If you create tables from functional dependencies you must be sure that there is no extra, or redundant, dependency information in the set of functional dependencies.

If there is you will have duplicated facts in your database.

That is, you must first make sure your dependencies are a minimal cover.

We can find a minimal cover $F_m$ of F and a minimal using a fairly simple algorithm.

We present it here by example and will look at algorithms in more detail later in the course.

Apply decomposition rule so all dependencies have a single attribute for their right-hand side

## Find a minimal cover of F

F= {
ABD->AC,
B->E,
BA->E,
C->BE,
AD->FB,
C->E
}

=

F= {
ABD->A //decomposition
ABD->C //decomposition
B->E,
BA->E,
C->B, //decomposition
C->E, //decomposition
AD->F, //decomposition
AD->B, //decomposition
C->E
}

Remove any trivial dependencies (reflexive rules) duplicate dependencies or dependencies implied directly by others.

F = {

~~ABD->A~~ //reflexive

ABD->C

B->E,

~~BA->E~~, //implied by //previous rule

C->B,

C->E,

AD->F,

AD->B,

~~C->E~~ //duplicate

}

=

F = {

ABD->C

B->E,

C->B,

C->E,

AD->F,

AD->B,

}

Remove any unecessary attributes from the left-hand side of dependencies.

Remove any dependencies that a transitively implied by others

F= {

A~~B~~D->C //since AD->B

B->E,

C->B,

~~C->E~~, //transitive

AD->F,

AD->B,

}

=

F= {

AD->C

B->E,

C->B,

AD->F,

AD->B,

}

Create a set of 3NF tables from the minimal cover by combining depedencies that have the same left-hand side into a single tables.

The left hand side forms the key.

$F_m$ = {AD->C, AD->F, B->E,  C->B}

**3NF Tables:**

**<u>AD</u>CF**

**<u>B</u>E**

**<u>C</u>B**

## Exercise 1

Consider the set of attributes R={A,B,C,D,E,F} and the following set of functional dependencies proposed by the table designer.

$F_1$= {ABD->AC, B->E, BA->E, C->BE, AD->FB, C->E }

A colleague suggests that they use the following dependency set instead

$F_2$ = { AD->CF, C->B, B->E}

Determine if this is a reasonable suggestion

## Approach 1

If we can show the sets are equivalent then F2 can be used instead of F1

F1 and F2 are equivalent if $F1^+ = F2^+$

To show this we must show that each functional dependency in F1 is implied by the set F2 and vice versa each functional dependency in F2 is implied by the set F1

## Approach 1

As illustration we will show that:

ABD->AC from F1 is implied by set F2

and

AD->CF from F2 is implied by set F1

The same would have to be done for each functional dependency in each set, but for illustration here we only show the above two.

## Approach 1

**AD->CF from F2 is implied by set F1**

**Proof**

AD>-CF implied because :

AD->BF, given in F1
AD->F decomposition rule

ABD->AC given in F1
ABD->C decompostion rule

AD->B given in in F1
AD->ABD augmentation rule
AD- >C transitive rule : AD- >ABD, ABD->C

AD->CF union rule, AD->C, AD->F

$F_1$= {
ABD->AC,
B->E,
BA->E,
C->BE,
AD->BF,
C->E }

$F_2$ = {
AD->CF,
C->B,
B->E}

## Approach 1

**ABD->AC from F1 is implied by set F2**

**Proof**

ABD>-AC implied because :

AD->CF, given in F2

AD- >ACF augmentation rule (add A to both sides)

ABD->ABCF augmentation rule (add B to both sides)

ABD- >AC decomposition rules (ABD->AC, ABD->BF)

$F_1$= {
ABD->AC,
B->E,
BA->E,
C->BE,
AD->FB,
C->E }

$F_2$ = {
AD->CF,
C->B,
B->E}

## Approach 2

We can find a minimal cover $F_{m1}$ of $F_1$ and a minimal cover $F_{m2}$ of $F_2$ and show that these minimal covers are equivalent

Again we would have to show that $F_{m1}^+ = F_{m2}^+$ but the hope is that this would be trivial by inspection, or an easier problem than working with the "raw" dependency sets.

