CMPT 354: Database System I

Lecture 9. Design Theory

Design Theory

- **Design theory** is about how to represent your data to avoid anomalies.

Design 1

Student	Course	Room
Mike	354	AQ3149
Mary	354	AQ3149
Sam	354	AQ3149
..	..	..

Design 2

Student	Course
Mike	354
Mary	354
Sam	354
..	..

Course	Room
354	AQ3149
454	T9204

Four Types of Anomalies - 1

- What’s wrong?

Student	Course	Room
Mike	354	AQ3149
Mary	354	AQ3149
Sam	354	AQ3149
..	..	..

If every course is in only one room, contains ***redundant*** information!

Four Types of Anomalies - 2

- What's wrong?

Student	Course	Room
Mike	354	AQ3149
Mary	354	T9204
Sam	354	AQ3149
..	..	..

If we update the room number for one tuple, we get inconsistent data = an ***update* anomaly**

Four Types of Anomalies - 3

- What’s wrong?

Student	Course	Room
..	..	..

If everyone drops the class, we lose what room the class is in! = a ***delete* anomaly**

Four Types of Anomalies - 4

- What's wrong?

...	454	T9204

Student	Course	Room
Mike	354	AQ3149
Mary	354	AQ3149
Sam	354	AQ3149
..	..	..

Similarly, we can't reserve a room without students = an ***insert* anomaly**

Elimination of Anomalies

- Is it better?

Student	Course
Mike	354
Mary	354
Sam	354
..	..

Course	Room
354	AQ3149
454	T9204

- Redundancy?
- Update anomaly?
- Delete anomaly?
- Insert anomaly?

Why this design may be better?
How to find this *decomposition?*

Normal Forms

- <u>1st Normal Form (1NF)</u> = All tables are flat
- *<u>2nd Normal Form</u> = disused*
- <u>Boyce-Codd Normal Form (BCNF)</u> = no bad FDs
- <u>3rd, 4th, and 5th Normal Forms</u> = see text books

1^{st} Normal Form (1NF)

Student	Courses
Mary	{CS145,CS229}
Joe	{CS145,CS106}
…	…

Violates 1NF.

Student	Courses
Mary	CS145
Mary	CS229
Joe	CS145
Joe	CS106

In 1^{st} NF

1NF Constraint: Types must be atomic!

Normal Forms

- 1^{st} Normal Form (1NF) = All tables are flat
- *2^{nd} Normal Form = disused*
- **Boyce-Codd Normal Form (BCNF) = no bad FDs**
- 3^{rd}, 4^{th}, and 5^{th} Normal Forms = see text books

What's this?

Outline

1. Functional Dependency (FD)

2. Inference Problem

3. Closure Algorithm

Functional Dependency

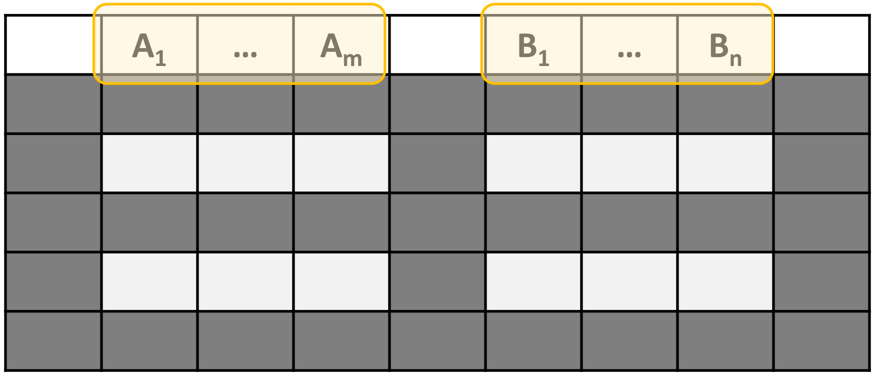
Def: Let A,B be *sets* of attributes
We write A $\rightarrow$ B or say A ***functionally determines*** B if, for any tuples t_1 and t_2:

