, C, D, E, F\}$$
$$\{B, C\} \to \{A, D, E, F\}$$
$$B \to F$$
$$D \to E$$



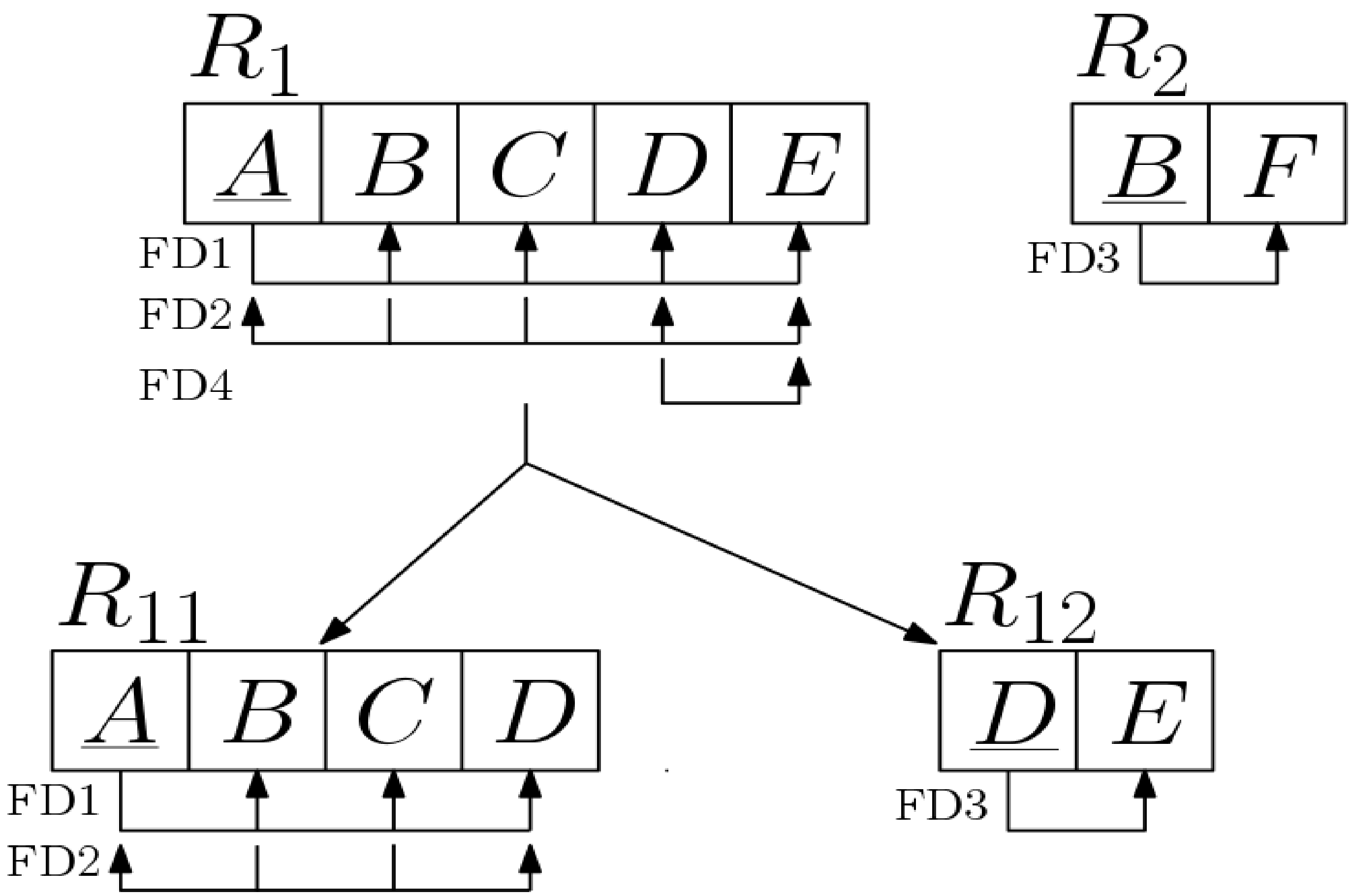


2. General definition of 3NF

- A relation schema ***R*** is in 3NF if, whenever a nontrivial functional dependency X → A (i.e., A is not a subset of X) holds in *R*, either (a) X is a superkey of R, or (b) A is a prime attribute of ***R***



2. General definition of 3NF (Contd.)



Boyce-Codd normal form (BCNF)

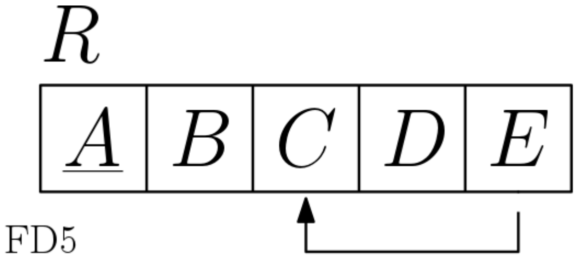
- Certain FDs escape 3NF test even though they may be potentially problematic FDs

