Lossless Decomposition& Dependency preservation

Lossless decomposition

Let *r* (*R*) be a relation schema, and let *F* be a set of functional dependencies on *r* (*R*). Let *R*1 and *R*2 form a decomposition of *R*.

A decomposition is said to be lossless when

$$\Pi_{R_1}(r) \bowtie \Pi_{R_2}(r) = r$$

```
select *
from (select R1 from r)
        natural join
        (select R2 from r)
```

Example

Employee(ID,name,street,city salary)

Employee is decomposed into 2 relations Employee1 and Employee2
Employee1(ID,name)
Employee2(name,street,city,salary)

Is this a lossy decomposition or lossless decomposition? Why?

Ans: Lossy

ID	Name	Street	City	Salary
1	John	MG road	Mumbai	30000
2	Lee	Clive road	Chennai	50000
3	John	Park road	Hyderabad	70000
4	Bill	Irwin street	Pune	90000

ID	Name
1	John
2	Lee
3	John
4	Bill

Name	street	City	salary
John	MG road	Mumbai	30000
Lee	Clive road	Chennai	50000
John	Park road	Hyderabad	70000
Bill	Irwin street	Pune	90000

NAME	ID	STREER	CITY	SALARY
John	1	MG road	Mumbai	30000
John	3	MG road	Mumbai	30000
Lee	2	clive road	chennai	50000
Bill	4	Irwin st	Pune	90000
John	1	park road	Hyderabad	70000
John	3	park road	Hyderabad	70000

Rule for Lossless Decomposition

We can use functional dependencies to show when certain decompositions are lossless. Let R, R_1, R_2, and F be as above. R_1 and R_2 form a lossless decomposition of R if at least one of the following functional dependencies is in F^+:

- $R_1 \cap R_2 \rightarrow R_1$
- $R_1 \cap R_2 \rightarrow R_2$

(ie) if $R_1 \cap R_2$ forms a superkey of either R_1 or R_2.

Then , the decomposition is lossless

inst_dept (ID, name, salary, dept_name, building, budget)

The inst_dept is decomposed into instructor and department

instructor (ID, name, dept_name, salary)
department (dept_name, building, budget)

R1 n R2= dept_name
Check whether
dept_name → instructor (or)
dept_name → department
holds true

dept_name → dept_name, building, budget

is true. Therefore the lossless decomposition rule is satisfied

Dependency preservation:

If we decompose a relation R into relations R1 and R2, All dependencies of R either must be a part of R1 or R2 or must be derivable from combination of FD's of R1 and R2.

For Example, A relation R (A, B, C, D) with FD set{A->BC, A->D} is decomposed into R1(ABC) and R2(AD)
which is dependency preserving because
FD A->BC is a part of R1(ABC). And
A->D is a part of R2(AD)

Algorithm for Dependency preservation

This algorithm is computationally Expensive.

```
compute F+;
for each schema Ri in D do
    begin
        Fi := the restriction of F+ to Ri;
    end
F' := Ø
for each restriction Fi do
    begin
        F' = F' ∪ Fi
    end
compute F'+;
if (F'+ = F+) then return (true)
              else return (false);
```

The **restriction** of F to R_i is the set F_i of all functional dependencies in $F+$ that include *only* attributes of R_i

Alternatives for checking Dependency preservation

Alternative1:

If each member of *F* can be tested on one of the relations of the decomposition, then the decomposition is dependency preserving. This is an easy way to show dependency preservation; however, it does not always work.

Reason:
There is a dependency in *F* that cannot be tested in any one relation in the decomposition

- It is just an sufficient condition.
- If it fails, we cannot conclude that the decomposition is not dependency preserving; instead we will have to apply the general test

Alternative2

The test applies the following procedure to each $\alpha \rightarrow \beta$ in *F*.

$result = \alpha$
repeat
    for each R_i in the decomposition
        $t = (result \cap R_i)^+ \cap R_i$
        $result = result \cup t$
until ($result$ does not change)

(Eg1)Consider a schema R(A,B,C,D) and functional dependencies A->B and C->D. Then the decomposition of R into R1(AB) and R2(CD) is

A. dependency preserving and lossless join
B. lossless join but not dependency preserving
C**. dependency preserving but not lossless join**
D. not dependency preserving and not lossless join

1. R1 n R2= null. Therefore it is not a lossless join
2. $A \rightarrow B$ is a part of R1 and $C \rightarrow D$ is a part of R2. Therefore FD are preserved

(Eg2)Let a relation R(A,B,C,D) and set a FDs F = { A -> B , A -> C , C -> D} are given.
A relation R is decomposed into –

R1 = (A, B, C) and R2 = (C, D)

R1 = (A, B, C) with FDs F1 = {A -> B, A -> C}, and
R2 = (C, D) with FDs F2 = {C -> D}.

F' = F1 ∪ F2 = {A -> B, A -> C, C -> D}

so, F' = F.

And so, F'+ = F+.

Thank you