$$t_1[A] = t_2[A] \text{ implies } t_1[B] = t_2[B]$$

and we call A $\rightarrow$ B a functional dependency

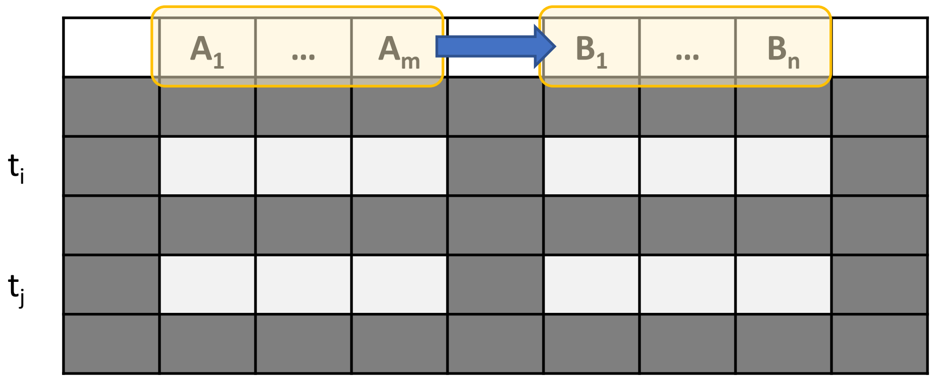
A->B means that
"whenever two tuples agree on A then they agree on B."

A Picture Of FDs



<u>Defn (again):</u>
Given attribute sets $A=\{A_1,\ldots,A_m\}$ and $B = \{B_1,\ldots B_n\}$ in R,

A Picture Of FDs

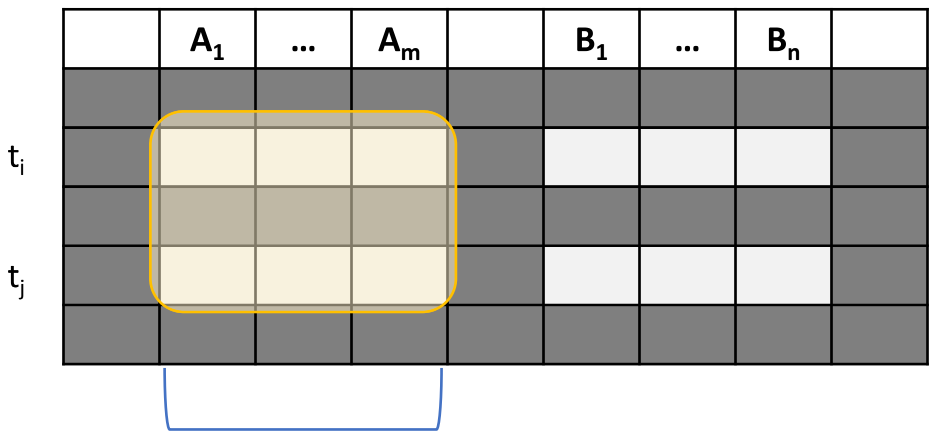


Defn (again):

Given attribute sets $A=\{A_1,\ldots,A_m\}$ and $B = \{B_1,\ldots B_n\}$ in R,

The *functional dependency* $A \rightarrow B$ on R holds if for ***any*** t_i,t_j in R:

