U O W

Database Normalization

(1NF and 2NF)

CSIT882: Data Management Systems

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Outline

- First Normal Form (1NF)
- Full/partial functional dependencies
- Second Normal Form (2NF)

First normal form (1NF)

Relational schema is in 1NF if all occurrences of rows in the respective relational table contain the same number of fields and include the atomic values only, (there is no repeating fields and groups)

e#	name	car used
950001	Peter	Toyota, PKR234, Ford, WER545
932345	Paul	Honda, RTQ456
960020	Joan	Holden, KLR197, Holden, KLR567

First normal form (1NF)

Relational schema is in 1NF if all occurrences of rows in the respective relational table contain the same number of fields and include the atomic values only, (there is no repeating fields and groups)

e#	name	Hobbies
950001	Peter	Cooking, Reading, Traveling
932345	Paul	Playing games, Playing chess, Reading
960020	Joan	Reading, Swimming

Outline

- First Normal Form (1NF)
- Full/partial functional dependencies
- Second Normal Form (2NF)

Full and partial functional dependencies

Full functional dependency

Full functional dependency is a functional dependency $X \rightarrow Y$ such that removal of any attribute A from X causes that $(X-A) \not\rightarrow Y$

Partial functional dependency

Partial functional dependency is a functional dependency that is not full.
Such that removal of any attribute A from X causes that $(X-A) \rightarrow Y$

Attributes

Prime attribute

Prime attribute is an attribute from relational schema R which is a member of at least one candidate key in R

Nonprime attribute

Nonprime attribute is an attribute which is a not prime

Outline

- First Normal Form (1NF)
- Full and partial functional dependencies
- Second Normal Form (2NF)

Second normal form (2NF)

Relational schema R is in 2NF if every nonprime attribute A in R is fully functionally dependent on a primary key of schema R

Second normal form (2NF)

Inventory

part	quantity	warehouse	warehouse-address

warehouse → warehouse-address

part, warehouse → quantity

---

part, warehouse → warehouse-address

---

part, warehouse → warehouse-address, quantity

Minimal key = (part, warehouse)

Second normal form (2NF)

Inventory

part	quantity	warehouse	warehouse-address

warehouse → warehouse-address

part, warehouse → quantity

Minimal key = (part, warehouse)

Schema `Inventory` <u>is not in 2NF</u> because nonprime attribute `warehouse-address` depends on a part (`warehouse`) of a key (`part, warehouse`)

Functional dependency which "violates 2NF" is
warehouse → warehouse-address

Second normal form (2NF)

Inventory

part	quantity	warehouse	warehouse-address

warehouse → warehouse-address

part, warehouse → quantity

Minimal key = (part, warehouse)

If all minimal keys in a schema consist of only one attribute (single attribute keys) then such schema is always in 2NF

This is because any nonprime attribute in the schema does not depend on a part of a key (because each key consists of one attribute only !)

Second normal form (2NF)

Inventory

part	quantity	warehouse	warehouse-address

Schema Inventory should be decomposed into the following schemas

Store

part	quantity	warehouse

Location

warehouse	warehouse-address

Second normal form (2NF)

Store

part	quantity	warehouse

part, warehouse → quantity

Minimal key = (part, warehouse)

Schema Store <u>is in 2NF</u> because no nonprime attributes (quantity) depends on a part of a key (part, warehouse)

Second normal form (2NF)

Location

warehouse	warehouse-address

warehouse → warehouse-address

Minimal key = (warehouse)

Schema Location is in 2NF because no nonprime attributes (warehouse-address) depends on a part of a key (warehouse)

Second normal form (2NF)

Location

warehouse	warehouse-address

warehouse → warehouse-address

Minimal key = (warehouse)

Every relational schema, which consists of at most 2 attributes is always in 2NF

References

Elmasri R., Navathe S. B., *Fundamentals of Database Systems*, chapters 10.3,10.4,10.5

R. Ramakrishnan, J. Gehrke *Database Management Systems*, chapters 19.2, 19.3

Database Normalization
(3NF and BCNF)

CSIT882: Data Management Systems

UNIVERSITY
OF WOLLONGONG
AUSTRALIA

Outline

- Third Normal Forms (3NF)
- Boyce-Codd Normal Form (BCNF)

