Lossless join - Execises

SAPIENZA
UNIVERSITÀ DI ROMA

Example 1

Sapienza Università di Roma
Dipartimento di Informatica

Given the following schema

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

and the following set of functional dependencies

$F = \{ C \rightarrow D, AB \rightarrow E, D \rightarrow B \}$

say whether decomposition

$\rho = \{ AC, ADE, CDE ,AD ,B \}$

has a lossless join

Example 1: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- let's start building the table:

	A	B	C	D	E
AC	a1	b12	a3	b14	b15
ADE	a1	b22	b23	a4	a5
CDE	b31	b32	a3	a4	a5
AD	a1	b42	b43	a4	b45
B	b51	a2	b53	b54	b55

Example 1: Verification

	A	B	C	D	E
AC	a_1	b_{12}	a_3	$b_{14} \rightarrow a_4{}^{(1)}$	b_{15}
ADE	a_1	$b_{22} \rightarrow b_{12}{}^{(2)}$	b_{23}	a_4	a_5
CDE	b_{31}	$b_{32} \rightarrow b_{12}{}^{(2)}$	a_3	a_4	a_5
AD	a_1	$b_{42} \rightarrow b_{12}{}^{(2)}$	b_{43}	a_4	b_{45}
B	b_{51}	a_2	b_{53}	b_{54}	b_{55}

F = {C→D, AB→E, D→B }

C → D: the first and third rows coincide on the attribute C=a_3, so we change b_{14} into a_4 so that the functional dependency is satisfied (if the rows are equal on C, they must be equal on D)

AB → E: the functional dependency is already satisfied, as there are not (yet) tuples equal on AB and different on E

D → B: in the first four lines D=a_4, so we change b_{22} into b_{12}, b_{32} into b_{12}, b_{42} into b_{12} (we could choose different b's, as long as we made them all the same)

we have completed the first execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 1: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

	A	B	C	D	E
AC	a1	b12	a3	a4	b15→a5
ADE	a1	b12	b23	a4	a5
CDE	b31	b12	a3	a4	a5
AD	a1	b12	b43	a4	b45→a5
B	b51	a2	b53	b54	b55

F = {C→D, AB→E, D→B }

C → D: the dependency is already satisfied

AB → E: the first, second and fourth rows coincide on the attributes AB = <a1, b12>, so we change b15 into a5 and b45 into a5, so that the functional dependency is satisfied (if the rows are equal on AB, they must be equal on E)

D → B: the functional dependency is already satisfied

we have completed the first execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 1: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

	A	B	C	D	E
AC	a1	b12	a3	a4	a5
ADE	a1	b12	b23	a4	a5
CDE	b31	b12	a3	a4	a5
AD	a1	b12	b43	a4	a5
B	b51	a2	b53	b54	b55

F = {C→D, AB→E, D→B }

C → D: no changes are necessary
AB → E: no changes are necessary
D → B: no changes are necessary

the table did not change, so the algorithm **ends**

since there is no row with all *a*'s, the join <u>IS NOT</u> lossless.

Example 2

- Given the relation scheme **R = ABCDEHI** and the set of functional dependencies:

$$\mathbf{F = \{ A \rightarrow B, B \rightarrow AE, DI \rightarrow B, D \rightarrow HI, HI \rightarrow C, C \rightarrow A\}}$$

- say whether decomposition:
- $\boldsymbol{\rho}$ **= { ACD, BDEH, CHI}** has a lossless join

Example 2: Verification

SAPIENZA
Università di Roma
Dipartimento di Informatica

F = { A → B, B → AE, DI → B, D → HI, HI → C, C → A}

	A	B	C	D	E	H	I
ACD	a1	b12	a3	a4	b15	b16	b17
BDEH	b21	a2	b23	a4	a5	a6	b27
WHO	b31	b32	a3	b34	b35	a6	a7

Example 2: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { A → B, B → AE, DI → B, D → HI, HI → C, C → A}

	A	B	C	D	E	H	I
ACD	a1	b12	a3	a4	b15	b16→a6$^{(1)}$	b17
BDEH	b21→a1$^{(3)}$	a2	b23→a3 (2)	a4	a5	a6	b27→b17$^{(1)}$
WHO	b31→a1$^{(3)}$	b32	a3	b34	b35	a6	a7

A → B: the dependency is already satisfied
B → AE: the dependency is already satisfied
DI → B: the dependency is already satisfied
D → HI: the first and second row coincide on the attribute D=a4, so we **separately** change H and I (b16 → a6 and b27 → b17)
HI → C: there are two tuples equal on HI (the first and second one, both with values <a6, b17>), so we change the value of C (b23 → a3)
C → A: the tuples are all equal on C, so we make them equal on A, and since we have the first one with value a, they all become a (b21 → a1, b31 → a1)

we have completed the first execution of the for and the table has been modified, so we iterate the for another time