A Picture Of FDs



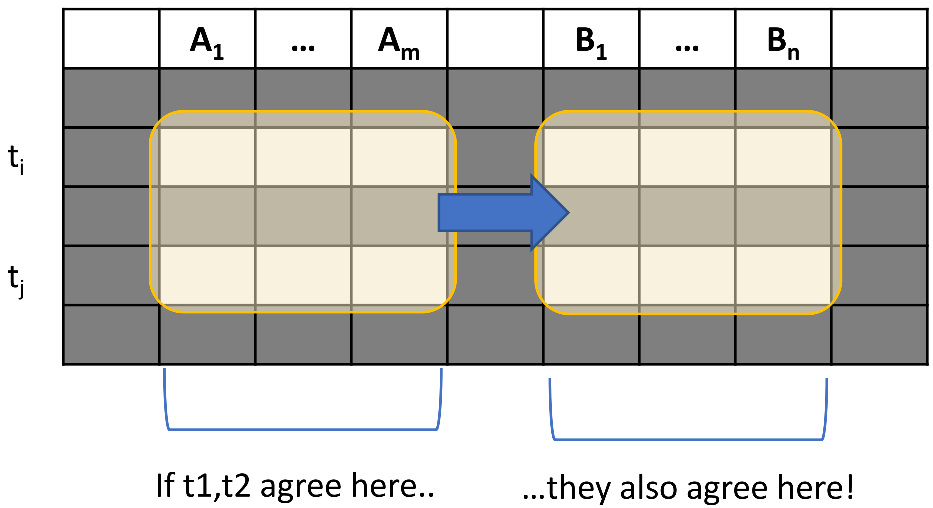
If t1,t2 agree here..

<u>Defn (again):</u>
Given attribute sets $A=\{A_1,\ldots,A_m\}$ and $B = \{B_1,\ldots B_n\}$ in R,

The *functional dependency* $A \rightarrow B$ on R holds if for ***any*** t_i,t_j in R:

$t_i[A_1] = t_j[A_1]$ AND $t_i[A_2]=t_j[A_2]$ AND … AND $t_i[A_m] = t_j[A_m]$

A Picture Of FDs



Defn (again):

Given attribute sets $A=\{A_1,\ldots,A_m\}$ and $B = \{B_1,\ldots B_n\}$ in R,

The *functional dependency* $A \rightarrow B$ on R holds if for ***any*** t_i,t_j in R:

if $t_i[A_1] = t_j[A_1]$ AND $t_i[A_2]=t_j[A_2]$ AND ... AND $t_i[A_m] = t_j[A_m]$

then $t_i[B_1] = t_j[B_1]$ AND $t_i[B_2]=t_j[B_2]$ AND ... AND $t_i[B_n] = t_j[B_n]$

Example

An FD <u>holds</u>, or <u>does not hold</u> on a table:

EmpID	Name	Phone	Position
E0045	Smith	1234	Clerk
E3542	Mike	9876	Salesrep
E1111	Smith	9876	Salesrep
E9999	Mary	1234	Lawyer

Position → Phone

Phone → Position

Phone, Name → Position

Exercise - 1

An FD holds, or does not hold on a table:

Name	Category	Color	Department	Price
Gizmo	Gadget	Green	Toys	49
Tweaker	Gadget	Green	Toys	49
Gizmo	Stationary	Green	Office-supply	59

Name → Color

Category → Department

Color, Category → Color

Exercise - 2

A	B	C	D	E
1	2	4	3	6
3	2	5	1	8
1	4	4	5	7
1	2	4	3	6
3	2	5	1	8

Find at least *three* FDs which **do not hold** on this table:

{ } → { }
{ } → { }
{ } → { }

Outline

1. Functional Dependency (FD)

2. **Inference Problem**

3. Closure Algorithm

An Interesting Observation

Provided FDs:

1. Name → Color
2. Category → Department
3. Color, Category → Price

Does it always hold? **Name, Category → Price**

If we find out from application domain that a relation satisfies some FDs, it doesn't mean that we found all the FDs that it satisfies! There could be more FDs implied by the ones we have

Inference Problem

Whether or not a set of FDs imply another FD?