Transitive functional dependency

In a relational schema R, if there exists nonempty subsets X, Y and Z of R, there are valid functional dependencies X → Z and Z → Y, and such that the functional dependencies X → Y is valid in R

We say that Y is transitively dependent on X in schema R if X → Y is valid in R and X → Y is a transitive functional dependency

Third normal form (3NF)

Relational schema R is in 3NF if it is in 2NF and no nonprime attribute of R is transitively dependent on the primary key

Third normal form (3NF)

Supplier

s# sname company-name city

s# → sname

s# → company-name

s# → city

company-name → city

Minimal key = (s#)

Schema Supplier is in 2NF because there are no nonprime attributes which are functionally dependent on a part of minimal key s#

Third normal form (3NF)

Supplier

s#	sname	company-name	city

s# → sname

s# → company-name

s# → city

company-name → city

Minimal key = (s#)

Schema Supplier is not in 3NF because attribute city is transitively dependent on s#

s# → company-name & company-name → city

Third normal form (3NF)

Supplier

s#	sname	company-name	city

Schema `Supplier` should be decomposed into the following schemas

Supplier-new

s#	sname	company-name

Company

company-name	city

Third normal form (3NF)

Supplier-new

s#	sname	company-name

s# → sname

s# → company-name

Minimal key = (s#)

Schema Supplier-new is in 3NF because no attribute is transitively dependent on s#

Third normal form (3NF)

Company

company-name	city

company-name → city

Minimal key = (company-name)

Schema Company is in 3NF because no attribute is transitively dependent on company-name

Third normal form (3NF)

Company

company-name	city

company-name → city

Every relational schema, which consists of at most 2 attributes is always in 3NF

Third normal form (3NF)

Alternative definition

A relational schema R is in 3NF if whenever a functional dependency X → A is valid in R then either:

(1) X is a superkey in R, or

(2) A is a prime attribute in R

Third normal form (3NF)

Supplier

s#	sname	company-name	city

s# → sname

s# → company-name

s# → city

company-name → city

Minimal key = (s#)

Schema Supplier is not in 3NF because

(1) attribute company_name is not a superkey and

(2) attribute city is not prime

Third normal form (3NF)

Location		
city	street	zip-code

city, street → zip-code

zip-code → city

Minimal key$_1$ = (city, street)

Minimal key$_2$ = (zip-code, street)

Schema Location is in 3NF because attribute city used on the right hand side of functional dependency
zip-code → city is prime

Third normal form (3NF)

Location

city	street	zip-code
NY	55	484
NY	56	484
LA	55	473
LA	56	473
LA	57	474

Repetition of [LA ... 473] and [NY ... 484] is forced by functional dependency zip-code → city

Outline

- Third Normal Forms (3NF)
- Boyce-Codd Normal Form (BCNF)

Boyce-Codd normal form (BCNF)

A relational schema R is in BCNF if whenever functional dependency X → A holds in R then:

(1) X is a superkey in R

~~**(2) or A is a prime attribute in R**~~ **(3NF only !!!)**

Boyce-Codd normal form (BCNF)

Location		
city	street	zip-code

city, street $\rightarrow$ zip-code

zip-code $\rightarrow$ city

Minimal key$_1$ = (city, street)

Minimal key$_2$ = (zip-code, street)

Schema Location is not in BCNF because zip-code is not a superkey

zipcode $\rightarrow$ city & zip_code $\not\rightarrow$ street

Boyce-Codd normal form (BCNF)

Location

city	street	zip-code

Schema Location should be decomposed into the following schemas

CS

street	zip-code

CZ

city	zip-code

Boyce-Codd normal form (BCNF)

CS

street zip-code

Minimal key$_2$ = (zip-code, street)

Schema CS is in BCNF because does not exist a functional dependency whose left hand side is not a superkey

Boyce-Codd normal form (BCNF)

CZ

city	zip-code

zip-code → city

Minimal key = (zip-code)

Schema CZ is in BCNF because zip-code is a superkey

Boyce-Codd normal form (BCNF)

CS

street	zip-code

Minimal key = (zip-code, street)

CZ

city	zip-code

zip-code → city

Minimal key = (zip-code)

Every relational schema, which consists of at most 2 attributes is always in BCNF

Boyce-Codd normal form (BCNF)

