Normalization

Anne Denton

Department of Computer Science
North Dakota State University

Outline

1. Normal Forms Introduction
   - Background
   - Second normal form
   - Third normal form
2. Normalization Considerations
   - Example
   - Boyce-Codd Normal Form
   - Decompositions
3. Fourth Normal Form
   - Example 4NF problem
   - Definitions

Table of Contents

History

- The history of normalization goes back to beginnings of relational databases
- Normal forms were proposed in 1972 by Codd
- The normalization process takes a relation through a series of steps
- Each one intended to eliminate designs that can cause problems
- First normal form
    - All attributes have to be atomic
    - We have already included that into our definition of a relation
- A relation is in first normal form by definition

Example

- Let us consider a mail order database

cid	c_name	oid	o_date	pid	i_count	wh

- The oid is unique for each order
- A customer can place many orders
- Many different parts can be ordered in one order
- Many instances of the same part can be ordered in one order (i_count).
- A part will be stored in one warehouse only

Functional dependencies

- What functional dependencies are expected to hold?
  - cid → c_name
  - oid → cid
  - oid → o_date
  - pid → wh
  - oid, pid → i_count
- What functional dependencies can be inferred?
  - oid -→ oid (reflexive rule)
  - oid → c_name (transitive rule)
  - oid, pid → cid, c_name, oid, o_date, pid, wh, i_count (combination of rules)

Candidate keys

- Candidate keys are attribute that functionally determine all others in the relation
- Determining candidate keys is important for normalization
  - Traditionally normalization also expected that one primary key be determined
  - Difference only matters in very rare cases that are becoming even rarer
- Consider the mail order example
  - We found that: oid, pid → cid, c_name, oid, o_date, pid, wh, i_count
  - Neither oid → pid nor pid → oid
  - Combination of oid and pid is only candidate key
  - In the following, we assume that a primary key has been selected
  - Components of the primary key will be underlined

Table of Contents

Second normal form

- The second normal form (2NF) is based on the concept of full functional dependency
- Definition of a full functional dependency
    - If X and Y are attributes of a relation, Y is fully functionally dependent on X if Y is functionally dependent on X but not on a subset of X
- Definition of 2NF
    - A relation (in first normal form) is in second normal form if every non-primary-key attribute is fully functionally dependent on the primary key

Question 1 (Multiple correct answers possible)

If X, Y → Z and X → Z

1. Then Z is fully functionally dependent on X, Y
2. Then Z is partially functionally dependent on X, Y
3. Then more information is needed to determine if Z is fully functionally dependent on X, Y

Question 2 (multiple answers can be correct)

Taking the example of the below mail-order database.

cid	c_name	oid	o_date	pid	i_count	wh

- cid → c_name
- oid → cid
- oid → o_date
- pid → wh
- oid, pid → i_count

Which of the following is a full functional dependency?

1. oid, pid -> o_date
2. oid, pid -> wh
3. oid, pid -> i_count

Question 3 (multiple answers can be correct)

Consider the mail order example

cid	c_name	oid	o_date	pid	i_count	wh

- cid → c_name
- oid → cid
- oid → o_date
- pid → wh
- oid, pid → i_count

What are candidate keys?

1. oid, pid
2. oid
3. pid

Partial functional dependencies on the key

- The example database shows partial functional dependencies on the key oid, pid
  - oid → cid, c_name, o_date,
  - pid → wh
- Only the functional dependency
  - oid, pid → i_count

  is a full functional dependency on the key

Question 4

Consider the mail order example

cid	c_name	oid	o_date	pid	i_count	wh

Is the relation in 2NF?

Decomposition into 2NF for mail order example

- Create a relation for each of the parts of the primary key
- Attributes that depend on oid form one table

cid	c_name	oid	o_date

- Attributes that depend on pid form a second table

pid	wh

- Original table remains except for those attributes that had partial functional dependencies on key

oid	pid	i_count

- Note that the last table would be necessary even if no attribute had a full functional dependency on the key!

Table of Contents

Third normal form

- The third normal form (3NF) is based on the concept of transitive dependency
- Definition of a transitive dependency
  - If X, Y, and Z are sets of attributes of a relation then Z is transitively functionally dependent on X via Y if the following both hold: $X \rightarrow Y$ and $Y \rightarrow Z$
  - X is not functionally dependent on Y or Z
- Definition of 3NF
  - A relation (in first normal form) is in third normal form if it is in second normal form, and no non-primary-key attribute is transitively dependent on the primary key.

