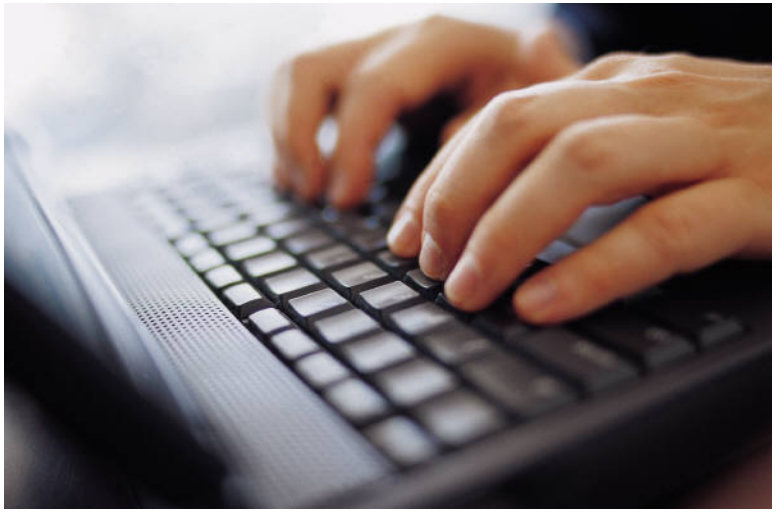
COSC344
Database Theory and Applications



Lecture 9
Normalisation

Questions to Ponder

- How can we decide whether a database design is good or not?
- How to improve the design if it is not good?

Learning Objectives of This Lecture

- You should
  - understand what is normalization, its goal and approach
  - be able to distinguish between prime and nonprime attributes
  - be able to distinguish between full and partial functional dependencies.
  - understand the definitions for 1NF, 2NF, 3NF and BCNF
  - be able to normalize a relation to 2NF, 3NF and BCNF based on the primary key

- Source
  - Textbook: Chapter 14.3-14.5

Introduction

- Normalisation

  "A step by step reversible process of replacing a given collection of relations by successive collections in which the relations have a simpler and more regular structure."

  -- E.F. Codd

- **Goal:** avoid insertion, deletion and update anomalies, and avoid spurious tuples.
- **Approach: decomposition**, that is, breaking relations into smaller relations to reduce redundancy of information
  - Not all ways of decomposing relations are equally good at minimising redundancy.
  - The normal forms give us good guidance.
  - A relation is said to be in a particular normal form if it satisfies certain constraints

Definitions

- Superkey, Key, Candidate Key, Primary Key

- **Prime Attribute:** An attribute of a relation is called a *prime attribute* if it is a member of *some candidate key* of the relation.

- **Nonprime Attribute:** An attribute is called a *nonprime attribute* if it is not a prime attribute, that is, it is not a member of *any* candidate key of the relation.

Overview of Normal Forms

- 1NF
  - All attribute values are atomic.
- 2NF
  - 1NF and every non-key attribute is fully dependent on the primary key.
- 3NF
  - 2NF and every non-key attribute is non-transitively dependent on the primary key.
- BCNF
  - Every LHS of a FD is a candidate key.
- 4NF
  - No multivalued dependencies
- 5NF
  - Every join dependency is a consequence of candidate keys

Normalization Based on Primary Key

- Given a set of functional dependencies (FDs) and a designated primary key.
- Analyse the set of relations based on the FDs and primary keys to achieve these properties:
  - minimise redundancy
  - minimise insertion, deletion and update anomalies
- Normal forms, when considered in isolation, do not guarantee a good database design. Must confirm the existence of additional properties of the relation schema:
  - Lossless join property: no spurious tuple generation problem [**Must be achieved at any cost**]
  - Dependency preservation property: all functional dependencies are represented [**Sometimes have to be sacrificed to improve performance**]

First Normal Form (1NF)

Definition: A relation schema is in 1NF if and only if it meets the following two requirements: (1) The domain of an attribute must include only atomic values; (2) the value of any attribute in a tuple must be a single value from the domain of the attribute.