CS

street	zip-code

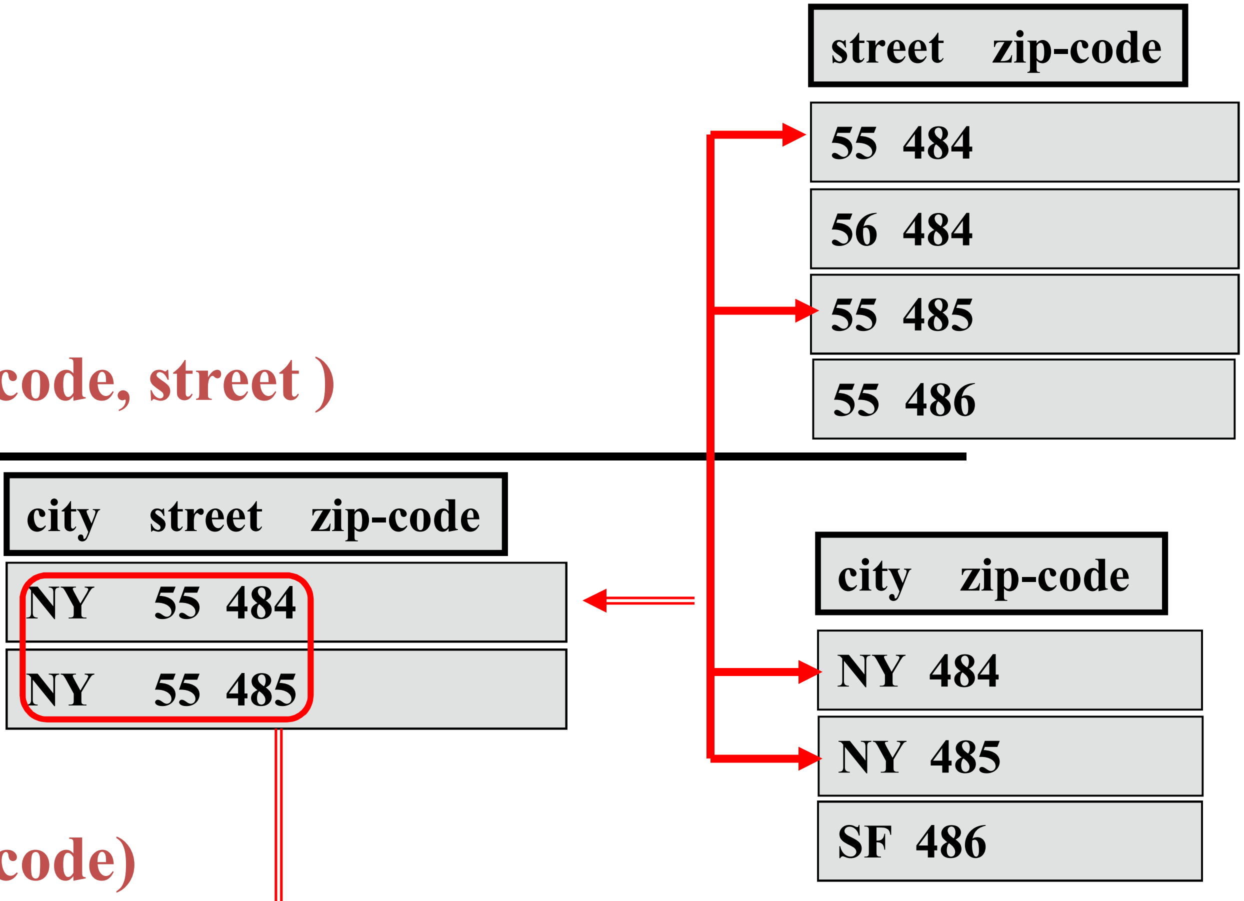
Minimal key = (zip-code, street)

CZ

city	zip-code

zip-code → city

Minimal key = (zip-code)



street	zip-code
55	484
56	484
55	485
55	486

city	street	zip-code
NY	55	484
NY	55	485

city	zip-code
NY	484
NY	485
SF	486

Normalization to BCNF costs a functional dependency

city, street → zip-code

References

Elmasri R., Navathe S. B., *Fundamentals of Database Systems*, chapters 10.3,10.4,10.5

R. Ramakrishnan, J. Gehrke *Database Management Systems*, chapters 19.2, 19.3