Introduction to Data Management CSE 344

Lecture 18:
Design Theory Wrap-up

Announcements

- WQ6 is due on Tuesday
- Homework 6 is due on Thursday
  - Be careful about your remaining late days.

- Today:
  - Midterm review
  - Review design theory (FD, BCNF) – 3.3.3, 3.3.4, 3.4.2

Midterm Review

- Midterm is graded: mean ≈ 54, median ≈ 55
- Solution is uploaded
  - Read the solutions
  - If you find a new solution/idea (or a bug), you should post it on the discussion board

Lessons for the final

- Do not panic
  - you know everything
- Attempt all questions (important)
  - Write partial solutions to get partial credit
- Do not get stuck on one question
  - there may be easier questions later
- Most important – make sure that you understand all concepts covered in class.
  - Don’t miss lectures/sections
  - Ask questions in class/office hours/discussion board
  - Book/notes won’t help much in the exam.
    You need to think to get a solution

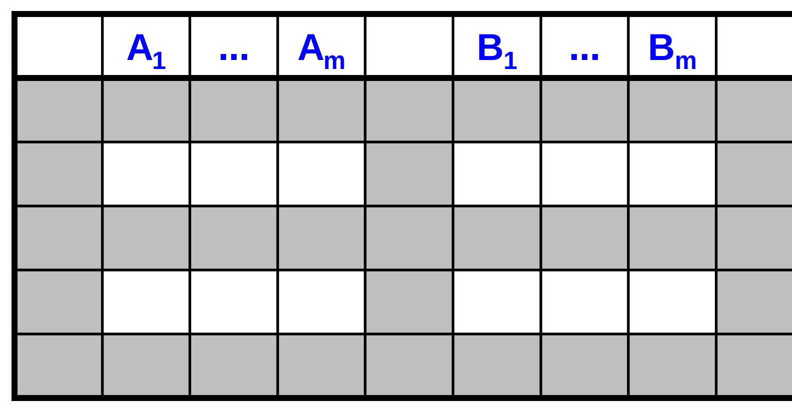
Armstrong’s Rules (1/3)

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

Splitting rule and Combing rule

Is equivalent to

$A_1, A_2, \ldots, A_n \rightarrow B_1$
$A_1, A_2, \ldots, A_n \rightarrow B_2$
. . . . .
$A_1, A_2, \ldots, A_n \rightarrow B_m$

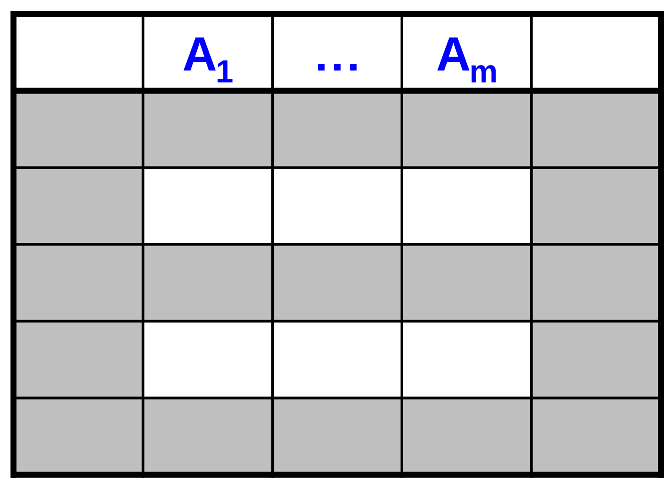


Armstrong's Rules (2/3)

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

Trivial Rule

where i = 1, 2, ..., n



Why ?