Question 5

X → Z is a transitive functional dependency if X → Y and Y → Z and the following also holds

1. Z does not functionally depend on X
2. Y does not functionally depend on Z
3. X does not functionally depend on Y or Z

Question 6

Consider the order table in the mail order example

cid	c_name	oid	o_date

- cid → c_name
- oid → cid
- oid → o_date

Is this relation in 3NF?

Example of a transitive FD on the primary key

- Consider order table in the mail order example

cid	c_name	oid	o_date

  - oid → o_date
  - oid → cid
  - cid → c_name
- Conditions for transitive FDs applied to this
  - oid → cid and cid → c_name
  - oid is not functionally dependent on either cid or c_name

Decomposition into 3NF

- Decompose by storing transitively dependent attributes in a separate table
- If X $\rightarrow$ Y and Y $\rightarrow$ Z,
  and X not functionally dependent on Y or Z
- Decompose into

X	Y

and

Y	Z

Decomposition for the mail order example

- Consider order table in the mail order example

cid	c_name	oid	o_date

- Decompose into

oid	cid	o_date

and

cid	c_name

Result of decomposition

The result of the decomposition in 3NF of the example database

cid	c_name	oid	o_date	pid	i_count	wh

is hence

oid	cid	o_date

cid	c_name

pid	wh

oid	pid	i_count

Humorous alternative definition of 3rd Normal Form

Every nonkey attribute must functionally depend upon

the key	1NF
the whole key	2NF
nothing but the key	3NF

so help you Codd

Table of Contents

1. Normal Forms Introduction
   - Background
   - Second normal form
   - Third normal form
2. Normalization Considerations
   - Example
   - Boyce-Codd Normal Form
   - Decompositions
3. Fourth Normal Form
   - Example 4NF problem
   - Definitions

Example: Room rental

Consider the following database of center that rents out rooms for special occasions

eid	rid	e_date	cid	c_name	c_tel	seats	sid	s_name

- Each renting event is identified by an event id (eid) and has one associated room id (rid) and a start date/time (e_date)
- Each room has a maximum number of seats
- Customer information includes an identifier (cid), name (c_name), and telephone number (c_tel)
- An event can involve multiple staff members, who are identified by their respective sid, and have a name (s_name)
- Note that if a customer rents multiple rooms on the same day, those will be considered separate renting events

Functional dependencies for room rental example

Some functional dependencies are relatively straightforward

- eid → e_date
- rid → seats
- cid → c_name
- sid → s_name

Others may be less clear, and will be considered next

Question 7 (Multiple answers can be correct)

What functional dependencies are likely to hold over the below relation? (Each renting event has one associated room id (rid) and one customer id (cid); An event can involve multiple staff members; Renting multiple rooms will be separate renting events)

eid	rid	e_date	cid	c_name	c_tel	seats	sid	s_name

1. eid → rid
2. rid → eid
3. eid → cid
4. cid → eid
5. eid → sid
6. sid → eid

Question 8 (Multiple answers can be correct)

What would be a candidate key of the relation (eid → e_date; rid → seats; cid → cname; sid → s_name; eid → cid; eid → rid)

eid	rid	e_date	cid	c_name	c_tel	seats	sid	s_name

1. eid
2. rid
3. cid
4. sid
5. eid rid cid sid
6. eid rid sid
7. eid sid

Alternate keys

- One could argue that in any one room, at any one time, only one event can be held
- That suggests the functional dependency {rid, e_date} → eid
- There would then be two candidate keys {eid, sid} and {rid, e_date, sid}
- The 2NF and 3NF definitions assume that a primary key has been selected
    - If {rid, e_date, sid} is selected decomposition would maintain {rid, e_date} as composite key of the relation it is part of
    - Hence {eid, sid} is preferable and will be used in the following
- Note that if {rid, e_date, sid} were selected, it would not create additional transitive functional dependencies on the key
- The definition of transitive FDs ($X \to Y$ and $Y \to Z$) includes that neither $Y \to X$ nor $Z \to X$

Question 9 (Multiple answers can be correct)

Given the below room rental database, which functional dependencies are partial functional dependencies on the key?

eid	rid	e_date	cid	c_name	c_tel	seats	sid	s_name

1. eid → e_date
2. rid → seats
3. cid → c_name
4. sid → s_name
5. eid → rid
6. eid → cid

Room rental database in 2NF

- Attributes that depend on eid (extracted because of partial dependence on key)

eid	rid	e_date	cid	c_name	c_tel	seats

- Attributes that depend on sid (extracted because of partial dependence on key)

sid	s_name

- Original relation from which all those attributes have been extracted that had a partial dependence on the key

eid	sid

Question 10 (Multiple answers can be correct)

Given the below event relation of the room rental database, which of the following attributes has a transitive functional dependencies on the key?