Remove repeating groups into a new relation

1NF disallows multivalued attributes, composite attributes or nested relations. All attribute values must be atomic.

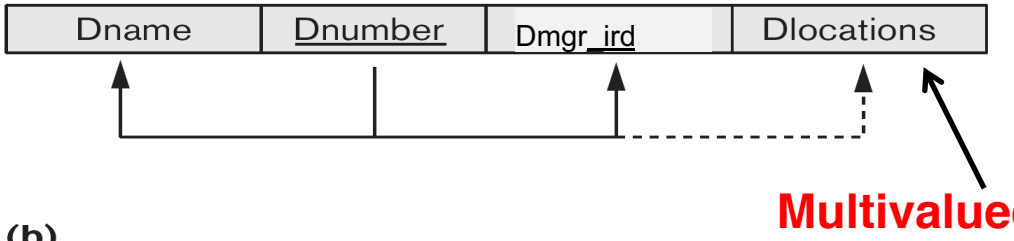
Steps to normalize a relation into 1NF:

- Remove the multivalued attribute or composite attribute, and create a new relation to contain it.
- Add to the new relation a copy of the PK of the original relation.
- Determine the PK of the new relation

First Normal Form Example (1)

(a)

DEPARTMENT



(b)

DEPARTMENT

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

(c)

DEPARTMENT

Dname	Dnumber	Dmgr_ird	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

Figure 15.9
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.

First Normal Form Example (2)

- Three main approaches to achieve 1NF
  - Remove Dlocations and put it into a separate relation along with the primary key of DEPARTMENT. The PK of the new relation will be {Dnumber, Dlocation}. **[The best approach]**
  - Expand the key so there will be a separate tuple in the original DEPARTMENT relation for each location of a department. PK becomes {Dnumber, Dlocation}. **[Introduce redundancy]**
  - If the maximum number of values is known for Dlocations, replace it with atomic attributes Dlocation1, Dlocation2, . . . , Dlocationn. **[Introduce NULLs]**

First Normal Form Example (3)

EMP_PROJ is shown with multivalued attributes that are composite (Called nested relations).

(a)

EMP_PROJ

		Projs	
IRD	Ename	Pnumber	Hours

(b)

EMP_PROJ

IRD	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

IRD	Ename

EMP_PROJ2

IRD	Pnumber	Hours

Figure 15.10
Normalizing nested relations 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.

Second Normal Form (2NF)

- **Full functional dependency**: a functional dependency X->Y is a full functional dependency if removal of any attribute from X means that the dependency does not hold any more.
- **Partial functional dependency**: a functional dependency X->Y is a partial functional dependency if some attribute can be removed from X and the dependency still holds.

Definition: A relation schema R is in 2NF if every nonprime attribute in R is fully functionally dependent on the primary key of R.

