Functional Dependencies

Wk8 Mon

Motivation Example

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

Observe these two columns

- The values of hourly_wages depend on that of rating.
  - Rating = 8 → hourly_wages = 10
  - Rating = 5 → hourly_wages = 7

Motivation Example

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

- Redundant Storage
  - The rating value 8 corresponds to the hourly wage 10, and this association is repeated three times.
  - Storage is not used efficiently.

Motivation Example

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

- Update Anomalies
  - The hourly_wages in the first tuple could be updated without making a similar change in the second tuple.
  - Therefore, updates may cause inconsistency.

Motivation Example

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

- Insertion Anomalies
  - We cannot insert a tuple for an employee unless we know the hourly wage for the employee's rating value.
  - Otherwise, null values may appear in the table.

Motivation Example

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

- Deletion Anomalies
  - If we delete all tuples with a given rating value (e.g. tuples of Mary and David) we lose the association between the rating value and its hourly_wage value.

Problems caused by redundancy

1. **Redundant Storage**
   - Some information is stored repeatedly.

2. **Update Anomalies**
   - If one copy of such repeated data is updated, an inconsistency is created, unless all copies are similarly updated.

3. **Insertion anomalies**
   - It may not be possible to store certain information unless some other, unrelated, information is stored.

4. **Deletion Anomalies**
   - It may not be possible to delete certain information without losing some other, unrelated, information.

Decompositions

- Intuitively, redundancy arise when a relational schema forces an association between attributes.
- Functional dependencies can be used to identify such situations and suggest refinements to the schema.
- The essential idea is that many problems arising from redundancy can be addressed by replacing a relation with a collection of ‘smaller’ relation.

Id	name	age	rating	Hourly_wages	Hours_worked
123-22-3666	Peter	48	8	10	40
231-31-5368	Paul	22	8	10	30
131-24-3650	Mary	35	5	7	30
434-26-3751	David	35	5	7	32
612-67-4134	Ada	35	8	10	40

Id	name	age	rating	Hours_worked
123-22-3666	Peter	48	8	40
231-31-5368	Paul	22	8	30
131-24-3650	Mary	35	5	30
434-26-3751	David	35	5	32
612-67-4134	Ada	35	8	40

rating	Hourly_wages
8	10
5	7

Functional dependency:
- rating determines Hourly_wages

A decomposition of a relation schema R consists of replacing the relation schema by two (or more) relation schemas that each contains a subset of attributes of R and together include all attributes in R

Why RDBS Design Again?

- Suppose we have a table *inst_dept which contain information for both instructor* and *department*.
- Result is possible repetition of information

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Relational Design by Decomposition

ID	*name*	*street*	*city*	*salary*
⋮				
57766	*Kim*	*Main*	*Perryridge*	*75000*
98776	*Kim*	*North*	*Hampton*	*67000*
⋮				

employee

ID	*name*
⋮	
57766	*Kim*
98776	*Kim*
⋮	

name	*street*	*city*	*salary*
⋮			
Kim	*Main*	*Perryridge*	*75000*
Kim	*North*	*Hampton*	*67000*
⋮			

natural join

ID	*name*	*street*	*city*	*salary*
⋮				
57766	*Kim*	*Main*	*Perryridge*	*75000*
57766	*Kim*	*North*	*Hampton*	*67000*
98776	*Kim*	*Main*	*Perryridge*	*75000*
98776	*Kim*	*North*	*Hampton*	*67000*
⋮				

Lossless-Join Decomposition

- A simple example of Lossy-Join (Non-Lossless Join):
- Decomposition of $R = (A, B)$ *into* $R_1 = (A)$ *and* $R_2 = (B)$

A	B
α	1
α	2
β	1

r

A
α
β

$\Pi_A(r)$

B
1
2

$\Pi_{B(r)}$

A	B
α	1
α	2
β	1
β	2

$\Pi_A (r) \bowtie \Pi_B (r)$

- Lossless-Join decomposition:
- For all possible relation instance r on schema R

$$r = \Pi_{R1}(r) \bowtie \Pi_{R2}(r)$$

Goal - Devise a Theory for what is Good

- We want to:
  - Decide whether a relation R has "**good**" form. If not… (see below)
  - Decompose it into a set of relations {R1, R2, ..., Rn} such that
    1. Each relation is in good form
    2. The decomposition is a lossless-join decomposition
- This theory/set of properties are defined based on functional dependencies.