eid	rid	e_date	cid	c_name	c_tel	seats

1. e_date
2. seats
3. c_name
4. rid
5. cid

Room rental database in 3NF

Normalization into 3NF of the following room rental database:

eid	rid	e_date	cid	c_name	c_tel	seats	sid	s_name

Results in the relations:

eid	e_date	rid	cid

rid	seats

cid	c_name	c_tel

sid	s_name

eid	sid

Question 11 (Multiple answers can be correct)

A relation is in Second Normal Form if it is in First Normal Form and no non-primary-key attribute

1. Has a full functional dependency on the primary key
2. Has a partial functional dependency on the primary key
3. Has a transitive functional dependency on the key
4. Has a trivial functional dependency on the key

Question 12 (Multiple answers can be correct)

A relation is in Third Normal Form if it is in Second Normal Form and no non-primary-key attribute

1. Has a full functional dependency on the primary key
2. Has a partial functional dependency on the primary key
3. Has a transitive functional dependency on the key
4. Has a trivial functional dependency on the key

Table of Contents

Boyce-Codd Normal Form

- Boyce-Codd Normal Form was meant as a different formulation of the 3NF but turned out to be stricter
- Based on the realization that partial and transitive functional dependencies share that the dependency is not on a superkey
- It is defined independent of the choice of primary key
  - All candidate keys are considered equivalent
  - No exemptions are made for attributes that are part of a primary key
- Only differs from 3NF under unusual circumstances

Definition of Boyce-Codd Normal Form

With

R a relation schema (in 1NF)
X a subset of attributes of R
A a single attribute of R

R is in BCNF if for every FD X → A that holds over R, one of the following statements is true

- X $\supset$ A, i.e., it is a trivial FD
- X is a superkey

Question 13

Is the decomposition of the mail order database in BCNF?

oid	cid	o_date

cid	c_name

pid	wh

oid	pid	i_count

Question 14

Is the decomposition of the room rental database in BCNF?

eid	e_date	rid	cid

rid	seats

cid	c_name	c_tel

sid	s_name

eid	sid

Difference between 3NF and BCNF

- The following relation is the example that is commonly given

city	street_address	postal_code

- Functional dependencies are typically listed as
  - city street_address → postal_code
  - postal_code → city
- Importance (or not)
  - Because city is part of the primary key, 3NF is not violated but BCNF is
  - Note that the functional dependencies do not hold in North Dakota where many towns may share a postal code
  - Note also that the prevalence of use of surrogate keys has let this rare case pretty much disappear

Table of Contents

1. Normal Forms Introduction
   - Background
   - Second normal form
   - Third normal form
2. Normalization Considerations
   - Example
   - Boyce-Codd Normal Form
   - Decompositions
3. Fourth Normal Form
   - Example 4NF problem
   - Definitions

Problems with Decompositions

- The motivation for decompositions was to eliminate uncontrolled redundancy
- The decomposition process itself may introduce problems
- One concern is that key constraints assume that functional dependencies are represented within one table
    - Decompositions should be dependency preserving
- Another concern is that recreating the combined table should be possible
    - Lossless join property

Dependency preservation

- In general, a decomposition should not lead to loss of a functional dependency
- Consider the mail order database used earlier

cid	c_name	oid	o_date	pid	i_count	wh

- The functional dependency oid → cid violates BCNF, since oid is not a candidate key
- The only key is oid pid
- However, decomposing by creating a relation

oid	cid

would leave

c_name	oid	o_date	pid	i_count	wh

which lacks the FD cid → c_name

Dependency preserving decompositions

- It is always possible to find a decomposition into 3NF that is dependency preserving
- From a procedural perspective:
  - Decomposing into 2NF first and only then considering transitive FDs helps avoid mistakes
- It is not always possible to find a decomposition into BCNF that is dependency preserving, regardless of procedure
  - Typical counter-example is the case of a primary key attribute that depends on a non-primary-key attribute
  - Should not normally be decomposed (see postal-code example), which is very rare
- Usually there is no difference between BCNF and 3NF and both can and should be dependency-preserving

Question 15 (Multiple answers can be correct)

Dependency preservation of decompositions means that

1. Stating a minimal set of functional dependencies does not require joining the decomposed relations
2. The relation was only decomposed into 3NF not BCNF
3. When joining the decomposed relations there will be no spurious tuples created
4. When joining the decomposed relations the original universal relation can be recreated

