

CIS560

Obtaining a Good Database Design

KANSAS STATE
UNIVERSITY

Computer Science



1

As we've seen, bad design...

- Lacks data integrity
- Decreases maintainability
- Performs poorly
- Introduces data anomalies
 - Redundancy
With redundant data, what's the truth?
 - Update anomalies
We have to update multiple places and may miss some.
 - Delete anomalies
We could unintentionally lose data.

KANSAS STATE
UNIVERSITY

Computer Science



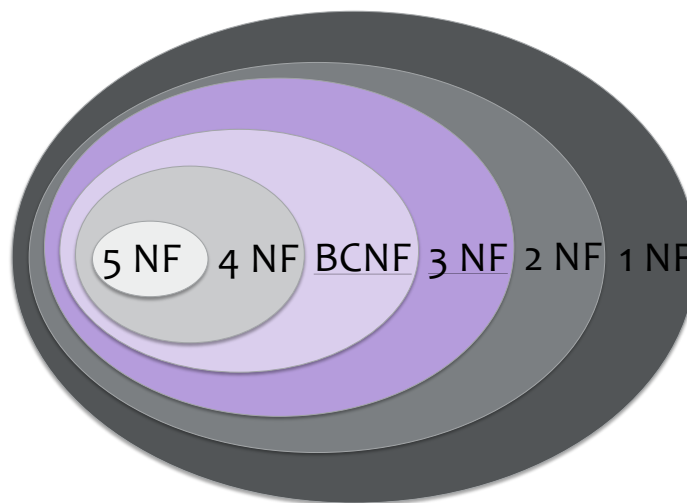
2

How do we achieve good design?

- Database researchers developed normal forms.
- If a database is in one of the normal forms, then it is guaranteed to have certain properties.
- The most important, and most common, are:
 - Third Normal Form (3NF)
 - Boyce-Codd Normal Form (BCNF)
- 3NF and BCNF guarantee to avoid certain types of redundancy.



Normal Forms



Normal forms are defined using...

Keys and Functional Dependencies



Functional Dependencies

- They are a form of constraint.
- Finding them is a part of database design.
- Also used in normalizing the relations.
 1. Start with some relational schema.
 2. Find the functional dependencies.
 3. Use them to design a better schema.



Functional Dependencies Defined

- If two tuples agree on the attributes

$$A_1, A_2, \dots, A_n$$

then they must agree on the attributes

$$B_1, B_2, \dots, B_m$$

- Formally

$$A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$$

- Informally

If we know A_1, A_2, \dots, A_n , then we know B_1, B_2, \dots, B_m



When Does a Functional Dependency Hold?

Definition: $A_1, \dots, A_n \rightarrow B_1, \dots, B_m$ holds in S if:

$$\forall t, t' \in S, (t.A_1=t'.A_1 \wedge \dots \wedge t.A_n=t'.A_n \Rightarrow t.B_1=t'.B_1 \wedge \dots \wedge t.B_m=t'.B_m)$$

S

	A_1	...	A_n		B_1	...	B_m		
t									
t'									

if t, t' agree here
then t, t' agree here

Example

A FD holds, or does not hold on an instance of a relation:

StudID	Name	Phone	Dept
0045	Smith	1234	Math
3542	Mike	9876	CIS
1111	Smith	9876	CIS
9999	Mary	1234	Eng

StudID \rightarrow Name, Phone, Dept

Dept \rightarrow Phone



Example

A FD holds, or does not hold on an instance:

StudID	Name	Phone	Dept
0045	Smith	1234	Math
3542	Mike	9876	CIS
1111	Smith	9876	CIS
9999	Mary	1234	Eng

StudID \rightarrow Name, Phone, Dept

Dept \rightarrow Phone



Example

A FD holds, or does not hold on an instance of a relation:

StudID	Name	Phone	Dept
0045	Smith	1234	Math
3542	Mike	9876	CIS
1111	Smith	9876	CIS
9999	Mary	1234	Eng

StudID \rightarrow Name, Phone, Dept

Phone \rightarrow Dept

Dept \rightarrow Phone



Example

A FD holds, or does not hold on an instance:

StudID	Name	Phone	Dept
0045	Smith	1234	Math
3542	Mike	9876 \rightarrow	CIS
1111	Smith	9876 \rightarrow	CIS
9999	Mary	1234	Eng

StudID \rightarrow Name, Phone, Dept

Phone \rightarrow Dept

Dept \rightarrow Phone



Example

A FD holds, or does not hold on an instance:

StudID	Name	Phone	Dept
0045	Smith	1234 →	Math
3542	Mike	9876	CIS
1111	Smith	9876	CIS
9999	Mary	1234 →	Eng

StudID → Name, Phone, Dept

Dept → Phone

~~Phone → Dept~~

Example

FD's are constraints

- On some instances they hold
- On others they don't

name → color
category → store
color, category → price

Do all the above functional dependencies hold on this instance?

name	category	color	store	price
iPad	Gadget	Silver	Campus store	529
iPhone	Gadget	Silver	Campus store	429

Example

FD's are constraints

- On some instances they hold
- On others they don't

$\text{name} \rightarrow \text{color}$
 $\text{category} \rightarrow \text{store}$
 $\text{color, category} \rightarrow \text{price}$

Do all the above functional dependencies hold on this instance?

name	category	color	store	price
iPad	Gadget	Silver	Campus store	529
iPhone	Gadget	Black	Campus store	429
iPad	Tablet	Silver	Best Buy	569

15

Class Exercise

If all these FDs hold:

$\text{name} \rightarrow \text{color}$
 $\text{category} \rightarrow \text{store}$
 $\text{color, category} \rightarrow \text{price}$

Then this FD also holds:

$\text{name, category} \rightarrow \text{price}$

Why?