- BCNF is a simpler form of 3NF, but is stricter than 3NF

- A relation schema ***R*** is in BCNF if whenever a nontrivial functional dependency $X \rightarrow A$ holds in ***R***, then ***X*** is a superkey of R

- Every relation in BCNF is also in 3NF; however, converse not necessarly be true

An example

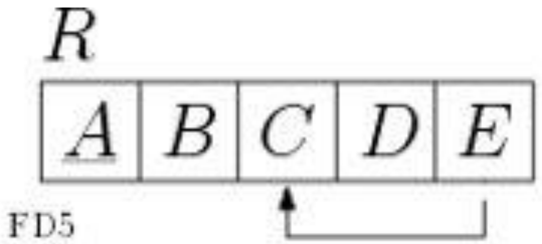


$$C = \{c_1, c_2\}$$

$$E = \{e_1, e_2, e_3, e_4, e_5\}$$

A	B	C	D	E
		c_1		e_3
		c_2		e_4
		c_1		e_1
		c_2		e_5
		$\vdots$		`$\v`
		c_2		e_4
		c_2		e_5
		c_1		e_2
		c_2		e_5

An example (Contd.)



$$C = \{c_1, c_2\}$$

$$E = \{e_1, e_2, e_3, e_4, e_5\}$$

R_1

A	B	D	E

R_2

E	C

E	C
e_1	c_1
e_2	c_1
e_3	c_1
e_4	c_2
e_5	c_2

Decomposition of Relations not in BCNF

TEACH

Student	Course	Instructor
Narayan	Database	Mark
Smith	Database	Navathe
Smith	Operating Systems	Ammar
Smith	Theory	Schulman
Wallace	Database	Mark
Wallace	Operating Systems	Ahamad
Wong	Database	Omiecinski
Zelaya	Database	Navathe
Narayan	Operating Systems	Ammar

FD1: {Student, Course} $\rightarrow$ Instructor
FD2: Instructor $\rightarrow$ Course

1. R_1 (**Student**, **Instructor**) and R_2(**Student**, **Course**)

2. R_1 (Course, **Instructor**) and R_2(**Course**, **Student**)

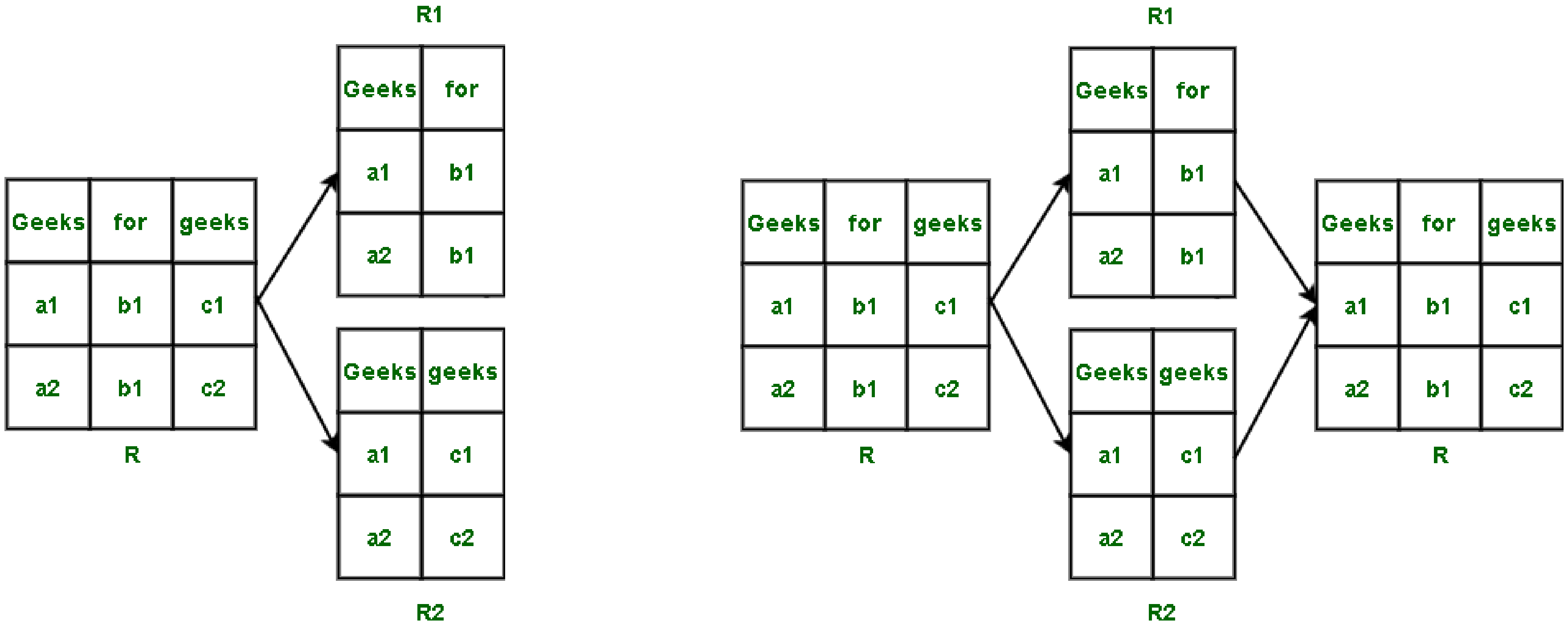
3. R_1 (**Instructor**, Course) and R_2(**Instructor**, **Student**)

