

CS 275: Introduction to Databases



Functional Dependencies

- Formal tool for analysis of relational schemas
- Enables us to detect and describe some problems in precise terms
- Theory of functional dependency

Definition of Functional Dependency

- Constraint between two sets of attributes from the database

Definition. A functional dependency, denoted by $X \rightarrow Y$, between two sets of attributes X and Y that are subsets of R specifies a *constraint* on the possible tuples that can form a relation state r of R . The constraint is that, for any two tuples t_1 and t_2 in r that have $t_1[X] = t_2[X]$, they must also have $t_1[Y] = t_2[Y]$.

- Property of semantics or meaning of the attributes
- **Legal relation states**
 - Satisfy the functional dependency constraints

Definition of Functional Dependency (cont.)

- A set of attributes X functionally determines a set of attributes Y if the value of X uniquely determines the values of Y
- If two tuples have the same X values, then they must have the same Y values
- A set of attributes Y is functionally dependent on the set of attributes X .

Functional Dependencies

- For instance:
 - SSN->ENAME
 - PNUMBER->{PNAME, PLOCATION}
 - {SSN, PNUMBER}->HOURS

Functional Dependencies

- Given a populated relation
 - Cannot determine which FDs hold and which do not
 - Can state that FD does not hold if there are tuples that show violation of such an FD

Normalization of Relations

- Takes a relation schema through a series of tests
 - Certify whether it satisfies a certain normal form
 - Proceeds in a top-down fashion
- **Normal form tests**

Definition. The **normal form** of a relation refers to the highest normal form condition that it meets, and hence indicates the degree to which it has been normalized.

Normalization of Relations (cont'd.)

- Properties that the relational schemas should have:
 - **Nonadditive join property**
 - Extremely critical
 - **Dependency preservation property**
 - Desirable but sometimes sacrificed for other factors

Normalization of Relations (cont'd.)

- Normalization: the process of decomposing unsatisfactory relations by breaking their attributes into smaller ones
- Normal form: condition using keys and FDs of a relation to certify whether a relation scheme is in a particular form

First Normal Form

- Part of the formal definition of a relation in the basic (flat) relational model
- Only attribute values permitted are single **atomic (or indivisible) values**
- Techniques to achieve first normal form
 - Remove attribute and place in separate relation
 - Expand the key
 - Use several atomic attributes

1NF

(a)

DEPARTMENT			
DNAME	<u>DNUMBER</u>	DMGRSSN	DLOCATIONS

(b)

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

(c)

DEPARTMENT			
DNAME	<u>DNUMBER</u>	DMGRSSN	<u>DLOCATION</u>
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

Definitions of Keys and Attributes Participating in Keys

- Definition of **superkey** and **key**
- **Candidate key**
 - If more than one key in a relation schema
 - One is **primary key**
 - Others are **secondary keys**

Definition. An attribute of relation schema R is called a **prime attribute** of R if it is a member of *some candidate key* of R . An attribute is called **nonprime** if it is not a prime attribute—that is, if it is not a member of any candidate key.

Second Normal Form

- Based on concept of **full functional dependency**

- Versus **partial dependency**

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

- Second normalize into a number of 2NF relations
 - Nonprime attributes are associated only with part of primary key on which they are fully functionally dependent

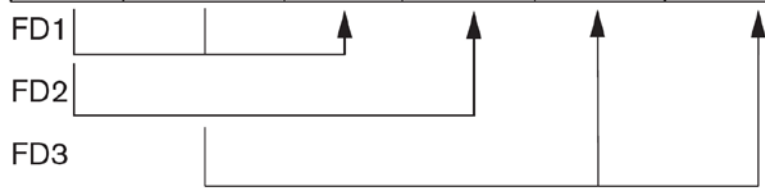
2NF

- Examples:
 - {SSN, Pnumber}->Hours is full.
 - {SSN}->EName is *not* full.

(a)

EMP_PROJ

<u>Ssn</u>	<u>Pnumber</u>	Hours	Ename	Pname	Plocation
------------	----------------	-------	-------	-------	-----------



2NF Normalization

EP1

<u>Ssn</u>	<u>Pnumber</u>	Hours
------------	----------------	-------



EP2

<u>Ssn</u>	Ename
------------	-------



EP3

<u>Pnumber</u>	Pname	Plocation
----------------	-------	-----------



Third Normal Form

- Based on concept of transitive dependency

Definition. According to Codd's original definition, a relation schema R is in 3NF if it satisfies 2NF *and* no nonprime attribute of R is transitively dependent on the primary key.