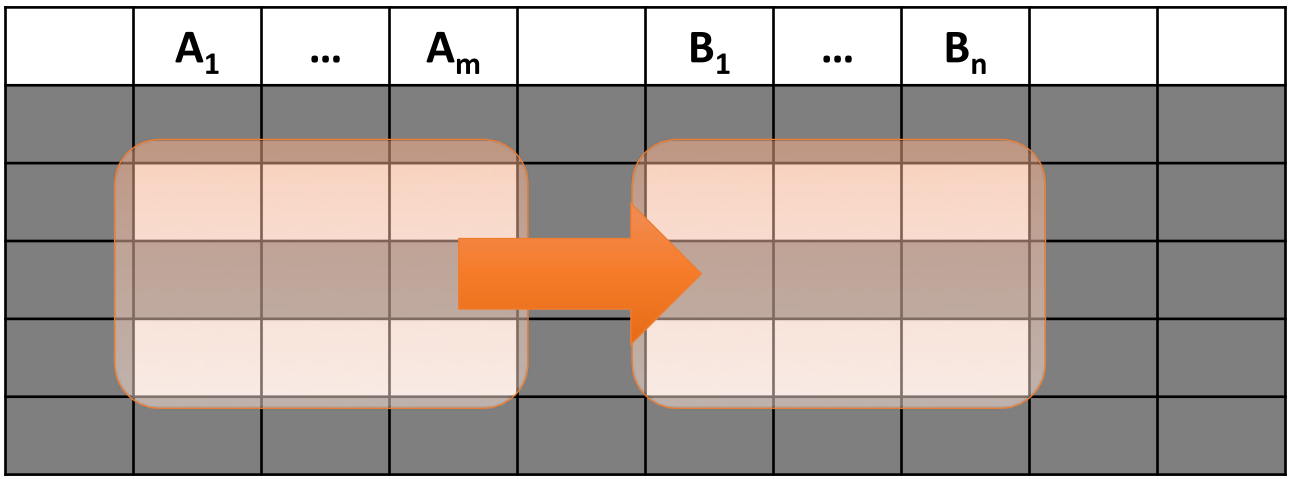
This is called **Inference problem**

Answer: Three simple rules called
Armstrong's Rules.

1. Split/Combine,
2. Reduction, and
3. Transitivity

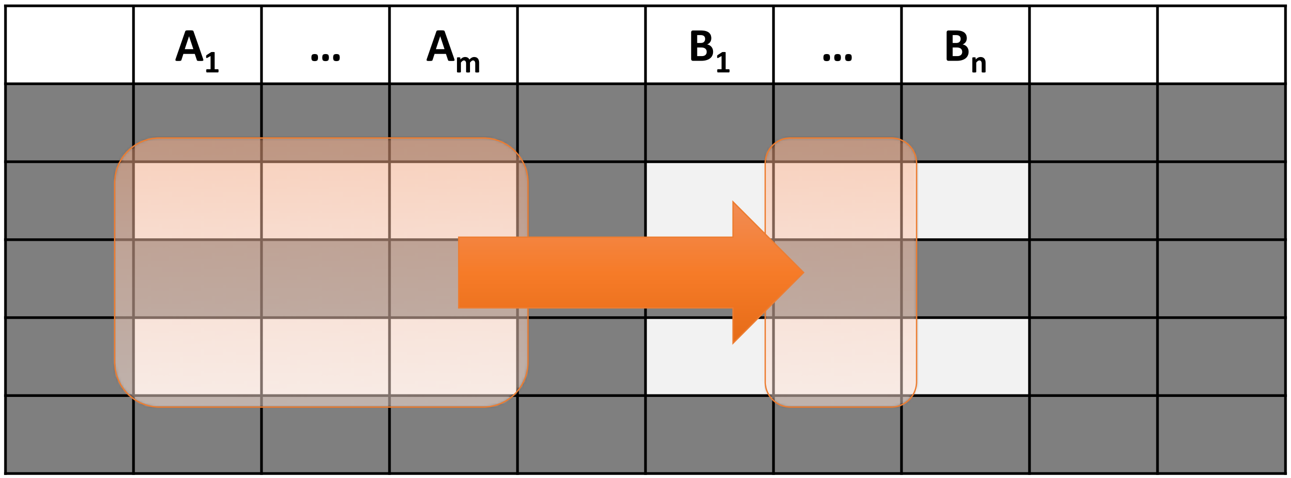
> **William Ward Armstrong** is a Canadian mathematician and computer scientist. He earned his Ph.D. from the University of British Columbia in 1966 and is most known as the originator Armstrong's axioms of dependency in a Relational database.[1]