Example 2: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { A → B, B → AE, DI → B, D → HI, HI → C, C → A}

	A	B	C	D	E	H	I
ACD	a1	b12→a2 (1)	a3	a4	b15→a5 (2)	a6	b17
BDEH	a1	a2	a3	a4	a5	a6	b17
WHO	a1	b32→a2 (1)	a3	b34	b35→a5 (2)	a6	a7

A → B: all the tuples are equal on A, so we make them equal on B, and since the second one is a, they all become a (b12→a2, b32→a2)

B → AE: all the tuples are equal on B, so they must be equal on AE; on A they are already equal, in the second tuple attribute E is a, so they all become a (b15→a5, b35→a5)

DI → B: the dependency is already satisfied

D → HI: the dependency is already satisfied

HI → C: the dependency is already satisfied

C → A: the dependency is already satisfied

we have completed the second execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 2: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { A → B, B → AE, DI → B, D → HI, HI → C, C → A}

	A	B	C	D	E	H	I
ACD	a1	a2	a3	a4	a5	a6	b17
BDEH	a1	a2	a3	a4	a5	a6	b17
WHO	a1	a2	a3	b34	a5	a6	a7

A → B: the dependency is already satisfied
B → AE: the dependency is already satisfied
DI → B: the dependency is already satisfied
D → HI: the dependency is already satisfied
HI → C: the dependency is already satisfied
C → A: the dependency is already satisfied

the table did not change, so the algorithm **ends**

since there is no row with all *a's*, the join IS NOT lossless

Example 3

given the following schema:

$$R = (A, B, C, D, E, G)$$

and the following set of functional dependencies:

$$F = \{ AB \rightarrow C, DG \rightarrow B, G \rightarrow D, E \rightarrow G \}$$

say whether the decomposition:

$$\rho = \{ ABD, AEG, BCE \}$$

has a lossless join

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22	b23	b24	a5	a6
ECB	b31	a2	a3	b34	a5	b36

Example 3: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22	b23	b24	a5	a6
ECB	b31	a2	a3	b34	a5	b36→a6 (1)

AB → C: the dependency is already satisfied

DG → B: the dependency is already satisfied

G → D: the dependency is already satisfied

E → G: the second and third rows coincide on the attribute E=a5, so we change G (b36→a6)

we have completed the first execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 3: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22	b23	b24	a5	a6
ECB	b31	a2	a3	b34→b24 (1)	a5	a6

AB → C: the dependency is already satisfied

DG → B: the dependency is already satisfied

G → D: the second and third rows coincide on the attribute G=a6, so we modify D (b34→b24)

E → G: the dependency is already satisfied

we have completed the second execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 3: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	b22→a2 (1)	b23	b24	a5	a6
ECB	b31	a2	a3	b24	a5	a6

AB → C: the dependency is already satisfied

DG → B: the second and third rows are equal <b24, a6>, so we modify B (b22 → a2)

G → D: the dependency is already satisfied

E → G: the dependency is already satisfied

we have completed the third execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 3: Verification

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	a2	b23 b13 (1)	b24	a5	a6
ECB	b31	a2	a3	b24	a5	a6

AB → C: the first and second row are equal on AB <a1, a2>, so we modify C (b23→b13)

DG → B: the dependency is already satisfied

G → D: the dependency is already satisfied

E → G: the dependency is already satisfied

we have completed the third execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 3: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { AB → C, DG → B, G → D, E → G }

	A	B	C	D	E	G
ABD	a1	a2	b13	a4	b15	b16
AEG	a1	a2	b13	b24	a5	a6
ECB	b31	a2	a3	b24	a5	a6

AB → C: the dependency is already satisfied

DG → B: the dependency is already satisfied

G → D: the dependency is already satisfied

E → G: the dependency is already satisfied

the table did not change, so the algorithm **ends**

since there is no row with all *a*'s, the join <u>IS NOT</u> lossless

Example 4

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

- given the schema:
- **R = ABCDEHI**
- and the set of functional dependencies:

F = { H → B, DI → H, D → I, B → E, E → C}

- say whether decomposition:
- **ρ = { ABDE, CDH, AHI}**
- has a lossless join

	A	B	C	D	E	H	I
ABDE	a1	a2	b13	a4	a5	b16	b17
CDH	b21	b22	a3	a4	b25	a6	b27
AHI	a1	b32	b33	b34	b35	a6	a7