## Approach 2

**Find a minimal cover of F1**

$F_1$= {
- ABD->AC,
- B->E,
- BA->E,
- C->BE,
- AD->FB,
- C->E

}

=

$F_1$= {
- ABD->A //decomposition
- ABD->C //decomposition
- B->E,
- BA->E,
- C->B, //decomposition
- C->E, //decomposition
- AD->F, //decomposition
- AD->B, //decomposition
- C->E

}

## Approach 2

$F_1$= {

~~ABD->A~~ //reflexive

ABD->C

B->E,

~~BA->E~~, //implied by
//previous rule

C->B,

C->E,

AD->F,

AD->B,

~~C->E~~ //duplicate

}

=

$F_1$= {

ABD->C

B->E,

C->B,

C->E,

AD->F,

AD->B,

}

## Approach 2

$F_1$= {

A~~B~~D->C //since AD->B

B->E,

C->B,

~~C->E~~, //transitive

AD->F,

AD->B,

}

=

$F_1$= {

AD->C

B->E,

C->B,

AD->F,

AD->B,

}

# Approach 2

$F_1$= {

AD->C

B->E,

C->B,

AD->F,

~~AD->B~~, //transitive

}

=

$F_1$= {

AD->C

B->E,

C->B,

AD->F,

}

$F_{m1}$ = {AD->C, AD->F, B->E, C->B}

## Approach 2

**Find a minimal cover of F2**

$F_2$= {
AD->CF,
C->B,
B->E
}

=

$F_2$= {
AD->C, //decomposition
AD->F, //decomposition
C->B,
B->E
}

$F_{m2}$ = { AD->C, AD->F, C->B, B->E}

## Approach 2

$F_{m1}$ = {AD->C, AD->F, B->E,  C->B}

$F_{m2}$ = { AD->C, AD->F, C->B,  B->E}

By inspection $F_{m1} = F_{m2}$ so the sets $F_1$ and $F_2$ are equivalent

## Exercise 2

- For each of the following cases a relation R has been defined over attributes A,B,C,D,E,F along with a set of functional dependencies that apply to them.

- Find all the candidate keys for the relation
- State the highest normal form the table R=ABCDEF would currently satisfy
- Decompose the table until all resulting tables are in BCNF form

## Exercise 2

**F={ AB-> CDEF, EF ->C }**

## Exercise 2

**F={ AB-> CDEF, EF ->C }**

**Candidate keys: AB**

**Current Normal Form: = 2$^{nd}$ NF since EF->C violates 3$^{rd}$ NF.**

**Decomposition:**

**<u>AB</u>DEF**

**<u>EF</u>C**

## Exercise 2

F={ AB-> CDEF, EF->B , D->B }

## Exercise 2

F={ AB-> CDEF, EF->B , D->B }

Candidate keys: AB, AD, AEF

Current Normal Form: = 3rd NF since D->B violates BCNF.

Decomposition:

ADCEF

DB

(the dependency EF->B is lost going to BCNF)

## Exercise 2

**F={ AB-> CDEF, BC->D, }**

## Exercise 2

**F={ AB-> CDEF, BC->D, }**

**Candidate keys: AB**

**Current Normal Form: = $2^{nd}$ NF since BC->D violates $3^{rd}$ NF.**

**(Recall: Y->A is a transitive dependency if Y is neither a superkey of R nor a proper subset of a key of R)**

**Decomposition:**

**<u>AB</u>CEF**

**<u>BC</u>D**

## Exercise 2

F={ AB-> CDEF, B->C , D->C}

## Exercise 2

F={ AB-> CDEF, B->C , D->C}

Candidate keys: AB

Current Normal Form: = 1st NF since B->C violates 2nd NF.

Decomposition:

<u>AB</u>DEF

<u>B</u>C

<u>D</u>C