**Normal Forms**

**First Normal Form**

o A relation is in 1NF if all attribute values are *atomic*: no repeating group, no composite attributes.

o Formally, a relation may only has atomic attributes. Thus, all relations satisfy 1NF.

Example: Consider the following table. It is not in 1 NF.

DEPT\_NO MANAGER\_NO EMP\_NO NAME

===================================================

D101 12345 30000 Carl Sagan

30001 Magic Johnson

30002 Larry Bird

---------------------------------------------------

D102 13456 40000 Jimmy Carter

40001 Paul Simon

The corresponding relation in 1 NF:

DEPT\_NO MANAGER\_NO EMP\_NO NAME

===================================================

D101 12345 30000 Carl Sagan

D101 12345 30001 Magic Johnson

D101 12345 30002 Larry Bird

D102 13456 40000 Jimmy Carter

D102 13456 40001 Paul Simon

o Problem: relational operations treat attributes as atomic.

o Relations satisfying only 1NF has unnecessary redundancy and anomalies.

**Second Normal Form**

o A relation R is in 2NF if

(a) R is in 1NF, and

(b) all *nonkey* attributes are *fully* dependent on the candidate keys.

o A key attribute appears in a candidate key.

o There is no *partial* dependency in 2NF.

If X -> A and X is a subset of a candidate key K, then X = K.

Example:

The following relation is not in 2NF. (Assume the number of credits of a given course does not change). Note the redundancy and anomalies

. Student\_ID Course Credit Grade

------------------------------------------------------

S10 CSCI 1000 3 A

S10 CSCI 1010 3 B+

S20 CSCI 1000 3 C+

S30 CSCI 1000 3 A-

S30 MATH 1111 1 A

**Third Normal Form**

o (Old definition) A relation R is in 3NF if

(1) R is in 2NF, and

(2) There is no *transitive* dependency of *nonkey* attributes on the keys.

Example 1:

The following relation *may* be in 2NF, but may not be in 3NF.

 EMP\_NO NAME DEPT\_NO MANAGER\_NO

 ------------------------------------------------------

 10000 Paul Simon D123 54321

 12000 Art Garfunkel D123 54321

 13000 Tom Jones D123 54321

 21000 Nolan Ryan D225 42315

 22000 Magic Johnson D225 42315

 31000 Carl Sagan D337 33323

o 3NF does not eliminate all redundancy due to functional dependencies.

Example 2: Consider the relation

S(SUPP#, PART#, SNAME, QUANTITY) with the following assumptions:

(1) SUPP# is unique for every supplier.

(2) SNAME is unique for every supplier.

(3) QUANTITY is the *accumulated* quantities of a part supplied by a supplier.

(4) A supplier can supply more than one part.

(5) A part can be supplied by more than one supplier.

We can find the following non-trivial functional dependencies:

(1) SUPP# --> SNAME

(2) SNAME --> SUPP#

(3) SUPP# PART# --> QUANTITY

(4) SNAME PART# --> QUANTITY

Note that SUPP# and SNAME are *equivalent*.

The candidate keys are:

(1) SUPP# PART#

(2) SNAME PART#

The relation is in 3NF.

Example 3: Consider the relation R(CITY, STREET, ZIP) with the FDs:

(1) CITY STREET --> ZIP, and

(2) ZIP --> CITY.

There are two candidate keys:

(1) CITY STREET, and

(2) ZIP STREET

Hence, all attributes are key attributes and the relation is in both 2NF and 3NF.

**BCNF (Boyce-Codd Normal Form)**

o A relation R is said to be in **BCNF** if for *every* *non-trivial* functional dependency X --> A in R, X is a *superkey*.

Example 1: EMPLOYEE(EMP\_NO, NAME, DEPT\_NO, MANAGER\_NO) with

EMP\_NO --> NAME

EMP\_NO --> DEPT

DEPT\_NO --> MANAGER\_NO

is not in BCNF.

The functional dependency DEPT\_NO --> MANAGER\_NO is

(1) non-trivial, and

(2) DEPT\_NO is not a superkey.

Recall that this is the example we used for illustrating bad design.

Example 2:

We can decompose

EMPLOYEE(EMP\_NO, NAME, DEPT\_NO, MANAGER\_NO) into

EMP(EMP\_NO, NAME, DEPT\_NO) with

EMP\_NO --> NAME

EMP\_NO --> DEPT

and

DEPARTMENT(DEPT\_NO, MANAGER\_NO) with

DEPT\_NO --> MANAGER\_NO

Both relations are in BCNF since

(1) EMP\_NO is a superkey of the relation EMP.

(2) DEPT\_NO is a superkey of the relation DEPARTMENT.

Recall that these are the good relations without the anomalies in the previous example.

Example 3:

 Consider the relation S(SUPP#, PART#, SNAME, QUANTITY) with the following FDs::

(1) SUPP# --> SNAME

(2) SNAME --> SUPP#

(3) SUPP# PART# --> QUANTITY

(4) SNAME PART# --> QUANTITY

The candidate keys of the relation S(SUPP#, PART#, SNAME, QUANTITY) are:

(1) SUPP# PART#

(2) SNAME PART#

S is not in BCNF because, for example, the functional dependency

SUPP# --> SNAME is

(1) non-trivial, and

(2) SUPP# is not a superkey.