1. Split/Combine



$A_1, \ldots, A_m \rightarrow B_1, \ldots, B_n$

1. Split/Combine

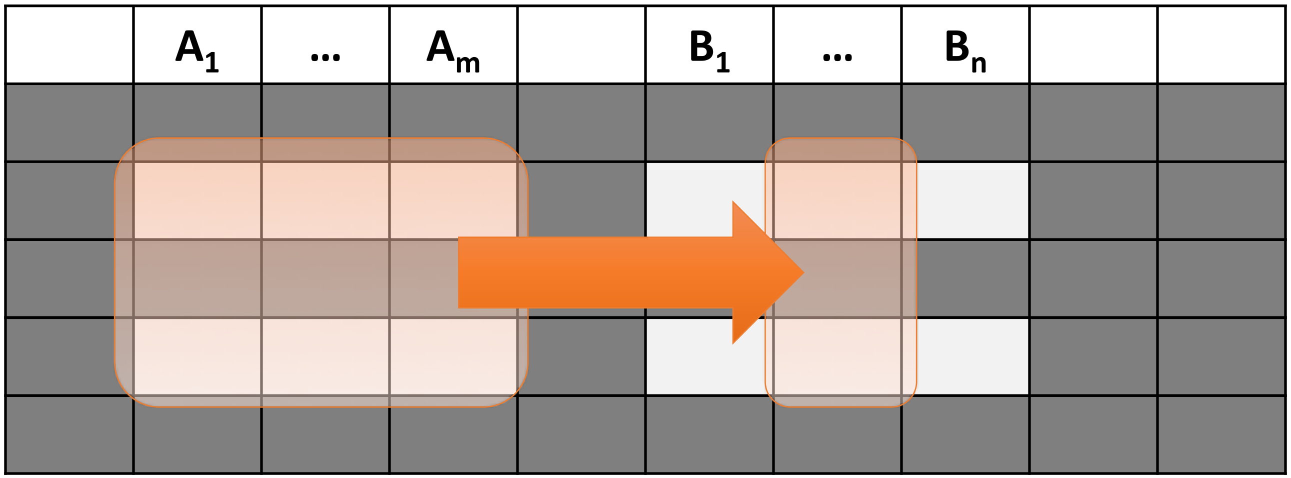


$A_1, \ldots, A_m \rightarrow B_1,\ldots,B_n$

… is equivalent to the following *n* FDs…

$A_1,\ldots,A_m \rightarrow B_i$ for $i=1,\ldots,n$