Lossless join property

- Joining of tables can create spurious tuples
- Note: The term "lossless join" can be misleading:
  - We won’t get fewer tuples through a lossy join but more that are incorrect
- Consider the decomposition of the following relation

cid	c_name	oid	o_date

- Assume a decomposition into

cid	c_name

and

c_name	oid	o_date

Lossless join example

- Now assume that there are two customers John Smith

cid	c_name
111	John Smith
567	John Smith

- With the order information

c_name	oid	o_date
John Smith	654321	8-Mar-20

- Joining both tables associates both with the order of one:

cid	c_name	oid	o_date
111	John Smith	654321	8-Mar-00
567	John Smith	654321	8-Mar-00

Losslessness Criterion

- All decompositions must be lossless!
- It is always possible to find a lossless decomposition
- Rule to insure lossless-join decomposition
  - The decomposition of relation R into R1 and R2 is lossless if and only if the attributes common to R1 and R2 contain a key for either R1 or R2
- You can see that primary key - foreign key combinations fulfill this

Question 16 (Multiple answers can be correct)

The lossless join property of decompositions means that

1. Stating a minimal set of functional dependencies does not require joining the decomposed relations
2. The relation was only decomposed into 3NF not BCNF
3. When joining the decomposed relations there will be no spurious tuples created
4. When joining the decomposed relations the original universal relation can be recreated

Decomposition into BCNF

Some considerations ahead of time:

- Dependency preservation has to be considered at every step
- Lossless join property is guaranteed
- Not recommended in the rare cases in which BCNF differs from 3NF
  - In cases where a non-primary-key attribute functionally determines a primary-key attribute, decomposition is usually not desirable
  - Not a common occurrence, as discussed earlier

Decomposition algorithm into BCNF

Suppose that R is not in BCNF
With

- X a subset of attributes of R
- A a single attribute of R
- X → A a FD that causes a violation of BCNF

Decompose R into R-A and XA

- If either R-A or XA is not in BCNF, decompose them further through a recursive application of this algorithm

Decomposition algorithms

- If several functional dependencies violate BCNF, the described algorithm can lead to different collections of BCNF relations
    - Decomposition algorithms are not deterministic in general
- Decomposition algorithms rely on the specification of all functional dependencies
- Post-processing may be needed to consolidate large number of small tables
    - Any attributes that depend on the same key can be combined

Table of Contents

Example 4NF problem

- Consider the textbook used in class
    - It has two authors
    - It is used in two classes
- Assume that a single relation represents the information on authors and classes
    - Which is clearly not what is recommended based on the design principles we discussed

Book	Author	Course
0-8053-1755-4	Elmasi	CSci 366
0-8053-1755-4	Navathe	CSci 366
0-8053-1755-4	Elmasi	CSci 765

- Every combination of Author and Course needed
- Does not violate any constraints on functional dependencies

Multi-valued dependencies (MVD)

- 4NF defined based on the concept of multi-valued dependencies
- Informal definition:
  The multivalued dependency X $\rightarrow\rightarrow$ Y holds if
  - For a selection on X=x (any value of x) the following must hold
  - The projection on anything other than X is a cross product of the projection on Y and the projection of R-Y
- Every FD is also a MVD, because there is only a single record with X=x

Table of Contents

Formal definition of a MVD

Let R be a relation schema and let X and Y be subsets of attributes of R.
The multivalued dependency X→→Y holds on R if in any legal relation r(R) the following holds:
For all pairs of tuples and t1 and t2 in R such that t1[X] = t2[X], there exist tuples t3 and t4 such that

- t1[X] = t2[X] = t3[X] = t4[X]
- t1[Y] = t3[Y]
- t2[R - Y] = t3[R - Y]
- t2[Y] = t4[Y]
- t1[R - Y] = t4[R - Y]

Fourth Normal Form

With

- R a relation schema (in 1NF)
- X and Y subsets of R
- R is in 4NF if it is in 3NF and for every MVD X $\rightarrow\rightarrow$ Y that holds over R, one of the following statements is true:
  - The MVD is trivial, i.e. Y is a subset of X or XY = R
  - X is a superkey

Properties of 4NF

- Decomposition into 4NF
  - A relation XYZ that has a non-trivial functional dependency X$\rightarrow\rightarrow$Y is decomposed into XY and XZ
- 4NF violations typically happen for all-key relations, i.e. relations where all attributes together form the primary key
- If a relation schema is in BCNF, and at least one of its keys consists of a single attribute, it is also in 4NF

Question 17 (Multiple answers can be correct)

Which of the following statements about 4NF are correct

1. A relation in 4NF does not have multi-valued dependencies
2. A relation is in 4NF if it is in 3NF and any of its multi-valued dependencies X→→Y are either trivial (Y is a subset of X or XY = R) or X is a superkey.
3. The following two conditions together ensure that a relation is in 4NF: It has an atomic key, and for all non-trivial FDs X→Y it is true that X is a superkey