Armstrong's Rules (3/3)

Transitive Rule

If $\boxed{A_1, A_2, \ldots, A_n \rightarrow B_1, B_2, \ldots, B_m}$

and $\boxed{B_1, B_2, \ldots, B_m \rightarrow C_1, C_2, \ldots, C_p}$

then $\boxed{A_1, A_2, \ldots, A_n \rightarrow C_1, C_2, \ldots, C_p}$

R(A,B,C,D)

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

Review: BCNF

R(A,B,C,D)

R(A,B,C,D)

subset

Review: BCNF

A → B
B → C

Recall: find X s.t.
$X \subsetneq X^+ \subsetneq$ [all-attrs]

R(A,B,C,D)

R(A,B,C,D)

subset

Review: BCNF

A → B
B → C

Recall: find X s.t.
$X \subsetneq X^+ \subsetneq$ [all-attrs]

R(A,B,C,D)
$A^+ = ABC \neq ABCD$

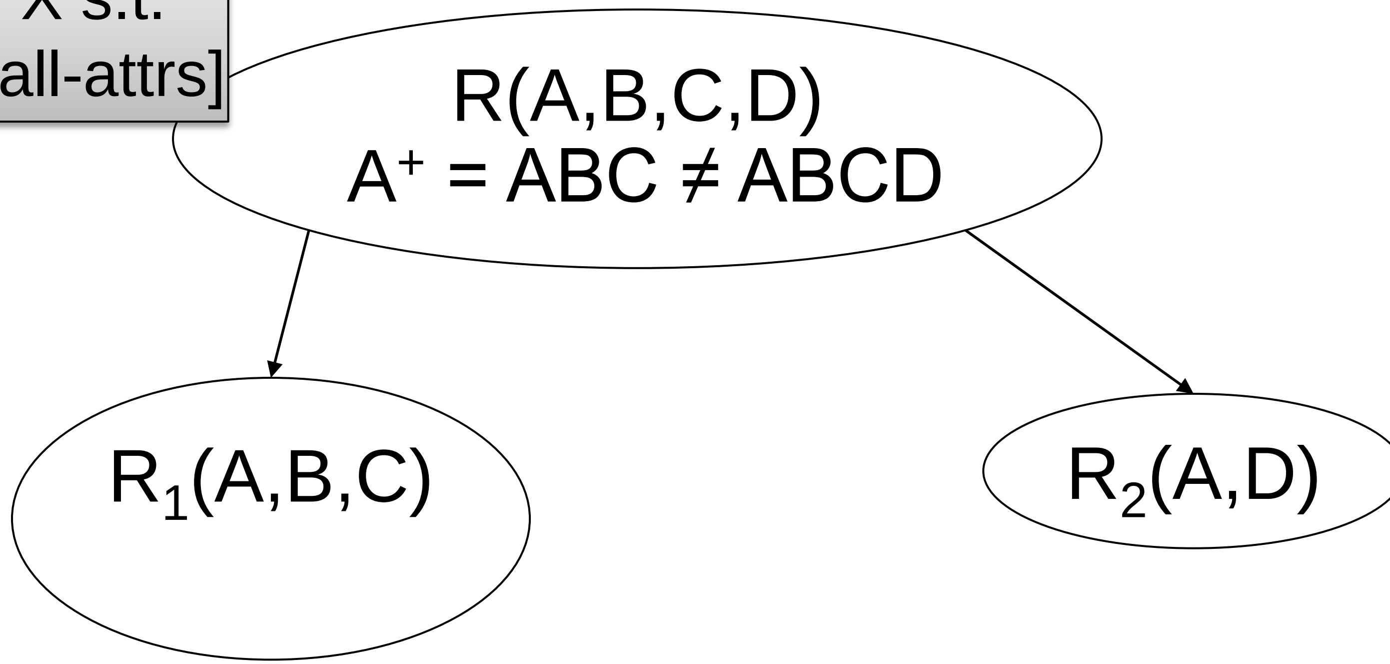
R(A,B,C,D)

subset

Recall: find X s.t.
$X \subsetneq X^+ \subsetneq$ [all-attrs]

Review: BCNF

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



R(A,B,C,D)

Review: BCNF

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

subset

Recall: find X s.t.
$X \subsetneq X^+ \subsetneq$ [all-attrs]

R(A,B,C,D)
$A^+ = ABC \neq ABCD$

$R_1(A,B,C)$
$B^+ = BC \neq ABC$

$R_2(A,D)$

R(A,B,C,D)

subset

Review: BCNF

$A \to B$
$B \to C$

Recall: find X s.t.
$X \subsetneq X^+ \subsetneq$ [all-attrs]

R(A,B,C,D)
$A^+ = ABC \neq ABCD$

$R_1(A,B,C)$
$B^+ = BC \neq ABC$

$R_2(A,D)$

$R_{11}(B,C)$

$R_{12}(A,B)$

What are the keys ?