1. Split/Combine

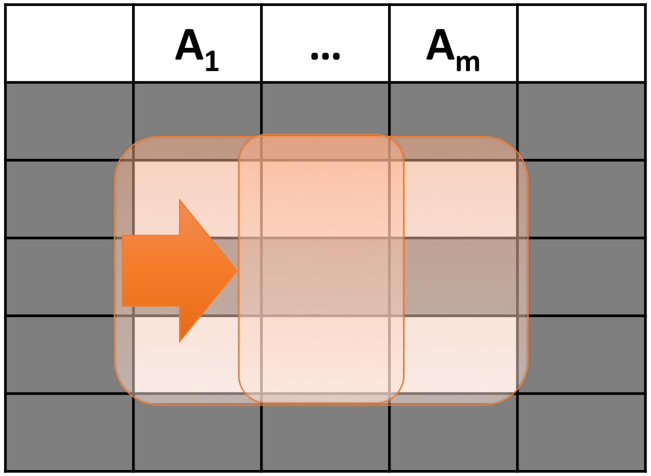


And vice-versa, $A_1,\ldots,A_m \rightarrow B_i$ for $i=1,\ldots,n$

… is equivalent to …

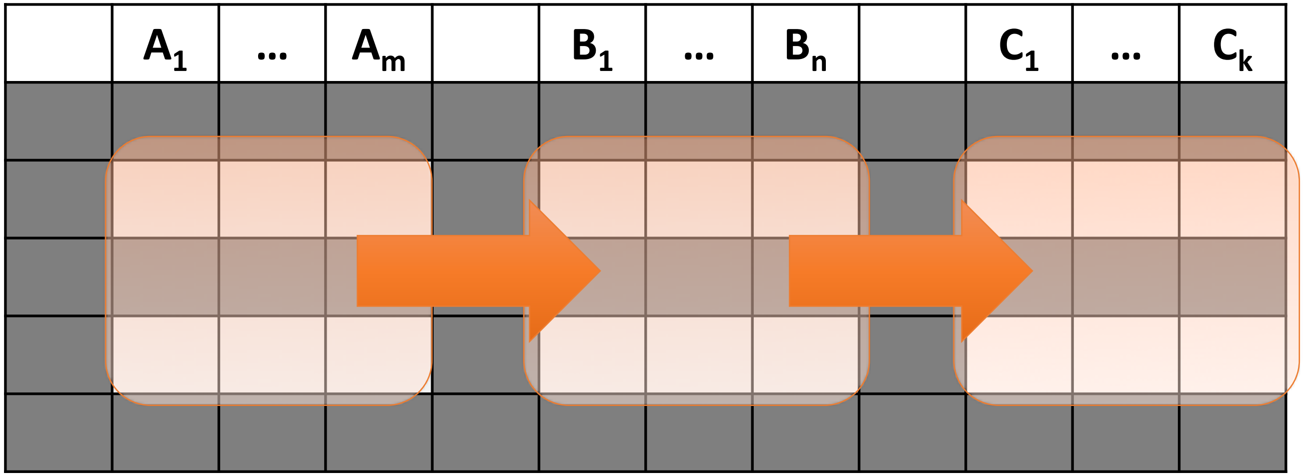
$A_1, \ldots, A_m \rightarrow B_1,\ldots,B_n$

