Multi Value dependencies

FDs are the source of certain types of update anomaly but other types of data dependency may also cause problems – consider the following:

Emp_Name	***Project***	***Children***
Smith	p1	Bob
Smith	p2	Pat
Smith	p1	Pat
Smith	p2	Bob

If this is an all key relation then this must be in BCNF which implies that all anomaly problems should have been resolved.

Exercise
As the above relation is all key, what is the only non-trivial functional dependency?
Prove that the above relation is in BCNF.

There are however possible problems – consider inserting a tuple *fred, p1 with no children* (an insert anomaly). Deleting *Smith* also loses all information about project P1 and P2 (a delete anomaly). Modifying *Smith* for instance changing name, requires multiple changes (a modification anomaly).

This is a complex form of data dependency in that employee determines sets of projects, employee determines sets of children and projects and children are unrelated. Data dependencies of this type are called Multi-Value Dependency.

Definition
Given two sets of attributes X and Y, X multidetermines Y, X->>Y in a relation scheme R, if there is a distinct set of Y values associated with each X value and Y is not related to the attribute values in R-XY.

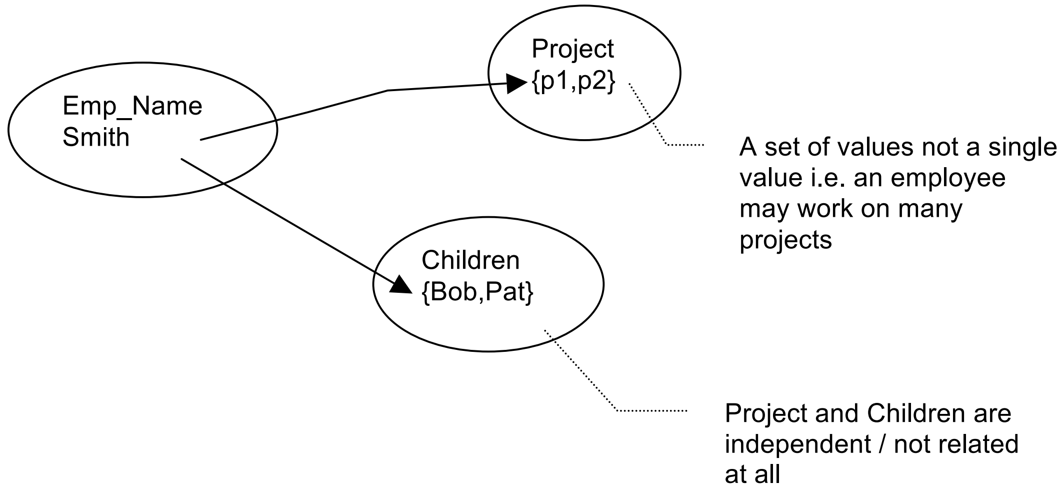
An MVD can also be expressed in terms of tuples

If R is a relation scheme, X and Y are subsets of R and Z=R-XY, then an instance of r of R satisfies the MVD X->>Y if whenever two tuples t_1 and t_2 exist in R such that $t_1[X]=t_2[X]$ then there exists two tuples t_3 and t_4 in R such that:

$t_1[X]=t_2[X]=t_3[X]=t_4[X]$
$t_1[Y]=t_3[Y]$
$t_2[Y]=t_4[Y]$
$t_2[Z]=t_3[Z]$ and
$t_1[Z]=t_4[Z]$

Note that t1, t2, t3 and t4are not necessarily distinct and that whenever X->>Y then so does X->>Z.

An MVD can be recognised by certain characteristics. Consider the employee relation:



There are two distinct 1:many relationships within the relation.

Another way of looking at these is as a nested relation:

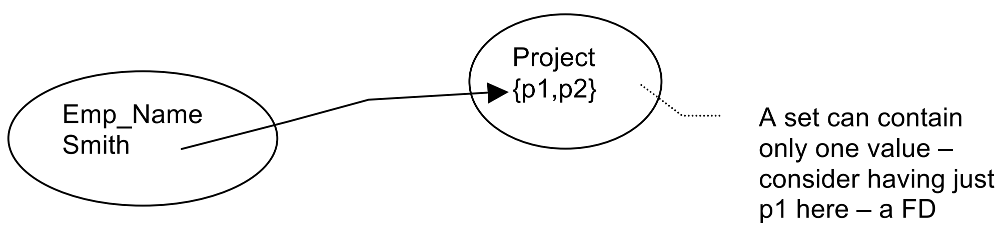
Emp_Name	***Project***	***Children***	
Smith	{p1,p2}	{Bob,Pat}	

Not in 1NF as non Atomic values

Sometimes called NF2 (normal form twice) or NFNF (Non first normal form).

Every FD is a MVD, i.e. FDs are just a special case of MVDs. Why?

Consider



Functional dependencies are known as 'equality generating dependencies' – if certain values are equal certain other values must be equal.

Consider the following scheme with A->B

A	***B***	***C***
a1	b1	c1
a1		c2

We can conclude the missing value from A->B - it must be b1 for A->B to hold.

MVDs are 'tuple generating dependencies' – if certain tuples are in a relation certain other tuples must also be in the relation.

Example
Does the MVD *Emp_name->>Children* hold?

Let	*X:* ***Emp_Name***	*Y:* ***Project***	*Z:* ***Children***
t_1=	Smith	p1	Bob
t_2=	Smith	p2	Pat
t_3=	Smith	p1	Pat
t_4=	Smith	p2	Bob

From the definition we can see

$t_1[X]=t_2[X]=t_3[X]=t_4[X]$
$t_1[X]=t_3[Y]$
$t_2[Y]=t_4[Y]$
$t_2[Z]=t_3[Z]$ and
$t_1[Z]=t_4[Z]$

So Emp_Name->>Children

Exercise
Does Emp_Name->>Project (as we expect)

Satisfying the MVD guarantees a non-loss 2-decomposition for the relation

Fagins theorem
If X->>Y holds (then X->Z holds) in a relation scheme R then R can be decomposed into

$R_1(XY)$ and $R_2(XZ)$

where R will be losslessly decomposed.

Example
Consider the 2-decomposition of Employee relation R into R_1 and R_2

Emp_Name	***Project***	***Children***
Smith	p1	Bob
Smith	p2	Pat
Smith	p1	Pat
Smith	p2	Bob

Emp_Name	***Project***
Smith	p1
Smith	p2

Emp_Name	***Children***
Smith	Bob
Smith	Pat

If we join R_1 and R_2 we return to R – a true non loss 2 decomposition

The closure M+ of an MVD is the basis for the definition of fourth normal form, 4NF, which is guaranteed to eliminate update anomalies due to FDs and MVDs.

Fourth Normal Form
A relation scheme R is in 4NF with respect to a set of MVDs M, if for every non-trivial MVD, X->>Y in M+, X is a superkey for R.

An MVD X->>Y on a relation scheme R is trivial if either Y is a subset of X or R=X u Y

Disproving the requirement for 4NF is easier than proving that a relation scheme is in 4NF as we do not have to compute M+.

Example
Is the Employee relation in 4NF?

Given Emp_Name->>Project we know that the relation is not in 4NF as Emp_name is not a superkey (Employee is an all key relation),

Exercise
Given R=(ABCDE) and M ={A->>BC, D->E}
and an instance of r={<a1,b1,c1,d1,e1>,<a1,b2,c2,d2,e2>}

How must r be extended to ensure a non loss 2 decomposition?