Properties of normalization

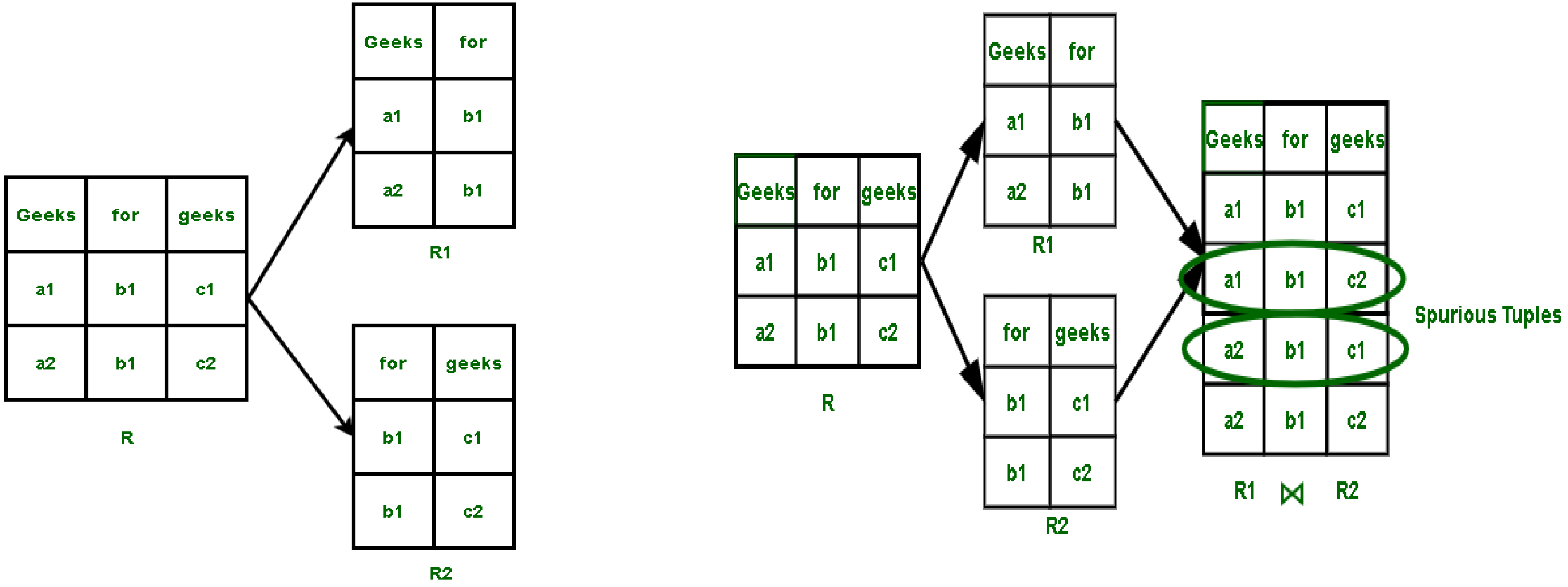
1. **The nonadditive join** or **lossless join property**
   - which guarantees that the ***spurious tuple generation problem*** does not occur with respect to the relation schemas created after decomposition

2. **The dependency preservation property**
   - which ensures that each functional dependency is represented in some individual relation resulting after decomposition

Spurious tuple generation problem



Spurious tuple generation problem



Nonadditive Join Test for Binary (NJB) Decompositions

A decomposition $D = \{R_1, R_2\}$ of R has the lossless (nonadditive) join property with respect to a set of functional dependencies F on R if and only if either

i. The FD $((R_1 \cap R_2) \rightarrow (R_1 - R_2))$ is in F^+, or

ii. The FD $((R_1 \cap R_2) \rightarrow (R_2 - R_1))$ is in F^+

Algorithm to achieve NJ decomposition

- A relation $\boldsymbol{R}$ not in BCNF can be decomposed to meet the nonadditive join property by decomposing $\boldsymbol{R}$ successively into a set of relations that are in BCNF:

- Let $\boldsymbol{R}$ be the relation not in BCNF, let $\boldsymbol{X} \subseteq \boldsymbol{R}$, and let $\boldsymbol{X} \rightarrow \boldsymbol{A}$ be the FD that causes a violation of BCNF. $\boldsymbol{R}$ may be decomposed into two relations:
  1. $\boldsymbol{R} - \boldsymbol{A}$
  2. $\boldsymbol{XA}$
  3. If either $\boldsymbol{R} - \boldsymbol{A}$ or $\boldsymbol{XA}$ is not in BCNF, repeat the process

Thank you!