To deal with it, we can decompose S(SUPP#, PART#, SNAME, QUANTITY) into

(1) SUPPLIER(SUPP#, SNAME) with

SUPP# --> SNAME

SNAME --> SUPP#

with two candidate keys:

(a) SUPP#

(b) SNAME

(2) ORDER(SUPP#, PART#, QUANTITY) with

SUPP# PART# --> QUANTITY.

Example 4: Consider the relation R(A, B, C, D) with

A --> B, B --> C, C --> A and C --> D.

There are three candidate keys:

(1) A,

(2) B, and

(3) C.

Since every left hand side of any non-trivial functional dependency is a superkey, R is in BCNF.

**Motivation of BCNF**

o The purpose of BCNF is to eliminate any redundancy that functional dependencies can made.

o In a BCNF relation, no value can be predicted from any other, using *only* functional dependencies.

o This is because in a BCNF relation, using functional dependencies only,

(1) any value can only be determined by a superkey, but

(2) the superkey is unique.

o However, there are other type of dependencies.

Therefore, there are higher normal forms.

Example 5:

Consider the relation R(CITY, ZIP, STREET)

Using the code for the postal office, we have

(1) CITY STREET --> ZIP, and

(2) ZIP --> CITY.

Hence, there are two candidate keys:

(1) CITY STREET, and

(2) ZIP STREET

Therefore, R is not in BCNF.

However, if we decompose R into two relations, each with two attributes, then the functional dependency

CITY STREET --> ZIP is *lost*.

Therefore, we better leave the relation alone.

o Sometimes it is not possible to make a relation in BCNF ==> need a less strict normal form (3NF).

**Third Normal Form Revisited**

o Alternative definition of 3NF:

A relation R is said to be in the third normal form if for every *non-trivial* functional dependency X --> A,

(1) X is a superkey, *or*

(2) A is a *prime* (key) attribute.

o An attribute is *prime* (a key attribute) if it appears in a candidate key. Otherwise, it is *non-prime*.

o 3NF cannot eliminate all redundancy due to functional dependencies.

Example 1:

For the relation R(CITY, STREET, ZIP) with

(1) CITY STREET --> ZIP, and

(2) ZIP --> CITY.

Hence, there are two candidate keys:

(1) CITY STREET, and

(2) ZIP STREET

Hence,

(1) CITY is a prime attribute because it appears in the candidate key CITY STREET.

(2) STREET is a prime attribute because it appears in the candidate key CITY STREET.

(3) ZIP is a prime attribute because it appears in the candidate key ZIP STREET.

R is in the 3NF because

(1) For the functional dependency CITY STREET --> ZIP, CITY STREET is a superkey.

(2) For the functional dependency ZIP --> CITY, CITY is a prime attribute.

Example 2:

Reconsider the relation

S(SUPP#, PART#, SNAME, QUANTITY) with

(1) SUPP# --> SNAME

(2) SNAME --> SUPP#

(3) SUPP# PART# --> QUANTITY

(4) SNAME PART# --> QUANTITY

The candidate keys are:

(1) SUPP# PART#

(2) SNAME PART#

The followings are prime attributes:

(1) SUPP#

(2) SNAME

(3) PART#

The following is a non-prime attribute:

(1) QUANTITY

S is in 3NF because

(1) for the functional dependencies (1) and (2), the right hand sides are prime attributes (SNAME and SUPP#).

(2) for the functional dependencies (3) and (4), the left hand sides are superkeys.

Example 3:

Reconsider

EMPLOYEE(EMP\_NO, NAME, DEPT\_NO, MANAGER\_NO) with

EMP\_NO --> NAME

EMP\_NO --> DEPT

DEPT\_NO --> MANAGER\_NO

There is only one candidate key: EMP\_NO.

There is only one prime attribute: EMP\_NO.

EMPLOYEE is not in 3NF because the functional dependency

DEPT\_NO --> MANAGER\_NO is

(1) non-trivial, and

(2) DEPT\_NO is not a superkey, and

(3) MANAGER\_NO is not a prime attribute.

**Normalization Theory Using Functional Dependencies**

o To use the theory on functional dependency:

(1) For a relation of a set of attributes, we analyze the assumptions of the applications.

(2) From the assumptions, we obtain the functional dependencies.

(3) We determine the candidate keys and prime attributes.

(4) If the relation is not in BCNF, we perform decomposition.

(5) If BCNF cannot be satisfied, we aim for 3NF.

**Decomposition**

o Decomposition is a major tool for generating relations satisfying normal forms.

o Decomposition should be disciplined:

(a) More relations may be less efficient in storage.

(b) More relations may be less efficient in executing query.

(c) Some decompositions are harmful:

(i) *Lossy* decompositions.

(ii) Decompositions that do not preserve dependencies.

o Hence, it is important to have *lossless dependency-preserving* decomposition.