- Problematic FD
 - Left-hand side is part of primary key
 - Left-hand side is a nonkey attribute

3NF

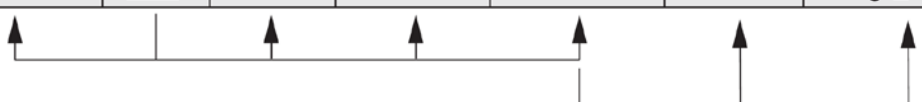
- Examples:

- SSN- \rightarrow DMgrSSN is a transitive FD since SSN- \rightarrow DNumber and DNumber- \rightarrow DMgrSSN hold
- SSN- \rightarrow ENAME is non-transitive since there is no X such that SSN- \rightarrow X and X- \rightarrow ENAME.

(b)

EMP_DEPT


Ename	<u>Ssn</u>	Bdate	Address	Dnumber	Dname	Dmgr_ssn
-------	------------	-------	---------	---------	-------	----------



3NF Normalization

ED1

Ename	<u>Ssn</u>	Bdate	Address	Dnumber
-------	------------	-------	---------	---------



ED2

<u>Dnumber</u>	Dname	Dmgr_ssn
----------------	-------	----------

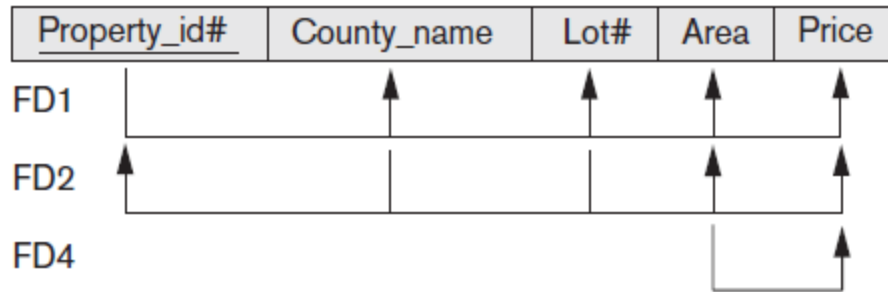


General Definitions

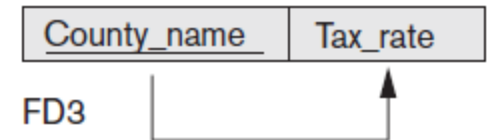
- Above definitions only consider primary key.
- General definitions:
 - A relation schema R is in 2NF if every nonprime attributes is fully functionally dependent on every key of R .
 - A relation schema R is in 3NF if whenever a FD $X \rightarrow A$ holds, then either X is a superkey or A is a prime attribute.

(b)

LOTS1

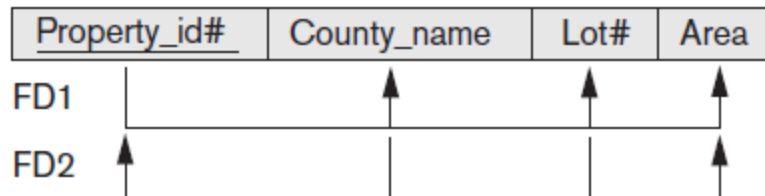


LOTS2

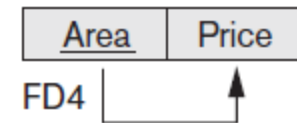


(c)

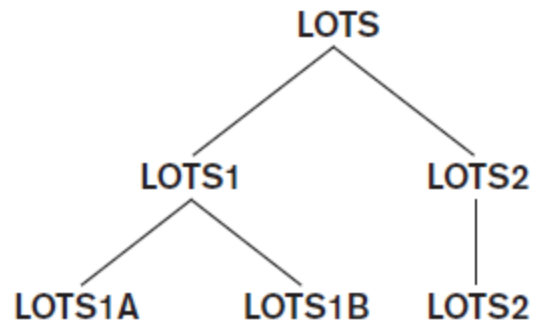
LOTS1A



LOTS1B



(d)



1NF

2NF

3NF

General Definitions of Second and Third Normal Forms

Table 15.1 Summary of Normal Forms Based on Primary Keys and Corresponding 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).