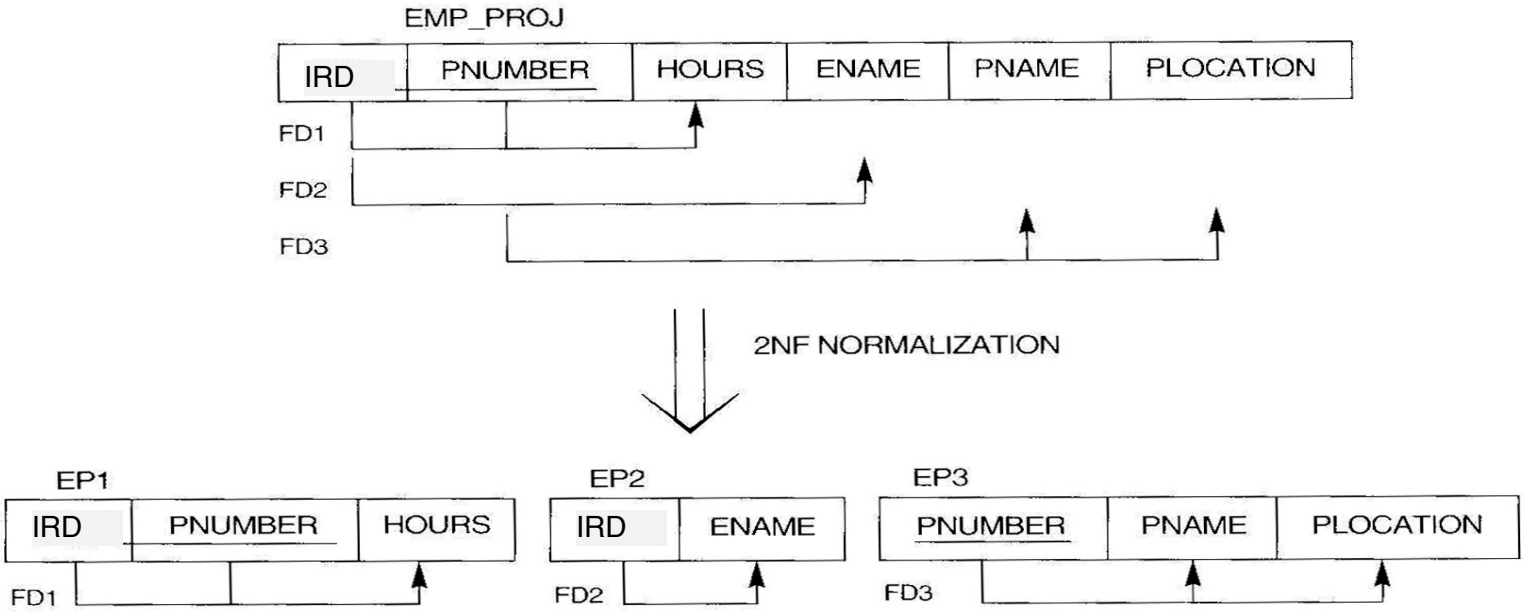
Remove partial functional dependencies into a new relation

- The test for 2NF involves testing for FDs whose LHS attributes are part of the PK.
  - If the PK contains a single attribute, the test does not need to be done.

2NF Normalization Steps and Example

Steps from 1NF to 2NF:

- Remove the nonprime attributes that are only partially functionally dependent on the primary key, and place them in a new relation.
- Add to this relation a copy of the attribute(s) which are the determinants of these nonprime attributes. These attribute(s) will automatically become the primary key of this new relation.



Third Normal Form (3NF)

- **Transitive dependency:** a functional dependency X->Y in a relation schema R is a transitive dependency if there exists a set of attributes Z in R that is neither a candidate key nor a subset of a key of R, and both X->Z and Z->Y hold.