Attribute Values can be Related

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
22222	Einstein	95000	Biology	Watson	90000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Functional Dependencies

- A functional dependency describes a **relation** between attributes
- Whenever any two tuples t_1 and t_2 of r agree on one attribute α, they also agree on another attribute β.
- $t_1[\alpha] = t_2[\alpha] \Rightarrow t_1[\beta] = t_2[\beta]$
- This relation is denoted $\alpha \rightarrow \beta$.

Functional Dependencies

- *ID* $\rightarrow$ *Name, Depart_name* $\rightarrow$ *Building*

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
22222	Einstein	95000	Biology	Watson	90000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

- Describes the **semantics** or **meaning of the attributes**

Functional Dependencies

The functional dependency

$X \rightarrow Y$ is true (holds)

if and only if

$t_1[X] = t_2[Y] \Rightarrow t_1[X] = t_2[Y]$

in relation R

ID	Name	Code	Grade
100	J	3550	A
200	X	3550	B
100	J	4540	B
100	J	4550	A

- Example: R = {ID, Name, Code, Grade}
  - ID → Name (OK)
  - ID → Grade (not OK), ID → Code (not OK)
  - ID, Name → Grade (not OK), ID, Code → Grade (OK)
  - ID, Name → Name (trivial)

Functional Dependencies

Check understanding (Test 1)

F: $X \rightarrow Y$

X Y

----

a b

a ?

Functional Dependencies

Check understanding (Test 2)

F: $X \rightarrow Y$

X	Y
a	b
?	b

Functional Dependencies

Check understanding (Test 3)

F: $X \rightarrow Y$

X   Y

----

a   b

c   b   ?

Functional Dependencies

Check understanding (Test 4, 5)

$X, Y \rightarrow X$

$X \rightarrow X$

Note: Functional dependencies like these are trivial

Functional Dependencies

Check understanding (Test 6)

Consider R (A , B) with the following instance r.

1	4
1	5
3	7

On this instance, $A \rightarrow B$ does NOT hold, but $B \rightarrow A$ does hold.

FD: relation between two sets

- A functional dependency is a relation between two **sets** of attributes.
  - I.e., the value for a set of attributes determines the value for another set of attributes.

- A functional dependency describes relation between two sets of attributes from are relation.

  Examples:

- $XY \rightarrow WZ$
- $XW \rightarrow Z$
- $Z \rightarrow XQ$

Functional Dependencies

- A functional dependency is a **constraint** between two sets of attributes.
  - A constraint means a constraint across <u>all it's relation instances</u> (extensions). i.e., it must be true for all relation instances.

- Aka, Let's see some examples…

Constraint on all Relations

- Example: *course* → *course_code in* Students

STUDENTS				
id	course	course_code	major	prof
1	Database	353	Comp Sci	Smith
2	Chem101	427	Chemistry	Turner
3	Database	353	Comp Sci	Clark

. . .

STUDENTS				
id	course	course_code	major	prof
1	Database	353	Comp Sci	Yu
4	Agile Dev	821	Comp Sci	Turner
		…		
5	Compiler	237	Comp Sci	Clark

Legal Extensions of R

- Relation extensions *r*(*R*) that satisfy the functional dependency constraints are called **legal relation states** (or **legal extensions**) of *R*. Let *course* → *course_code be the only FD for* Students

STUDENTS				
id	course	course_code	major	prof
1	Database	353	Comp Sci	Smith
2	Chem101	427	Chemistry	Turner
3	Database	353	Comp Sci	Clark

Legal

STUDENTS				
id	course	course_code	major	prof
1	Database	353	Comp Sci	Yu
4	Agile Dev	821	Comp Sci	Turner
5	Compiler	237	Comp Sci	Clark

Also legal

Violations in FD

- But in practice, it’s possible to have FD violations.

STUDENTS				
id	course	course_code	major	prof
1	Database	353	Comp Sci	Smith
2	Chem101	427	Chemistry	Turner
3	Database	353	Comp Sci	Clark

STUDENTS				
id	course	course_code	major	prof
1	Database	353	Data Science	Smith
2	Chem101	427	Chemistry	Turner
3	Database	353	Comp Sci	Clark

Questions

- Where are the FDs? Isn’t there a constraint?
- How should it be enforced? how come I seen it yet?