What happens if in R we first pick B^+ ? Or AB^+ ?

Why BCNF decomposition?

- We want to ensure that the join is “lossless”
- Suppose we decompose R(A, B, C) to R1(A, B), and R2(A, C)
  - If we join R1, R2 on A, we will get all tuples in R.
  - But will we get additional spurious tuples that were not in R?
  - Not if the decomposition is lossless, like BCNF. Then we get exactly the original relation R back.

Decompositions in General

$R(A_1, \ldots, A_n, B_1, \ldots, B_m, C_1, \ldots, C_p)$

$S_1(A_1, \ldots, A_n, B_1, \ldots, B_m)$

$S_2(A_1, \ldots, A_n, C_1, \ldots, C_p)$

S_1 = projection of R on $A_1, \ldots, A_n, B_1, \ldots, B_m$
S_2 = projection of R on $A_1, \ldots, A_n, C_1, \ldots, C_p$

Lossless Decomposition

Name	Price	Category
Gizmo	19.99	Gadget
OneClick	24.99	Camera
Gizmo	19.99	Camera

Name	Price
Gizmo	19.99
OneClick	24.99
~~Gizmo~~	~~19.99~~

Name	Category
Gizmo	Gadget
OneClick	Camera
Gizmo	Camera

Lossy Decomposition

What is lossy here?

Name	Price	Category
Gizmo	19.99	Gadget
OneClick	24.99	Camera
Gizmo	19.99	Camera

Name	Category
Gizmo	Gadget
OneClick	Camera
Gizmo	Camera

Price	Category
19.99	Gadget
24.99	Camera
19.99	Camera

Lossy Decomposition

What is lossy here?

Name	Price	Category
Gizmo	19.99	Gadget
OneClick	24.99	Camera
Gizmo	19.99	Camera

Name	Category
Gizmo	Gadget
OneClick	Camera
Gizmo	Camera

Price	Category
19.99	Gadget
24.99	Camera
19.99	Camera

Decomposition in General

$R(A_1, ..., A_n, B_1, ..., B_m, C_1, ..., C_p)$

$S_1(A_1, ..., A_n, B_1, ..., B_m)$

$S_2(A_1, ..., A_n, C_1, ..., C_p)$

Let: S_1 = projection of R on $A_1, ..., A_n, B_1, ..., B_m$
S_2 = projection of R on $A_1, ..., A_n, C_1, ..., C_p$

The decomposition is called *lossless* if $R = S_1 \bowtie S_2$

Verify yourself:
If $A_1, ..., A_n \rightarrow B_1, ..., B_m$ then the decomposition is lossless

It follows that every BCNF decomposition is losseless

Chase Test

Suppose we have decomposed a relation
R(A,B,C,D)
into
S1(A,D)   S2(A,C)   S3(B,C,D)

We want to test if this decomposition is Lossless given a set of func. dependencies F

Reading: 3.4.2

Example from textbook Ch. 3.4.2

The Chase Test for Lossless Join

$R(A,B,C,D) = S1(A,D) \bowtie S2(A,C) \bowtie S3(B,C,D)$
R satisfies: $A \rightarrow B$, $B \rightarrow C$, $CD \rightarrow A$

$S1 = \Pi_{AD}(R)$, $S2 = \Pi_{AC}(R)$, $S3 = \Pi_{BCD}(R)$,
hence $R \subseteq S1 \bowtie S2 \bowtie S3$
Need to check: $R \supseteq S1 \bowtie S2 \bowtie S3$

The Chase Test for Lossless Join

$R(A,B,C,D) = S1(A,D) \bowtie S2(A,C) \bowtie S3(B,C,D)$
R satisfies: $A \rightarrow B$, $B \rightarrow C$, $CD \rightarrow A$

$S1 = \Pi_{AD}(R)$, $S2 = \Pi_{AC}(R)$, $S3 = \Pi_{BCD}(R)$,
hence $R \subseteq S1 \bowtie S2 \bowtie S3$
Need to check: $R \supseteq S1 \bowtie S2 \bowtie S3$
Suppose $(a,b,c,d) \in S1 \bowtie S2 \bowtie S3$ Is it also in R?
R must contain the following tuples:

A	B	C	D
a	b1	c1	d

Why ?
$(a,d) \in S1 = \Pi_{AD}(R)$

Example from textbook Ch. 3.4.2

The Chase Test for Lossless Join

R(A,B,C,D) = S1(A,D) ⋈ S2(A,C) ⋈ S3(B,C,D)
R satisfies: A→B, B→C, CD→A

S1 = Π_{AD}(R), S2 = Π_{AC}(R), S3 = Π_{BCD}(R),
hence R⊆ S1 ⋈ S2 ⋈ S3
Need to check: R ⊇ S1 ⋈ S2 ⋈ S3
Suppose (a,b,c,d) ∈ S1 ⋈ S2 ⋈ S3 Is it also in R?
R must contain the following tuples:

A	B	C	D	
				Why ?
a	b1	c1	d	(a,d) ∈S1 = Π_{AD}(R)
a	b2	c	d2	(a,c) ∈S2 = Π_{AC}(R)

The Chase Test for Lossless Join

$R(A,B,C,D) = S1(A,D) \bowtie S2(A,C) \bowtie S3(B,C,D)$
R satisfies: $A \rightarrow B$, $B \rightarrow C$, $CD \rightarrow A$

$S1 = \Pi_{AD}(R)$, $S2 = \Pi_{AC}(R)$, $S3 = \Pi_{BCD}(R)$,
hence $R \subseteq S1 \bowtie S2 \bowtie S3$
Need to check: $R \supseteq S1 \bowtie S2 \bowtie S3$
Suppose $(a,b,c,d) \in S1 \bowtie S2 \bowtie S3$ Is it also in R?
R must contain the following tuples:

A	B	C	D	Why ?
a	b1	c1	d	$(a,d) \in S1 = \Pi_{AD}(R)$
a	b2	c	d2	$(a,c) \in S2 = \Pi_{BD}(R)$
a3	b	c	d	$(b,c,d) \in S3 = \Pi_{BCD}(R)$

Example from textbook Ch. 3.4.2

The Chase Test for Lossless Join

R(A,B,C,D) = S1(A,D) ⋈ S2(A,C) ⋈ S3(B,C,D)
R satisfies: A→B, B→C, CD→A

S1 = Π_{AD}(R), S2 = Π_{AC}(R), S3 = Π_{BCD}(R),
hence R⊆ S1 ⋈ S2 ⋈ S3
Need to check: R ⊇ S1 ⋈ S2 ⋈ S3
Suppose (a,b,c,d) ∈ S1 ⋈ S2 ⋈ S3 Is it also in R?
R must contain the following tuples:

A	B	C	D	Why ?
a	b1	c1	d	(a,d) ∈S1 = Π_{AD}(R)
a	b2	c	d2	(a,c) ∈S2 = Π_{BD}(R)
a3	b	c	d	(b,c,d) ∈S3 = Π_{BCD}(R)

"Chase" them (apply FDs):

A→B

A	B	C	D
a	b1	c1	d
a	b1	c	d2
a3	b	c	d

Example from textbook Ch. 3.4.2

The Chase Test for Lossless Join

$R(A,B,C,D) = S1(A,D) \bowtie S2(A,C) \bowtie S3(B,C,D)$
R satisfies: $A \rightarrow B$, $B \rightarrow C$, $CD \rightarrow A$

$S1 = \Pi_{AD}(R)$, $S2 = \Pi_{AC}(R)$, $S3 = \Pi_{BCD}(R)$,
hence $R \subseteq S1 \bowtie S2 \bowtie S3$
Need to check: $R \supseteq S1 \bowtie S2 \bowtie S3$
Suppose $(a,b,c,d) \in S1 \bowtie S2 \bowtie S3$ Is it also in R?
R must contain the following tuples:

A	B	C	D	Why ?
a	b1	c1	d	$(a,d) \in S1 = \Pi_{AD}(R)$
a	b2	c	d2	$(a,c) \in S2 = \Pi_{AC}(R)$
a3	b	c	d	$(b,c,d) \in S3 = \Pi_{BCD}(R)$

"Chase" them (apply FDs):