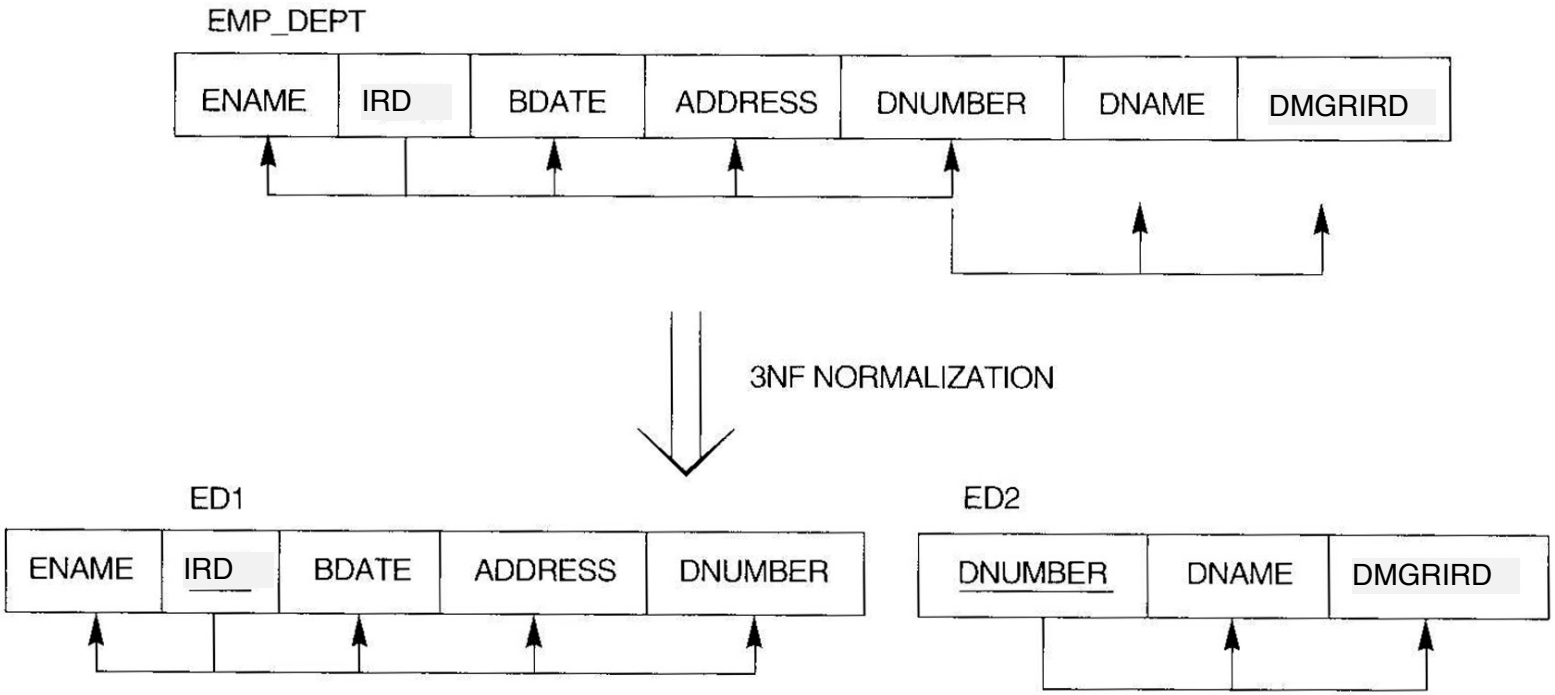
Definition: A relation is in 3NF if
(a) it is in 2NF
(b) no nonprime attribute is transitively dependent on the primary key.

Remove transitive dependencies into a new relation

Steps from 2NF to 3NF:

- Remove the nonprime attributes that are transitively dependent on the key attribute(s), and place them in a new relation.
- Add to this relation a copy of the attribute(s) which are the determinants of these nonprime attributes. These attributes will automatically become the primary key of this new relation.

Third Normal Form Example



Any functional dependency in which the LHS is part (a subset) of the primary key or a nonkey attribute is a problematic FD.

General Definitions of 2NF and 3NF

- Previous normal forms are defined based on the primary key.
- The general definitions take into account any candidate key.

2NF Definition: A relation schema R is in 2NF if every nonprime attribute in R is not partially dependent on **any** key of R. (**OR** every nonprime attribute in R is fully functionally dependent on **every** key of R.