Notation and Terminology

Let $\underline{X \rightarrow Y}$ be a functional dependency on <u>relation R</u>

We say that

1. $X \rightarrow Y$ *holds* on R
2. X *functionally determines* Y
3. Y is *functionally dependent* on X
4. X is *determinant* of the dependency
5. Y is *dependent* of the dependency

Functional Dependencies

A *WORKS_ON* relation

- *Ssn* = social security number
- *Pnumber* = project number

Question:
What might be the FDs of *WORKS_ON*?

WORKS_ON

Ssn	Pnumber	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	Null

Functional Dependencies

- A *EMPLOYEE* relation
  - *SSn* = social security number
  - *Bdate* = birthday
  - *Dnumer* = department number
- Question: What might be the FDs of *EMPLOYEE?*

EMPLOYEE

Ename	<u>Ssn</u>	Bdate	Address	Dnumber
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston, TX	5
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston, TX	5
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4
Wallace, Jennifer S.	987654321	1941-06-20	291Berry, Bellaire, TX	4
Narayan, Ramesh K.	666884444	1962-09-15	975 Fire Oak, Humble, TX	5
English, Joyce A.	453453453	1972-07-31	5631 Rice, Houston, TX	5
Jabbar, Ahmad V.	987987987	1969-03-29	980 Dallas, Houston, TX	4
Borg, James E.	888665555	1937-11-10	450 Stone, Houston, TX	1

Functional Dependencies

- Question: What about Students? which columns might be dependent functionally?

STUDENTS				
ID	**Phone**	**Major**	**Prof**	**Grade**
1	237-4539	Comp Sci	Smith	A
2	427-7390	Chemistry	Turner	B
3	237-4539	Comp Sci	Clark	B
4	388-5183	Physics	James	A
5	371-6259	Decision Sci	Cook	C
6	823-7293	Mathematics	Lamb	B
7	823-7293	Mathematics	Bond	UN
8	237- 4539	Comp Sci	Cross	UN
9	839-0827	English	Broes	C

Functional Dependencies

R = {ID, Name, Code, Grade}

r(R) Instance A

ID	Name	Code	Grade
100	J	3550	A
200	X	3550	B
100	J	4540	B
100	J	4550	A

- ID → Name (OK),
- ID → Grade (not OK),
- ID → Code (not OK),
- ID, Name → Grade (not OK),
- ID, Code → Grade (OK).

r(R) Instance B

ID	Name	Code	Grade
100	J	3550	A
200	X	3550	B
100	J	4540	A
100	J	4550	A

- ID → Name (OK)
- ID → Grade (OK),
- ID → Code (not OK)
- ID, Name → Grade (not OK),
- ID, Code → Grade (OK).

Important: You can't infer FD's from a relations instances

Functional Dependencies

- Functional dependencies…
  1. Specify the semantics between attributes within a relation
  2. A collection of FDs from each relation together specify constraints across the entire relational schema
  3. This semantics is not captured by ER, which doesn't mention relationship between attributes

Designing FDs

- Recall: FD cannot be inferred automatically from a given relation extension r.

- So given a relation, where do its FDs come from? Where do we find it?
  - Deciding the FDs of a table is part of a design decision.
  - Defined explicitly by someone who knows the semantics of the attributes of R.

Designing FDs

Assume we need to define the FDs of this relation

STUDENTS					
ID	**Course**	**Phone**	**Major**	**Prof**	**Grade**

What can we know about the columns?

Could each ID have a unique phone number and major?

Which Columns are Related?

STUDENTS					
ID	**Course**	**Phone**	**Major**	**Prof**	**Grade**

- Every ID has a unique phone number and major?
  - We can say {ID} →{Phone , Major}

- Other relations between columns:
  - Every course has a unique professor {Course} → {Prof}
  - Every ID and course has a unique grade {ID , Course} → {Grade}

- Whenever the semantics of two sets of attributes in R indicate that a functional dependency should hold, we specify the dependency as a constraint.