2. Reduction/Trivial



$A_1,\ldots,A_m \rightarrow A_j$ for any $j=1,\ldots,m$

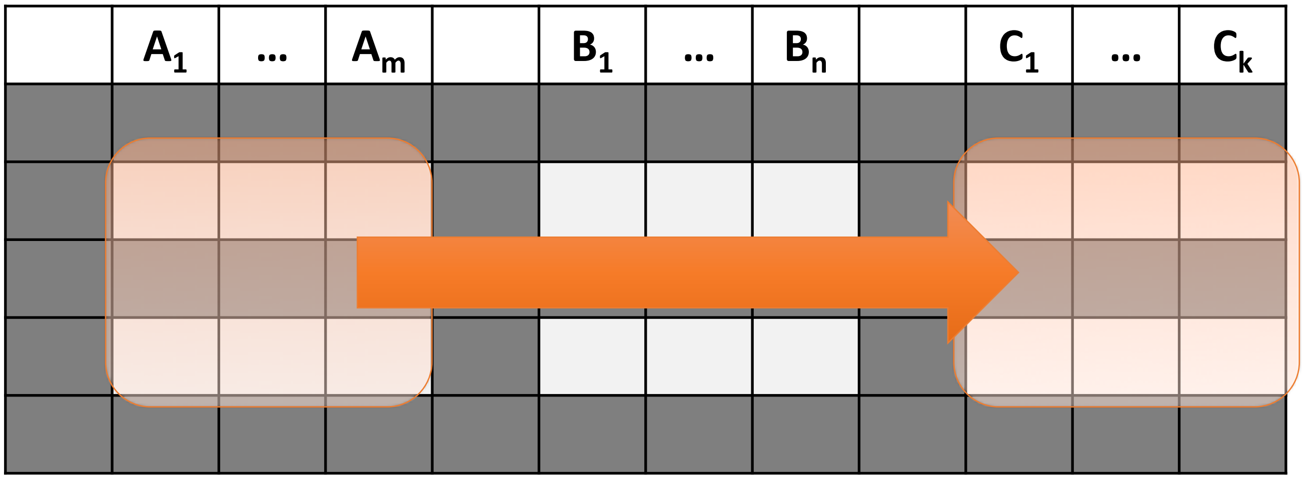
3. Transitive



$A_1, \ldots, A_m \rightarrow B_1,\ldots,B_n$ and

$B_1,\ldots,B_n \rightarrow C_1,\ldots,C_k$

3. Transitive



$A_1, \ldots, A_m \rightarrow B_1,\ldots,B_n$ and
$B_1,\ldots,B_n \rightarrow C_1,\ldots,C_k$

implies

$A_1,\ldots,A_m \rightarrow C_1,\ldots,C_k$

Inferred FDs

Example:

Inferred FDs:

Inferred FD	Rule used
4. Name, Category → Name	?
5. Name, Category → Color	?
6. Name, Category → Category	?
7. Name, Category → Color, Category	?
8. Name, Category → Price	?

Provided FDs:

1. {Name} → {Color}
2. {Category} → {Dept.}
3. {Color, Category} → {Price}

Which / how many other FDs hold?

Inferred FDs

Example:

Inferred FDs:

Inferred FD	Rule used
4. Name, Category → Name	Trivial
5. Name, Category → Color	Transitive (4 -> 1)
6. Name, Category → Category	Trivial
7. Name, Category → Color, Category	Split/combine (5 + 6)
8. Name, Category → Price	Transitive (7 -> 3)

Provided FDs:

1. {Name} → {Color}
2. {Category} → {Dept.}
3. {Color, Category} → {Price}

Can we find an algorithmic way to do this?

Outline

1. Functional Dependency (FD)

2. Inference Problem

3. Closure Algorithm

Closure of a set of Attributes

Given a set of attributes $A_1, \ldots, A_n$ and a set of FDs **F:**

Then the <u>closure</u>, $\{A_1, \ldots, A_n\}^+$ is the set of attributes **B** s.t. $\{A_1, \ldots, A_n\} \rightarrow B$

<u>Example:</u> F =

```
name → color
category → department
color, category → price
```

Closures:

```
{name}+ = {name, color}
{name, category}+ = {name, category, color, dept, price}
{color}+ = {color}
```

Closure Algorithm

Start with X = $\{A_1, \ldots, A_n\}$ and set of FDs F.

Repeat until X doesn't change; **do**:

if $\{B_1, \ldots, B_n\} \rightarrow C$ is in F

and $\{B_1, \ldots, B_n\} \subseteq X$

then add C to X.

Return X as X^+

Closure Algorithm

Start with $X = \{A_1, \ldots, A_n\}$, FDs F.
Repeat until X doesn't change; **do**:
  if $\{B_1, \ldots, B_n\} \rightarrow C$ is in F
    and $\{B_1, \ldots, B_n\} \subseteq X$:
      then add C to X.
Return X as X^+

```
{name, category}+ =
{name, category}
```

F =

```
name → color

category → dept

color, category → price
```

Closure Algorithm

Start with $X = \{A_1, \ldots, A_n\}$, FDs F.
Repeat until X doesn't change; **do**:
  if $\{B_1, \ldots, B_n\} \rightarrow C$ is in F
    and $\{B_1, \ldots, B_n\} \subseteq X$:
      then add C to X.
Return X as X^+

```
{name, category}+ =
{name, category}
```

```
{name, category}+ =
{name, category, color}
```