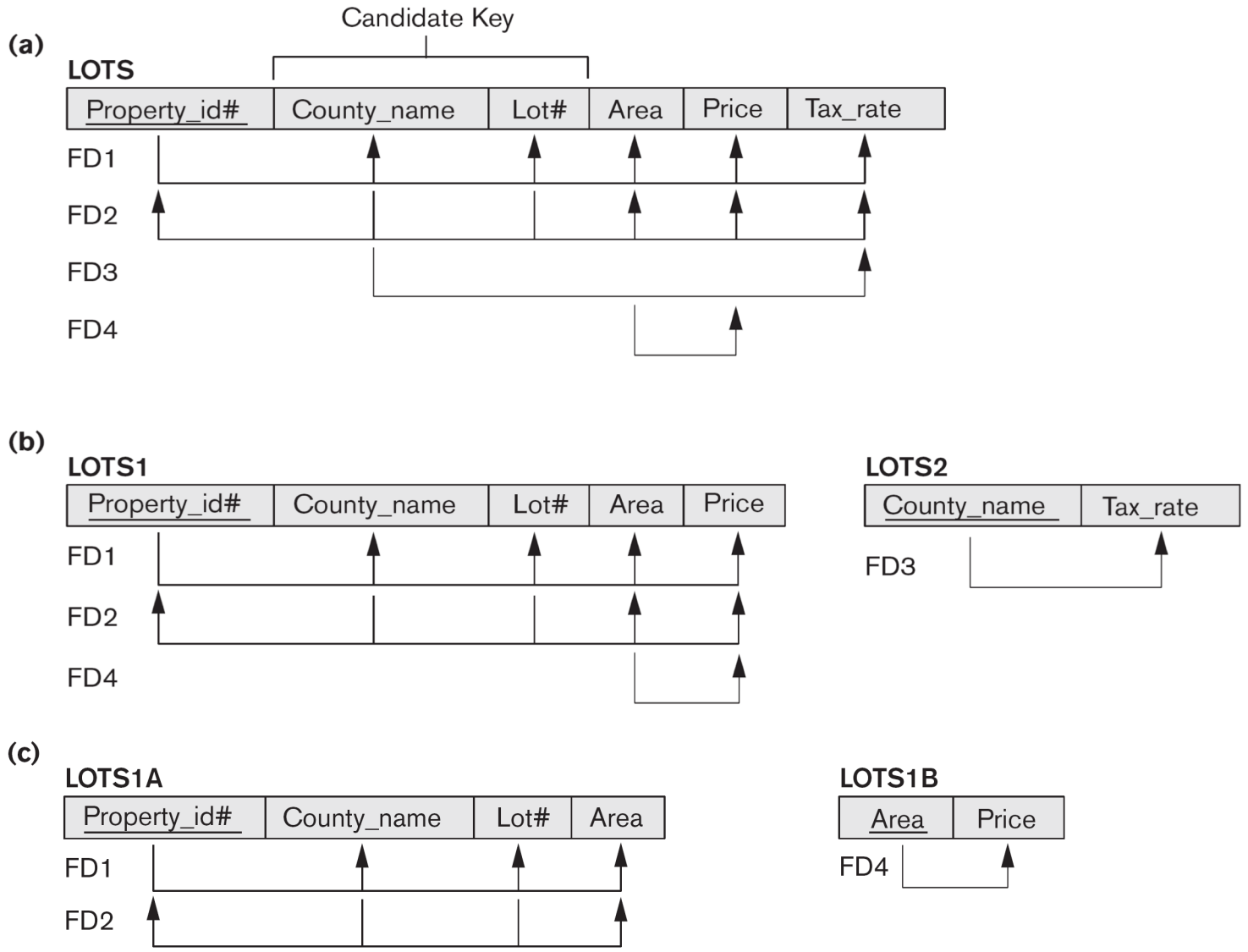
3NF Definition: A relation schema R is in 3NF if, whenever a nontrivial functional dependency X->A holds in R, either (a) X is a superkey of R, or (b) A is a prime attribute of R.

Alternative Definition of 3NF: A relation schema R is in 3NF if every nonprime attribute of R meets the following conditions:

(a) It is fully functionally dependent on every key of R.

(b) It is nontransitively dependent on every key of R.