Final Notations

- We may denote the attributes sets with/without curly brackets
  - With curly brackets, attributes are comma separated
  - {X,Y} = XY

- The order of the attribute sets doesn't matter
  - ZY = YZ
  - {Z,Y} = {Y, Z}

Topics So Far

1. What is a functional dependency?

2. What could decide the functional dependencies that hold among the attributes of a relation schema?

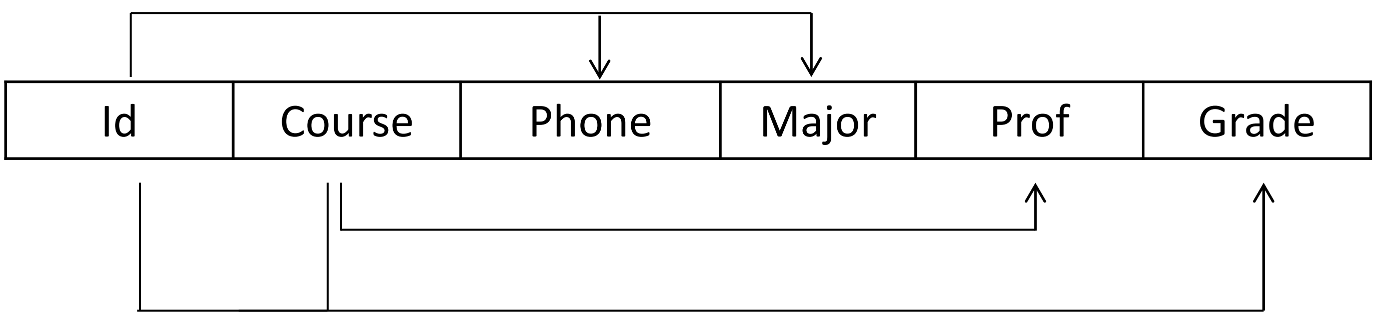
3. Why can we not infer a functional dependency automatically from a particular relation state?

4. Are there always functional dependencies in any relation?

Dependency Diagram

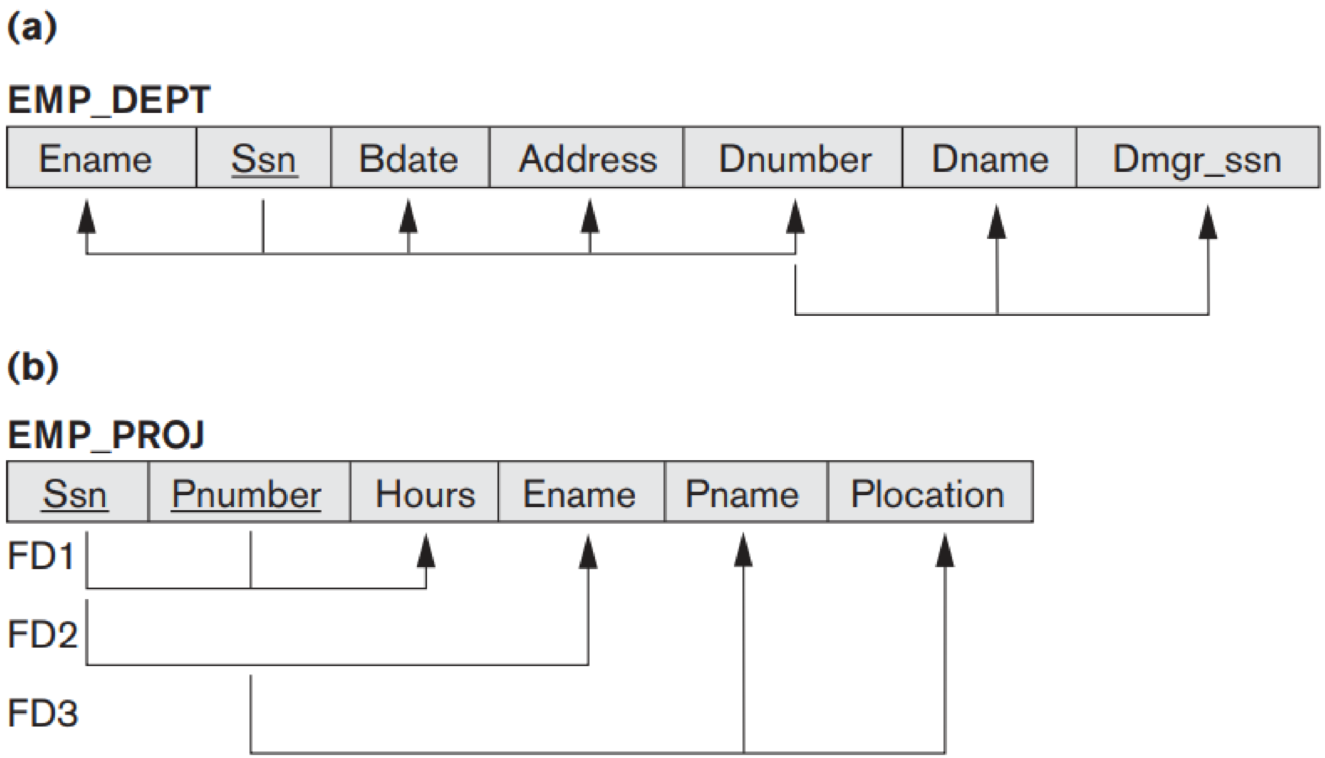
- Each *horizontal line* represents a FD
  - Left-hand side attr. connected by vert. lines to the horizontal line,
  - Right-hand side attr. connected by vert. lines ending with arrows
    - Arrow pointing toward the attributes

