TUTORIAL FOR WEEK 12 (Prof. Navathe)

RELATIONAL DESIGN ALGORITHMS

Q1. Binary Decomposition with Non-additive decomposition

Consider the relation
SUPPLY (Supplier#, Part#, Date, Project, Quantity, Supp_name, Part_name)
The FDs are:
Fd1: Supplier#, Part#, Date → SUPPLY
Fd2: Supplier# → Supp_name
Fd3: Part# → Part_name.
If we decompose SUPPLY into:

A. SUPPLY1 (Supplier#, Part#, Date, Project, Quantity, Part_name)
SUPPLIER (Supplier#, Supp_name)

- Have we preserved the Fds?
- Is this decomposition non-additive (lossless) – why?
- What NF is SUPPLY1 in?

B. Show a further decomposition of SUPPLY 1 and show that the decomposition is non-additive and achieves BCNF.

SOLUTION:

A.

(i) Yes, all of Fd1 , Fd2 and Fd3 are preserved in the design.

(ii) To test for non-additive decomposition, we apply the NJB test.
R1 = SUPPLY1; R2 = SUPPLIER.
R1 ∩ R2 = Supplier#
R2 – R1 = Supp_name
Because R1 ∩ R2 → (R2 – R1), we conclude that the decomposition is non-additive.

(iii)SUPPLY1 is still only in 1NF because of the Fd3 present in it.

B.

Further 2nd normalization of SUPPLY1 produces:
SUPPLY11 (Supplier#, Part#, Date, Project, Quantity) which preserves Fd1

PART (Part#, Part_name).

Now, R1 = SUPPLY11 and R2 = PART
R1 ∩ R2 = Part#
R2 – R1 = Part_name
Because R1 ∩ R2 → (R2 – R1), we conclude that the decomposition is non-additive.
Hence the **final design contains SUPPLIER, PART and SUPPLY11** where successive decompositions were non-additive.
Hence the final set of relations also posseses the non-additive join

property. Since all of them contain only FDs with LHS as the key, the design is in BCNF.

Q2. Relational Synthesis into 3NF relations.

Assume that we are given a universal relation corresponding to the data we have on students and courses which looks like:

STUDENT_COURSE (Stud#, Course#, St_name, Course_name, Course_credit_hr, Grade, Major_dept, Dept_phone_no)

The given set of F.d.s for the universal relation are:

F: {Stud#, Course# → St_name, Course_name, Course_credit_hr, Grade, Major_dept, Dept_phone_no;

Stud# → St_name, Major_dept, Dept_phone_no;

Course# → Course_name, Course_credit_hr

Major_dept → Dept_phone_no.}.

A. Apply the synthesis algorithm 15.4 that constructs 3NF relations from a given set of F.d.'s, on the universal relation. What 3NF design will be produced by this algorithm? Show how you get the answer
B. Evaluate your 3NF design and evaluate if it meets BCNF design.

SOLUTION:

PART A:

Given F:

{FD1: Stud#, Course# → St_name, Course_name, Course_credit_hr, Grade, Major_dept, Dept_phone_no;

FD2: Stud# → St_name, Major_dept, Dept_phone_no;

FD3: Course# → Course_name, Course_credit_hr

FD4: Major_dept → Dept_phone_no.},

Computing Min. Cover of F:

Note: We are skipping the second step of algo. 15.2 where each FD is decomposed to have only one attribute on RHS.

The minimal cover is:

F_{min} = {Stud# → St_name, Major_dept, Course# → Course_name, Course_credit_hr,

Major_dept → Dept_phone_no,

Stud#, Course# → Grade }

Why? Because,

(i) The part of FD1 based on the key (Stud#, Course#) on LHS **has extraneous attribute Course#** in order to functionally determine Stud#, Course# . Hence it is identical to FD2 and can be eliminated.

(ii) The part of FD1 based on the key (Stud#, Course#) on LHS **has extraneous attribute Stud#** in order to functionally determine Course_name, Course_credit_hr . Hence it is identical to FD3 and can be eliminated.

(iii) After removing these attributes on the RHS of FD1, it reduces to
Stud#, Course# → Grade.
That results in the above min. cover.

NOTE Also that the min cover in standard form contains FDs as follows:

Stud# → St_name,

Stud# → Major_dept,

Course# → Course_name,

Course# → Course_credit_hr.

We **REGROUP them into**

Stud# → St_name, Major_dept

And

Course# → Course_name, Course_credit_hr

Before going to form the 3NF design in step 3 of the .

Once, the above min.cover is obtained, the second step in algorithm 15.4 constructs one relation per FD in the min. cover, giving:

R1(Stud#, St_name, Major_dept)

R2 (Course#, Course_name, Course_credit_hr)

R3 (Major_dept , Dept_phone_no)

R4(Stud#, Course# , Grade)

PART B: Each of R1, R2, R3 and R4 contains an FD where the LHS is the key for that relation. There are no other known FDs. Hence, it meets the requirement of BCNF and hence the design is in BCNF.

Q3. BCNF Decomposition and n-ary non-additive decomposition test:

Consider a relation:

PATIENT_PROC (Patient#, Doctor#, Date, Doctor_name, Doctor_specialty, Procedure, Charge)

The Fds are:
FD1:Patient#, Doctor#, Date → PATIENT_PROC
FD2:Doctor# → Doctor_name, Doctor_specialty
FD3:Procedure → Doctor

A. Successive Normalization
- (i) Evaluate and explain the normal form status of the given relation.
- (ii) Follow the practice of successive normalization upto BCNF. For converting 3Nf to BCNF, apply the decomposition as per the decomposition in algorithm 15.5.