Example 4: Verification

F = { H → B, DI → H, D → I, B → E, E → C}

	A	B	C	D	E	H	I
ABDE	a1	a2	b13	a4	a5	b16	b17→a7(3)
CDH	b21	b22	a3	a4	b25	a6	b27→b17(2) b17→a7(3)
AHI	a1	b32→b22(1)	b33→a3(5)	b34	b35→b25(4)	a6	a7

H → B: the second and third row are equal on H, so we modify B (b32→b22)

DI → H: the dependency is already satisfied

D → I: the first and second row are equal on D, so we modify I (b27→b17)

B → I: the second and third row are now equal (b22), so we modify I (b17(2)→a7; note that at the next iteration, by reapplying D→I also on the first tuple, we can transform b17→a7, so we can already change all the values that are equal)

B → E: the second and third row are equal on B (b22), so we modify E (b35→b25)

E → C: the second and third row are equal on E (b25), so we modify C (b33→a3)

we have completed the first execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 4: Verification

F = { H → B, DI → H, D → I, B → E, E → C}

	A	B	C	D	E	H	I
ABDE	a1	a2	b13	a4	a5	b16→a6 (1)	a7
CDH	b21	b22	a3	a4	b25	a6	a7
AHI	a1	b22	a3	b34	b25	a6	a7

H → B: the dependency is already satisfied

DI → H: the first and second row are equal on DI <a4, a7>; we modify H (b16→a6)

D → I: the dependency is already satisfied

B → I: the dependency is already satisfied

B → E: the dependency is already satisfied

E → C: the dependency is already satisfied

we have completed the second execution of the for and the table <u>has been modified</u>, so we iterate the for another time

Example 4: Verification

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

F = { H → B, DI → H, D → I, B → E, E → C}

	A	B	C	D	E	H	I
ABDE	a1	a2	b13→a3$^{(3)}$	a4	a5	a6	a7
CDH	b21	b22→a2$^{(1)}$	a3	a4	b25→a5$^{(2)}$	a6	a7
AHI	a1	b22→a2$^{(1)}$	a3	b34	b25→a5$^{(2)}$	a6	a7

H → B: all tuples are equal on H, so they should become equal on B (b22→a2)

DI → H: the dependency is already satisfied

D → I: the dependency is already satisfied

B → I: the dependency is already satisfied

B → E: all tuples are equal on B, so they should become equal on E (b25→a5)

E → C: all tuples are equal on E, so they should become equal on C (b13→a3)

we have completed the third for iteration and the table has been modified, so we should continue but ...

...since there is a row with all *a*'s (the first one), we can stop and the join IS lossless

Older example

let's reconsider the schema:

R = (Matriculation, Province, Municipality)

with the set of functional dependencies :

$F = \{Matriculation \rightarrow Province,\ Town \rightarrow Province\}$

the schema is not in 3NF due to the presence in F^+ of the partial dependencies *Matriculation→Province* and *Town→Province*, as the key is (*Matriculation, Town*)

let us reconsider the decomposition:

R1 = (Matriculation, Province) with *Matriculation→Province* and

R2 = (Province, Town) with *Town→Province.*

we remember that the schema, **while preserving all dependencies in *F*,** is not satisfactory

Older example

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

let us consider the **legal instance** of *R:*

R

Matriculation	Province	Town
501	Rome	Tivoli
502	Rome	Mandela

the two facts (501, *Rome, Tivoli)* and *(501, Rome, Mandela)* are true, and **no other**

based on the given decomposition, this instance decomposes into:

R1

Matriculation	Province
501	Rome
502	Rome

R2

Province	Town
Rome	Tivoli
Rome	Mandela

Remember our example

SAPIENZA
Università di Roma
Dipartimento di Informatica

- if we join the two legal instances resulting from the decomposition we get:

R

Matriculation	Town	Province
501	Rome	Tivoli
502	Rome	Mandela
501	Rome	Mandela
502	Rome	Tivoli

tuples unrelated to the reality of interest
so
loss of information

Let's check

- we rename the attributes A = Matriculation, B = Town, C = Province and we consider the schema R=ABC with the dependencies F={A→C, *B*→*C*}
- we check if the decomposition ρ **= {AC, BC} has a lossless join**

	A	**B**	**C**
AC	a1	b12	a3
BC	b21	a2	a3

- A → C: the dependency is already satisfied
- B → C: the dependency is already satisfied

- in practice, the table **is never modified** and **does not have a row with all *a***, so we obtain the same result that we verified empirically, that is, **the decomposition does not have a lossless join**