16

Anomalies

Anomalies occur when “bad” FDs hold

- We know some of the FDs
- Need to find *all* FDs
- Then we can look for the bad ones



How do we find all Functional Dependencies?

Armstrong's Rules



Armstrong's Rules (1/3)

Splitting rule and Combining rule

$$A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$$

Is equivalent to

$$\begin{array}{l} A_1, A_2, \dots, A_n \rightarrow B_1 \\ A_1, A_2, \dots, A_n \rightarrow B_2 \\ \dots \dots \dots \\ A_1, A_2, \dots, A_n \rightarrow B_m \end{array}$$

19

Armstrong's Rules (2/3)

Trivial Rule

$$A_1, A_2, \dots, A_n \rightarrow A_i$$

where $i = 1, 2, \dots, n$

In other words:

$$A, B, C \rightarrow A$$

$$A, B, C \rightarrow B$$

$$A, B, C \rightarrow C$$

20

Armstrong's Rules (3/3)

Transitive Closure Rule

If $A_1, A_2, \dots, A_n \rightarrow B_1, B_2, \dots, B_m$

and $B_1, B_2, \dots, B_m \rightarrow C_1, C_2, \dots, C_p$

then $A_1, A_2, \dots, A_n \rightarrow C_1, C_2, \dots, C_p$

21

Example

Start from the following FDs:

1. name \rightarrow color
2. category \rightarrow store
3. color, category \rightarrow price

Infer the following FDs:

Inferred FD	Which Rule did we apply ?
4. name, category \rightarrow name	Trivial Rule
5. name, category \rightarrow color	Transitivity on 4, 1
6. name, category \rightarrow category	Trivial Rule
7. name, category \rightarrow color, category	Split/combine on 5, 6
8. name, category \rightarrow price	Transitivity on 7, 3

22

THIS IS TOO HARD!

Let's see an easier way.

23

Closure of a Set of Attributes

Given a set of attributes A_1, \dots, A_n

the **closure** $\{A_1, \dots, A_n\}^+$ = the set of attributes B
such that $A_1, \dots, A_n \rightarrow B$

Example:

name \rightarrow color
category \rightarrow store
color, category \rightarrow price

What are the closures?

name⁺ = {name, color}

{name, category}⁺ = {name, category, color, store, price}

color⁺ = {color}

24

Closure Algorithm

$X = \{A_1, \dots, A_n\}$.

Repeat until X doesn't change:

if $B_1, \dots, B_n \rightarrow C$ is a FD **and**
 B_1, \dots, B_n are all in X
then add C to X.

Example:

name \rightarrow color
 category \rightarrow store
 color, category \rightarrow price

$\{\text{name, category}\}^+ =$
 $\{\text{name, category, color, store, price}\}$

25

Closure Algorithm

$X = \{A_1, \dots, A_n\}$.

Repeat until X doesn't change:

if $B_1, \dots, B_n \rightarrow C$ is a FD **and**
 B_1, \dots, B_n are all in X
then add C to X.

Example:

name \rightarrow color
 category \rightarrow store
 color, category \rightarrow price

$\{\text{name, category}\}^+ =$
 $\{\text{name, category, color, store, price}\}$

Hence: name, category \rightarrow color, store, price

26

Class Exercise

$R(A,B,C,D,E,F)$

$A, B \rightarrow C$
$A, D \rightarrow E$
$B \rightarrow D$
$A, F \rightarrow B$

Compute $\{A,B\}^+$ $X = \{A, B, C, D, E\}$

Compute $\{A, F\}^+$ $X = \{A, F, B, C, D, E\}$

27

Why Do We Need Closure?

- With closure we can find all Functional Dependencies
- You can check if $X \rightarrow A$
 - Compute X^+
 - Check if $A \in X^+$

28

Class Exercise

29

Using Closure to Infer ALL FDs

Relation: $R(A, B, C, D)$

Functional
Dependencies:

A, B	$\rightarrow C$
A, D	$\rightarrow B$
B	$\rightarrow D$

Step 1: Compute X^+ , for every X :

$A^+ = A, B^+ = BD, C^+ = C, D^+ = D$ $AB^+ = ABCD, AC^+ = AC, AD^+ = ABCD,$ $BC^+ = BCD, BD^+ = BD, CD^+ = CD$ $ABC^+ = ABD^+ = ACD^+ = ABCD$ (no need to compute – why?) $BCD^+ = BCD, ABCD^+ = ABCD$

Step 2: Find all FD's $X \rightarrow Y$, s.t. $Y \subseteq X^+$ and $X \cap Y = \emptyset$:

$B \rightarrow D, AB \rightarrow CD, AD \rightarrow BC, BC \rightarrow D,$ $ABC \rightarrow D, ABD \rightarrow C, ACD \rightarrow B$

30