$A \rightarrow B$

A	B	C	D
a	b1	c1	d
a	b1	c	d2
a3	b	c	d

$B \rightarrow C$

A	B	C	D
a	b1	c	d
a	b1	c	d2
a3	b	c	d

Example from textbook Ch. 3.4.2

The Chase Test for Lossless Join

$R(A,B,C,D) = S1(A,D) \bowtie S2(A,C) \bowtie S3(B,C,D)$
R satisfies: $A \rightarrow B$, $B \rightarrow C$, $CD \rightarrow A$

$S1 = \Pi_{AD}(R)$, $S2 = \Pi_{AC}(R)$, $S3 = \Pi_{BCD}(R)$,
hence $R \subseteq S1 \bowtie S2 \bowtie S3$
Need to check: $R \supseteq S1 \bowtie S2 \bowtie S3$
Suppose $(a,b,c,d) \in S1 \bowtie S2 \bowtie S3$ Is it also in R?
R must contain the following tuples:

A	B	C	D	Why ?
a	b1	c1	d	$(a,d) \in S1 = \Pi_{AD}(R)$
a	b2	c	d2	$(a,c) \in S2 = \Pi_{AC}(R)$
a3	b	c	d	$(b,c,d) \in S3 = \Pi_{BCD}(R)$

"Chase" them (apply FDs):

$A \rightarrow B$

A	B	C	D
a	b1	c1	d
a	b1	c	d2
a3	b	c	d

$B \rightarrow C$

A	B	C	D
a	b1	c	d
a	b1	c	d2
a3	b	c	d

$CD \rightarrow A$

A	B	C	D
a	b1	c	d
a	b1	c	d2
a	b	c	d

Hence R contains (a,b,c,d)

i.e. lossless

Schema Refinements = Normal Forms

- 1st Normal Form = all tables are flat
  - all attribute values are atomic
- 2nd Normal Form = obsolete
- Boyce Codd Normal Form = discussed in class
- 3rd Normal Form = see book (optional 3.5)
  - BCNF is lossless, but after join the relation may not satisfy some original F.D.
  - 3NF fixes that (both lossless and dependency-preserving)

Views
(more in Lecture 19)

Views

- A view in SQL =
  - A table computed from other tables, s.t., whenever the base tables are updated, the view is updated too
- More generally:
  - A view is derived data that keeps track of changes in the original data
- Compare:
  - A function computes a value from other values, but does not keep track of changes to the inputs

A Simple View

Create a view that returns for each store
the prices of products purchased at that store

```
CREATE VIEW  StorePrice AS
  SELECT DISTINCT x.store, y.price
  FROM  Purchase x, Product y
  WHERE x.product = y.pname
```

This is like a new table
StorePrice(store,price)

We Use a View Like Any Table

- A "high end" store is a store that sell some products over 1000.
- For each customer, return all the high end stores that they visit.

```
SELECT DISTINCT u.name, u.store
FROM Purchase u, StorePrice v
WHERE u.store = v.store
    AND v.price > 1000
```

Purchase(customer, product, store)
Product(pname, price)

StorePrice(store, price)

Query Modification

For each customer, find all the high end stores that they visit.

```
CREATE VIEW  StorePrice AS
  SELECT DISTINCT x.store, y.price
  FROM  Purchase x, Product y
  WHERE x.product = y.pname
```

```
SELECT DISTINCT u.name, u.store
FROM Purchase u, StorePrice v
WHERE u.store = v.store
    AND v.price > 1000
```

Purchase(customer, product, store)
Product(<u>pname</u>, price)

StorePrice(store, price)

Query Modification

For each customer, find all the high end stores that they visit.

```
CREATE VIEW  StorePrice AS
  SELECT DISTINCT x.store, y.price
  FROM  Purchase x, Product y
  WHERE x.product = y.pname
```

```
SELECT DISTINCT u.name, u.store
FROM Purchase u, StorePrice v
WHERE u.store = v.store
   AND v.price > 1000
```

Modified query:

```
SELECT DISTINCT u.customer, u.store
FROM Purchase u,
 (SELECT DISTINCT x.store, y.price
  FROM  Purchase x, Product y
  WHERE x.product = y.pname) v
WHERE u.store = v.store
    AND v.price > 1000
```