- Dependency diagram from previous example.



Dependency Diagram (Cont.)

- Some more examples of dependency diagrams.



Inferring other FDs - Q

- $A \rightarrow B$ and $B \rightarrow C$, what do we know about $A \rightarrow C$?

Inferring other FDs - A

- Given $A \rightarrow B$ and $B \rightarrow C$ on relation R,
- We know $A \rightarrow C$ holds on R, given A determines B, and B determines C.

- There may be additionally functional dependencies that also hold on R!

Infering Other FDs

- It's true that given a set F of functional dependencies, there are other functional dependencies that are logically implied by F.
- F |= X→ Y
- Denotes that set of FDs F infers X → Y if all relation instances satisfying F also satisfies X → Y .
- Example:
  F = $A \rightarrow B$, $B \rightarrow C$,
  F |= $A \rightarrow C$

- Usually, the schema designer will only specify the functional dependencies that are semantically obvious.

Inference Rules

- These are the inference rules for functional dependencies

  - Rule 1 (**reflexivity**)
    - if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$
  - Rule 2 (**augmentation**)
    - if $\alpha \rightarrow \beta$, then $\gamma \alpha \rightarrow \gamma \beta$
  - Rule 3 (**transitivity**)
    - if $\alpha \rightarrow \beta$, and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$
  - Where α, β, γ are all (nonempty) sets of attributes

- The above are the primary rules/axioms from **Armstrong's Axioms**

Practice

$R = (A, B, C, G, H, I)$

$F = \{ A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H \}$

These FDs can be inferred/ deduced.

$A \rightarrow H$

$AG \rightarrow I$

$CG \rightarrow HI$

Practice

$R = (A, B, C, G, H, I)$

$F = \{ A \rightarrow B, A \rightarrow C, CG \rightarrow H, CG \rightarrow I, B \rightarrow H \}$

These FDs can be inferred/ deduced.

$A \rightarrow H$

$AG \rightarrow I$

$CG \rightarrow HI$

But are they part of F?

Closure of F

- Note: We denote by *F* the set of functional dependencies that are specified on relation schema *R*.

- FDs (Functional dependencies) logically implied other FDs.
- The set of all FDs logically implied by F is the closure of F.
- F+ denotes the **closure** of F

- In general, given F, we can find F+ (the closure of F) by repeatedly applying Armstrong's Axioms.
- F+ is a superset of F.

Armstrong's Axioms:

- Armstrong's Axioms are proven to be sound and complete

- Notes:
  1. Sound = generates only functional dependencies that hold
  2. Complete = generates all functional dependencies that hold

Armstrong's Axioms (Cont.)

- Additional Rules we inferred from Armstrong's axioms.
  - Rule 4 (**additivity**):
    - If $\alpha \rightarrow \beta$ holds and $\alpha \rightarrow \gamma$ holds, then $\alpha \rightarrow \beta\gamma$ holds
  - Rule 5 (**projectivity**):
    - If $\alpha \rightarrow \beta\gamma$ holds, then $\alpha \rightarrow \beta$ holds and $\alpha \rightarrow \gamma$ holds
  - Rule 6 (**pseudo-transitivity**):
    - If $\alpha \rightarrow \beta$ holds and $\gamma\beta \rightarrow \delta$ holds, then $\alpha\gamma \rightarrow \delta$ holds
- **Other names:**
  Additivity aka Union
  Projectivity aka Decomposition