Join dependencies

MVDs are sometimes too weak an integrity constraint in database design – Join dependencies are stronger.

Consider the following all key relation in 4NF.

Supplier	***Item***	***Producer***
s1	Nut	p2
s1	Bolt	p1
s1	Nut	p1
s2	Nut	p1

Despite being in 4NF there are obvious problems with the table (insert anomalies and many repeating values / redundancy)

If we attempt a 2-decompostion (ref: Fagins theorem)

Supplier	***Item***
s1	Nut
s1	Bolt
s2	Nut

Supplier	***Producer***
s1	p2
s1	p1
s2	p1

Reconstructing the original scheme with a join:

Supplier	***Item***	***Producer***	
s1	Nut	p2	
s1	Bolt	p1	
s1	Bolt	p2	Problem here – a spurious relation
s1	Nut	p1	
s2	Nut	p1	

The 2-decompostion isn't reversible. Consider a 3-decompostion:

Item	***Producer***
Nut	p2
Bolt	p1
Nut	p1

Supplier	***Producer***
s1	Nut
s1	Bolt
s2	Nut

Supplier	***Item***
s1	Nut
s1	Bolt
s2	Nut

A natural join on the 3 decomposition gives

Supplier	***Item***	***Producer***	
s1	Nut	p2	
s1	Bolt	p1	No Spurious tuples
s1	Nut	p1	
s2	Nut	p1	

The supplier relation here is an example of the following observation:

Whilst MVDs guarantee the LJD of a relation scheme R into projections, there may exist a situation where there is no lossless 2 decomposition of R though there is a LJD into n projections where n>2.

Join dependencies (JDs) are stronger constraints that MVDs and guarantee lossless joins into n projections where n>2.

An MVD is a special case of a JD. JDs are tuple generating dependencies, so we could describe them in a form like

$t_1[X]=t_2[X]=t_3[X]=t_4[X]$
$t_1[X]=t_3[Y]$
$t_2[Y]=t_4[Y]$
... etc

However – it's easier to formalise the above definition:

Definition
If {R1,R2,…,Rn} is a set or relations produced by decomposing a relation scheme R then R satisfies the join dependency *[R1,R2,R3] iff every instance r of R has a LJD onto R1, R2,…,Rn

r must always equal the natural join of the projections of R onto R1, R2,…Rn

This is a computationally heavy definition as many joins are required to check for integrity constraints.

Attraction – this is the ultimate form of database dependency and leads a definition of fifth normal form.

Fifth Normal form
If J is a set of FDs and JDs one relation scheme R then R is in 5NF if for every JD *[R1,R2,…,Rn] in J+, JD is trivial or every R_i(i=1,2,…,n) is a superkey for R.

A JD *[R1,R2,…,Rn] on a relation scheme R is trivial if one of R_i is equal to R.

Again – easier to disprove than prove as computationally expansive to calculate J+.

Example
Let R=ABC, D={A->BC, C->B, *[AB, BC]}
Show that R is not in 5NF.

Clearly

A+={A,B,C}	Must be a superkey
B+={B}	Not a key
C+={C}u{B}={BC}	Not a key
AB+={A,B,C}	Candidate key (not minimal as A+ demonstrates)
BC+={BC}	Not a key

For R to be in 5NF both AB and BC must be superkeys – which they clearly are not.
<u>R is not in 5NF</u>

Exercise
Let R=ABCDEF, X={A->BCDE, BC->AF, *[ABC,BDEF]}
Show that R is not in 5NF.