**Lossy Decomposition**

Example:

Consider the relation EMP(EMP\_NO, DEPT, MGR\_NO) with

EMP\_NO --> DEPT

DEPT --> MGR\_NO

Note that we do not have MGR\_NO --> DEPT, since one manager can manage more than one department under the assumptions made.

 EMP\_NO DEPT MGR\_NO

 ---------------------------------

 12345 ACCT 90000

 20000 PR 90000

 30000 CAD 95000

The relation is not in BCNF because of the FD

DEPT --> MGR\_NO

Suppose we decompose the relation into

EMP1(EMP\_NO, MGR\_NO)

DEPT(DEPT, MGR\_NO)

They are obtained by projections from EMP:

 EMP1: DEPT:

 EMP\_NO MGR\_NO DEPT MGR\_NO

 ------------------- ---------------------

 12345 90000 ACCT 90000

 20000 90000 PR 90000

 30000 95000 CAD 95000

o If we do not *loss* any information by the decomposition, we should get the original relation from the natural join.

However, EMP1 |x| DEPT:

 EMP\_NO DEPT MGR\_NO

 ---------------------------------

 12345 ACCT 90000

 **20000 ACCT 90000**

 **12345 PR 90000**

 20000 PR 90000

 30000 CAD 95000

This is not the same as the original relation EMP:

Hence, the decomposition of EMP(EMP\_NO, DEPT, MGR\_NO) into

EMP1(EMP\_NO, MGR\_NO) and

DEPT(DEPT, MGR\_NO)

is lossy. It is not a good decomposition.

**Lossless Decomposition**

Example:

Consider now the following decomposition of EMP(EMP\_NO, DEPT, MGR\_NO):

EMP2(EMP\_NO, DEPT) and

EMP3(EMP\_NO, MGR\_NO)

We have EMP2 and EMP3:

 EMP2: EMP3:

 EMP\_NO DEPT EMP\_NO MGR\_NO

 ------------------- ---------------------

 12345 ACCT 12345 90000

 20000 PR 20000 90000

 30000 CAD 30000 95000

Hence, EMP2 |x| EMP3:

 ENP\_NO DEPT MGR\_NO

 ---------------------------------

 12345 ACCT 90000

 20000 PR 90000

 30000 CAD 95000

This is exactly the same as the original relation EMP. Therefore, the decomposition does not loss any information. It is a *lossless* decomposition.

**Theory of Lossless Decompositions**

Example:

Why is the decomposition of EMP(EMP\_NO, DEPT, MGR\_NO) into

(1) EMP1(EMP\_NO, MGR\_NO) and

DEPT(DEPT, MGR\_NO) *lossy*, and

(2) EMP2(EMP\_NO, DEPT) and

EMP3(EMP\_NO, MGR\_NO) *lossless*?

**Theorem**: Suppose R(X, Y, Z) is decomposed into R1(X, Y) and R2(X, Z). X is the set of common attributes in R1 and R2. The decomposition is lossless if and only if

(a) X --> Y, *or*

(b) X --> Z.

Example:

In case (1), X is MGR\_NO, Y is EMP\_NO, Z is DEPT.

None of condition (a) or (b) is satisfied. Hence, (1) is lossy.

In case (2), X is EMP\_NO, Y is DEPT, Z is MGR\_NO.

Both conditions (a) and (b) are satisfied. Hence, (2) is lossless.

**Dependency-Preserving Decompositions**

Example: For the relation EMP(EMP\_NO,DEPT,MGR\_NO) with

EMP\_NO --> DEPT

DEPT --> MGR\_NO,

The decomposition of EMP into

EMP2(EMP\_NO, DEPT) and

EMP3(EMP\_NO, MGR\_NO)

is lossless, but it does not *preserve dependencies*:

the FD DEPT --> MGR\_NO

cannot be enforced *by any relation*.

For example, if we add the information EMP 23000 work in the ACCT department under manager 97000 and are not careful, we may have:

 EMP2: EMP3:

 EMP\_NO DEPT EMP\_NO MGR\_NO

 ------------------- ---------------------

 12345 ACCT 12345 90000

 20000 PR 20000 90000

 30000 CAD 30000 95000

 **23000 ACCT 23000 97000**

The FD DEPT --> MGR\_NO is violated.

Thus, for the relation EMP(EMP\_NO,DEPT,MGR\_NO) with

EMP\_NO --> DEPT

DEPT --> MGR\_NO,

the best decomposition is into

EMP1(EMP\_NO, DEPT) and

DEPT(DEPT, MGR\_NO)

It is easy to show that, the decomposition is lossless, preserves dependencies, and that

EMP1 and DEPT are both in BCNF.

**Teaching Notes:**

o Give the anomalies that NFNF relation has: e.g. difficulty of finding which department Magic Johnson is working for.

o Give an example of a key attribute that is in a candidate key, but not in a primary key.

o Discuss the anomalies of the relation not in 2NF in the example.

o For each example, show the assumptions, FD's, candidate keys and highest normal form the relation is in.

o The relation of Example 1 of BCNF is also not in 3NF.

o Note that lossy decomposition is usually *gainly*.