F =

```
name → color

category → dept

color, category → price
```

Closure Algorithm

Start with $X = \{A_1, \ldots, A_n\}$, FDs F.
Repeat until X doesn't change; **do**:
  if $\{B_1, \ldots, B_n\} \rightarrow C$ is in F
    and $\{B_1, \ldots, B_n\} \subseteq X$:
      then add C to X.
Return X as X^+

F =

```
name → color
category → dept
color, category → price
```

```
{name, category}⁺ =
{name, category}
```

```
{name, category}⁺ =
{name, category, color}
```

```
{name, category}⁺ =
{name, category, color, dept}
```

Closure Algorithm

Start with X = $\{A_1, \ldots, A_n\}$, FDs F.
Repeat until X doesn't change; **do**:
  if $\{B_1, \ldots, B_n\} \rightarrow C$ is in F
    and $\{B_1, \ldots, B_n\} \subseteq X$:
      then add C to X.
Return X as X^+

F =

```
name → color

category → dept

color, category → price
```

```
{name, category}+ =
{name, category}
```

```
{name, category}+ =
{name, category, color}
```

```
{name, category}+ =
{name, category, color, dept}
```

```
{name, category}+ =
{name, category, color, dept,
price}
```

Exercise - 3

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

A,B	→	C
A,D	→	E
B	→	D
A,F	→	B

Compute $\{A,B\}^+ = \{A, B,$ $\}$

Compute $\{A, F\}^+ = \{A, F,$ $\}$

Exercise - 3

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

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

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

Compute $\{A, F\}^+ = \{A, F, B \qquad \}$

Exercise - 3

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

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

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

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

Exercise - 4

- Find all FD's implied by

$$
\begin{aligned}
A,B &\rightarrow C \\
A,D &\rightarrow B \\
B &\rightarrow D
\end{aligned}
$$

Requirements

1. **Non-trivial** FD (i.e., no need to return A, B → A)
2. The right-hand side contains **a single** attribute (i.e., no need to return A, B → C, D)

Exercise - 4

Given F =

A,B	→	C
A,D	→	B
B	→	D

Step 1: Compute X^+, for every set of attributes X:

$\{A\}^+ = ?$
$\{B\}^+ = ?$
$\{C\}^+ = ?$
$\{D\}^+ = ?$
$\{A,B\}^+ = ?$
$\{A,C\}^+ = ?$
$\{A,D\}^+ = ?$
$\{B,C\}^+ = ?$
$\{B,D\}^+ = ?$
$\{C,D\}^+ = ?$
$\{A,B,C\}^+ = ?$
$\{A,B,D\}^+ = ?$
$\{A,C,D\}^+ = ?$
$\{B,C,D\}^+ = ?$
$\{A,B,C,D\}^+ = ?$

Exercise - 4

Given F =

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

Step 1: Compute X^+, for every set of attributes X:

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

Exercise - 4

Given F =

```
A,B → C
A,D → B
B   → D
```

Step 2: Enumerate all FDs X → Y, s.t. Y ⊆ X⁺ and X ∩ Y = ∅:

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

```
B     → D
A,B   → C
A,B   → D
A,D   → B
A,D   → C
A,B,C → D
A,B,D → C
A,C,D → B
```

Summary

1. Functional Dependency (FD)
   - What is an FD?

2. Inference Problem
   - Whether or not a set of FDs imply another FD?

3. Closure
   - How to compute the closure of attributes?

Acknowledge

- Some lecture slides were copied from or inspired by the following course materials
  - "W4111: Introduction to databases" by Eugene Wu at Columbia University
  - "CSE344: Introduction to Data Management" by Dan Suciu at University of Washington
  - "CMPT354: Database System I" by John Edgar at Simon Fraser University
  - "CS186: Introduction to Database Systems" by Joe Hellerstein at UC Berkeley
  - "CS145: Introduction to Databases" by Peter Bailis at Stanford
  - "CS 348: Introduction to Database Management" by Grant Weddell at University of Waterloo