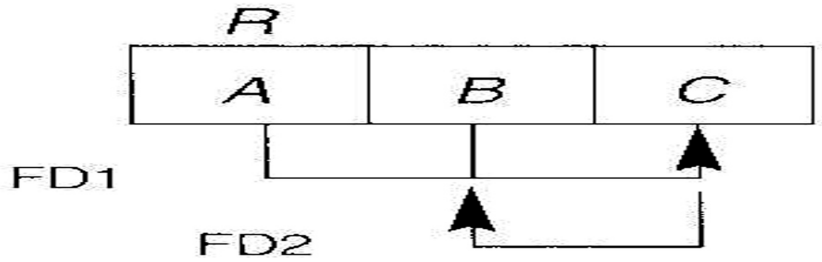
Example - General Definition of 2NF and 3NF



Boyce-Codd Normal Form (BCNF)

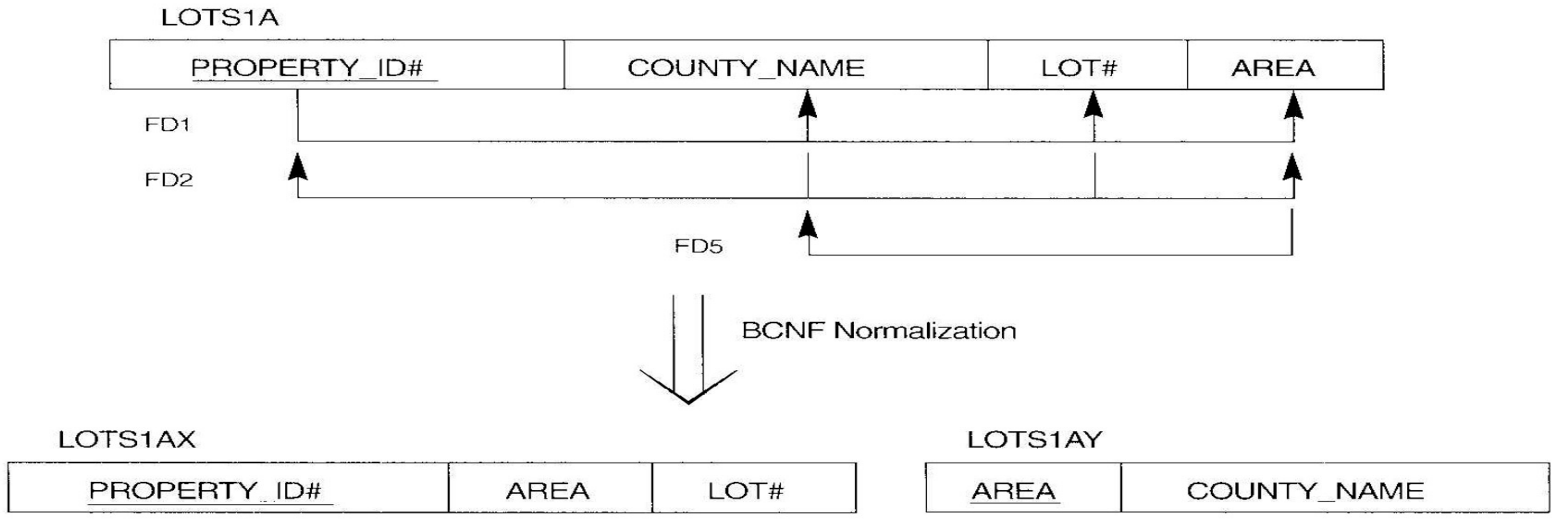
BCNF Definition: A relation schema R is in 3NF if whenever a nontrivial functional dependency X->A holds in R, X is a superkey of R.

- Every relation in BCNF is also in 3NF; However, a relation in 3NF is not necessarily in BCNF
- In practice, most relation schemas that are in 3NF are also in BCNF. Only if X -> A holds in a relation schema R with X not being a superkey and A being a prime attribute will R be in 3NF but not in BCNF.