B. Direct testing and application of Algo 15.5
- (i) Argue using the BCNF general definition that PATIENT_PROC does not meet BCNF '
- (iii) . By applying decomposition in algorithm 15.5 successively produce the BCNF design.

C. Applying n-ary non-additive decomposition test in Algo 15.3

Show that the resulting relations you produced as BCNF design in A meet the non-additive join property using this algorithm.

SOLUTION:

(i) The FD2 shows non-full functional dependencies of the attributes Doctor_name and Doctor_specialty on Doctor# and hence is only in 1 NF.

(ii) Successive decomposition:

Second normalization:
PP1 (Patient#, Doctor#, Date, Procedure, Charge)
DOCTOR (Doctor#, Doctor_name, Doctor_specialty)

Third Normalization:
PP1 and DOCTOR are in 3NF .
PP1 is in 3 NF because it has 2 Fds:
FD11: Patient#, Doctor#, Date → Procedure, Charge
FD12: Procedure → Doctor#
FD11 meets the condition that LHS is superkey
FD12 meets the condition that RHS is prime attribute.

BCNF Normalization
However, FD12 violates BCNF.
Hence apply Algo 15.5 to PP1 and decompose it into:
PP11(Patient#, Date, Procedure, Charge) and
PP12 (Procedure, Doctor#).
Hence the final design is: DOCTOR, PP11 and PP12 tables.
NOTE: The dependency FD1 which is primary-key based FD **has been lost**.

B.

FIRST ATTEMPT (CASE X):

By Applying the BCNF definition directly, we see that PATIENT_PROC is not in BCNF because of
FD2 and FD3 which do not meet the requirement of LHS being a superkey. Suppose we fix the FD2 first. Then the first decomposition will produce:

PX1 (Patient#, Doctor#, Date, Procedure, Charge)
DOCTOR (Doctor#, Doctor_name, Doctor_specialty)

Now PX1 has the FDX1: Procedure → Doctor where LHS is not a superkey.
Hence we apply 15.5 and decompose into:
PX11 (Patient#, Date, Procedure, Charge) and
PX12 (Procedure, Doctor#)

Thus the final design is DOCTOR, PX11, PX12 which is identical with A.

<u>SECOND ATTEMPT (CASE Y):</u>
Suppose we fix FD3 first. Then the first decomposition produces:
PY1 (Patient#, Date, Doctor_name, Doctor_specialty, Procedure, Charge) and
PY2 ((Procedure, Doctor#)

PY1 now has the FD:
FY11: Patient#, Date, Procedure → Doctor_name, Doctor_specialty, Charge
And
FY12: Procedure → Doctor_name, Doctor_specialty.
Now, FY11 has an LHS that is a superkey;
However, FY12 does not meet BCNF. Hence we apply 15.5 to decompose PY1 as:
PY11(Patient#, Date, Procedure, Charge) and
PY12 (Procedure, Doctor_name, Doctor_specialty).
Thus the final BCNF design contains:

PY2, PY11,PY12. Notice that it is almost identical to the first case X except the Doctor# from first design X is replaced by Procedure in PY12 in the second design Y.

C.
TESTING FOR non-additive decomposition.

Set up the initial tableau for the 3 decomposed relations as 3 rows and one column for each attribute. A’s represent columns that are present in the relation. B’s represent columns that are absent in the relation.

The goal is to see if we can get any row of the tableau to have all a’s after the algorithm is applied. That proves that the n-ary decomposition is non-additive.

(iv)

P#	Doc#	Date	Dname	Doctor_	Procedu	Charge

					Specialty	re	
R1	b	a2	b	a4	a5	b	b
R2	a1	b	a 3	a4	b	a 6	a 7
R3	b	a2	b	b	b	a 6	b

R1: DOCTOR (Doctor#, Doctor_name, Doctor_specialty)

R2: PP11(Patient#, Date, Procedure, Charge)

R3: PP12 (Procedure, Doctor).

FIRST STEP: Because Procedure → Doctor and a6 → a2 is present in row3,
We can set Doc# column from b to a in R2

(v)	P#	Doc#	Date	Dname	Doctor_ Specialty	Procedure	Charge
R1	b	a2	b	a4	a5	b	b
R2	a1	~~b~~ a2	a 3	b	b	a 6	a 7
R3	b	a2	b	b	b	a 6	b

SECOND STEP: Because Doctor# → Doctor_name, Doctor_specialty and a2 → a4,a5 is present in row1, and now we have a2 present in row2, we can set Doc# column from b to a in R2

(vi)	P#	Doc#	Date	Dname	Doctor_ Specialty	Procedure	Charge
R1	b	a2	b	a4	a5	b	b
R2	a1	~~b~~ a2	a 3	~~b~~ a4	~~b~~ a 5	a 6	a 7
R3	b	a2	b	b	b	a 6	b

Now, entire row 2 for R2 has been set to a's. That concludes the procedure and determines that the R1, R2, R3 decomposition has the non-additive join property.

++++++++++++++++ END OF TUTORIAL 12 WITH SOLUTIONS +++++++++
++++++++++++++++++++