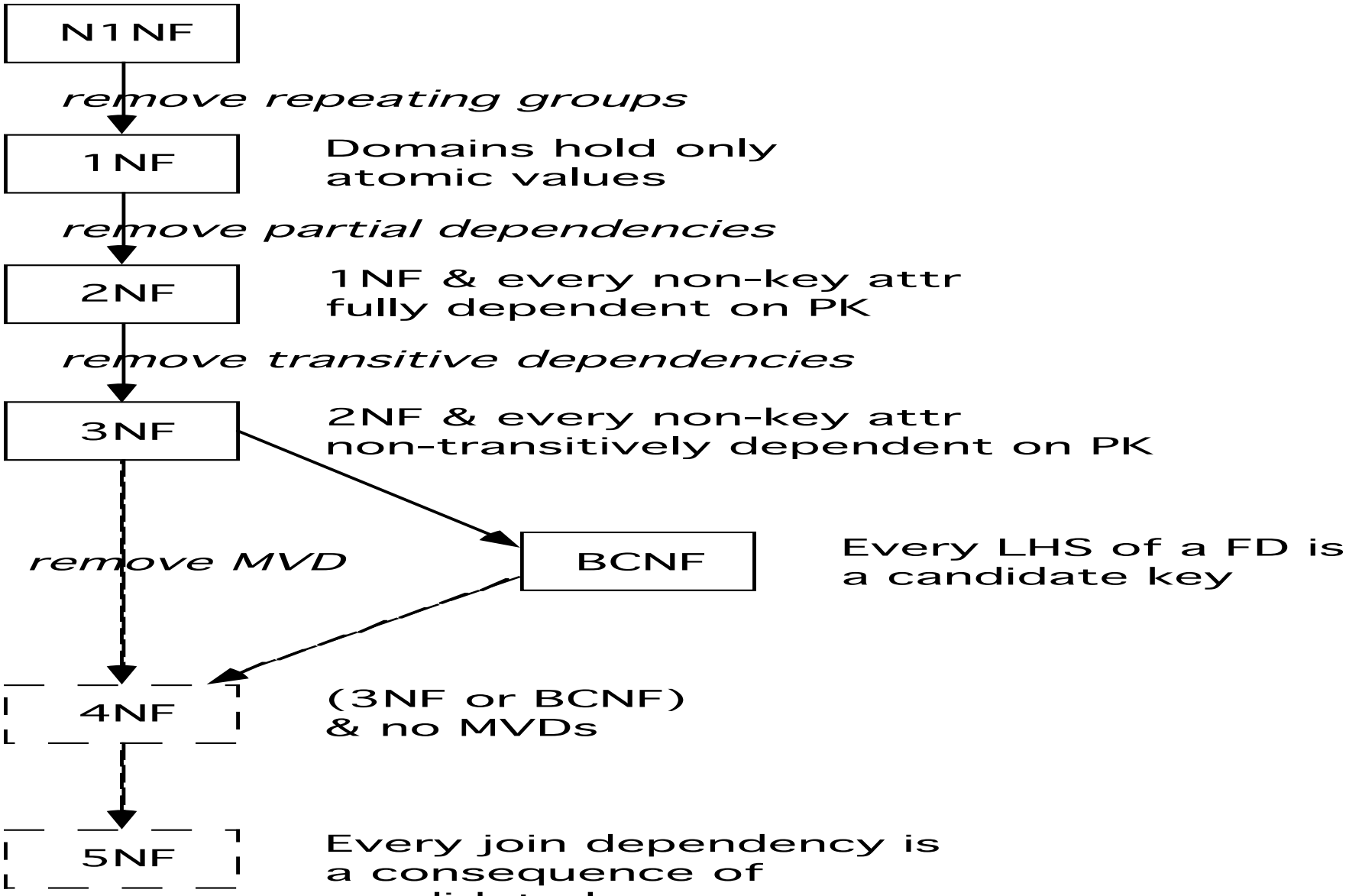


Example - BCNF

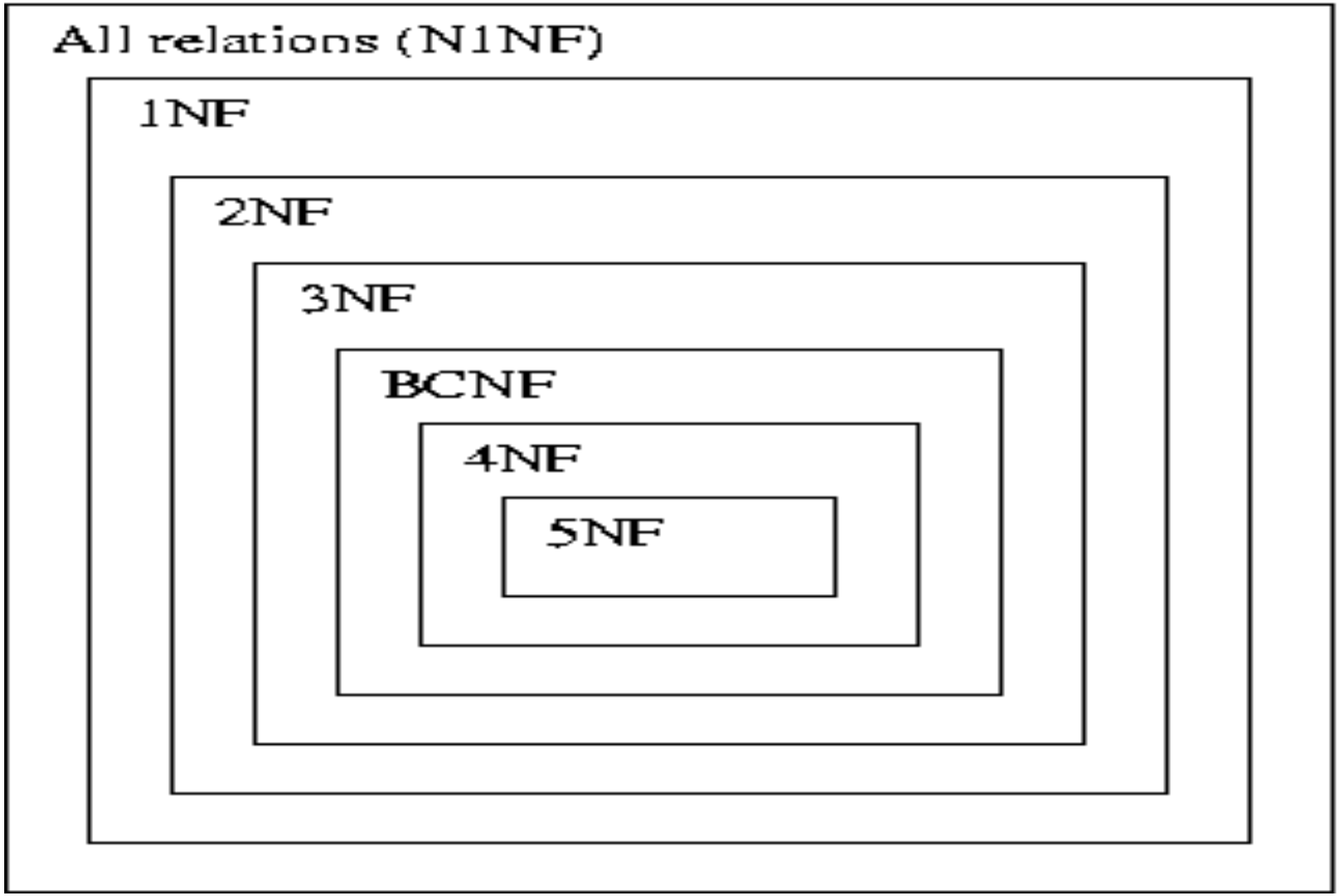
- Suppose we have thousands of lots, but only 2 counties. Also assume that lots in *county1* have sizes of 0.5, 0.6, 0.7, 0.8, 0.9 and 1.0 acres. Lots in *county2* have sizes of 1.1, 1.2, . . ., 2.0 acres.
- FD5 holds



A Simplified Normalisation Sequence



A step by step reversible process



Do not necessarily need to normalise to the highest form.