Relation Decomposition Algorithms

Source/References:
Database Systems: The Complete Book, and Elmasir/Navathe

pm jat @ daiict

Decomposition

- Relation Decomposition is done for deriving better normal form relations from a given relation

- Decomposition Algorithms
  - BCNF decomposition algo
  - 3NF synthesis algo

What is Decomposition?

- Let us say R(A1,A2,A3,A4,A5,A6) getting decomposed into R1(A1,A2,A3,A4) and R2(A4,A5,A6) – schema decomposition.

- Decomposition also mean instance decomposition, for above schema decomposition, we have instance r(R), getting decomposed as following –

  $\mathbf{r1} \leftarrow \pi_{a1,a2,a3,a4}(r)$, **and**

  $\mathbf{r2} \leftarrow \pi_{a4,a5,a6}(r)$

- Decomposition <u>should guarantee</u> **`r = r1 * r2`**

Example

- R(SSN, FNAME, SALARY, DNO, DNAME, MGRSSN)
  FD (Minimal) Set:
  SSN → FNAME, SALARY, DNO
  DNO → DNAME, MGRSSN (violates BCNF)
  Key: SSN.

ssn	fname	salary	dno	dname	mgrssn
103	Alicia	25000	4	Administration	102
104	Ramesh	38000	5	Research	101
105	Joyce	25000	5	Research	101
106	Ahmad	25000	4	Administration	102
107	John	30000	5	Research	101
101	Franklin	40000	5	Research	101
102	Jennifer	43000	4	Administration	102
108	James	56000	5	Research	101

r(R) decomposed into r1(R1) and r2(R2)

ssn	fname	salary	dno
103	Alicia	25000	4
104	Ramesh	38000	5
105	Joyce	25000	5
106	Ahmad	25000	4
107	John	30000	5
101	Franklin	40000	5
102	Jennifer	43000	4
108	James	56000	5

Decomposition should guarantee
`r1 * r2 = r`

dno	dname	mgrssn
4	Administration	102
5	Research	101

Requirements for relational decomposition

- Should be attribute preserving

- Should be lossless

- Should be FD preserving

Decomposition should be “lossless”

- As already stated, decomposition should guarantee

  `r = r1 * r2`

- When it does, then it is “loss-less join decomposition” or simply “loss-less decomposition”
- It is called “loss-less” because natural join of decomposed relation results into original relation, that is there is no loss of “information”
- Lossy decomposition normally results into additional tuples (also referred as spurious tuples) on having natural join of decomposed relations.

Lossless Decomposition Examples

- Given relation R(PNO, PNAME, ESSN, HOURS), and FDs-

  PNO → PNAME

  PNAME → PNO

  {PNO, ESSN} → HOURS

  {PNAME, ESSN} → HOURS

  Keys: {PNO, ESSN}, {PNAME, ESSN}

- Relation is in **3NF**
- Loss-lessly, there are two possible decompositions-

  R1(PNO, ESSN, HOURS), R2(PNO, PNAME), or

  R1(PNAME, ESSN, HOURS), R2(PNO, PNAME)

Lossless Decomposition Examples

SSN	PNO	PNAME	HOURS
101	1	PATR	38
101	2	XURT	20
102	1	PATR	64
103	2	XURT	58

SSN	PNO	HOURS
101	1	38
101	2	20
102	1	64
103	2	58

PNO	PNAME
1	PATR
2	XURT

SSN	PNAME	HOURS
101	PATR	38
101	XURT	20
102	PATR	64
103	XURT	58

PNO	PNAME
1	PATR
2	XURT

Decomposition Examples

- In given relation R(PNO, PNAME, ESSN, HOURS), let us say PName is no more unique; then our FD set is shortened to-

  PNO → PNAME

  {PNO, ESSN} → HOURS

  Keys: {PNO, ESSN}

  Relation is in **1NF**

- Now, with modified set of FDs, second decomposition is no more lossless, it is lossy?

  R1(PNO, ESSN, HOURS), R2(PNO, PNAME)

  R1(PNAME, ESSN, HOURS), R2(PNO, PNAME)

Decomposition Examples

SSN	PNO	PNAME	HOURS
101	1	PATR	38
101	2	XURT	20
102	1	PATR	64
103	3	XURT	58

SSN	PNO	HOURS
101	1	38
101	2	20
102	1	64
103	3	58

SSN	PNAME	HOURS
101	PATR	38
101	XURT	20
102	PATR	64
103	XURT	58

PNO	PNAME
1	PATR
2	XURT
3	XURT

PNO	PNAME
1	PATR
2	XURT
3	XURT

ON JOIN we do not get back original relation

Binary Lossless Decomposition

- A test for binary decomposition: Let us say R is decomposed into if R1 and R2, then it is lossless, when
- We have one of following FD
  (R1 intersection R2) → (R1-R2), or
  (R1 intersection R2) → (R2-R1), or
- Or in other words, <u>common attribute(s) are key of one of the decomposed relation</u>.

BCNF Decomposition Algorithm

(source: Ullman et al.)

- Input: A relation **R** with set (minimal) of FD's **F**.
- Output: A decomposition of R into a collection of relations, all of which are in BCNF.
- Method:

  T ← R, and G ← F

  Repeat till T is in BCNF.

  - If there is a FD X → Y that violates BCNF condition. Compute **X^+**, and choose X^+ as one relation as R1, and another R2 as **$\{(R - X^+) \cup X\}$**
  - Map FD set G on R1 and R2 (determine FDs on R1 and R2)

  – Recursively repeat the algo on R1 and R2

BCNF Decomposition Algorithm

(source: Ullman et al.)

- Replace relation R with two relations R1 and R2 such that

  $R_1 = X^+$ (with X as key).

  $R_2 = R - (X^+ - X)$, or $R - X^+ \cup X$.

- Project given FD's G onto the two new relations.

BCNF Decomposition Algorithm Worked Examples

BCNF Decomposition– Example #1

Given:

R(SSN,FNAME,PNO,PNAME,HOURS)

F={ {SSN, PNO} → HOURS; SSN → FNAME; PNO → PNAME }

Key: {SSN, PNO}

R

SSN	FNAME	PNO	PNAME	HOURS
101	Sumit	1	P-1	38
101	Sumit	2	P-2	20
102	Vipul	1	P-1	64
103	Ajay	2	P-2	58

FD SSN → FNAME
violates BCNF requirement;

SSN^+={SSN,FNAME}

R1(SSN, FNAME);
R2(SSN,PNO,PNAME,HOURS)

Decomposition of R based on FD **SSN →FNAME**

R2

SSN	PNO	PNAME	HOURS
101	1	P-1	38
101	2	P-2	20
102	1	P-1	64
103	2	P-2	58

FD set F2 :
{SSN, PNO} → HOURS
PNO → PNAME

R1

SSN	FNAME
101	Sumit
102	Vipul
103	Ajay

FD set F1:
SSN → FNAME

Decomposition of R2 based on FD PNO→PNAME

R2(SSN, PNO, PNAME, HOURS)

{SSN, PNO} → HOURS

PNO → PNAME

FD PNO → PNAME violates BCNF requirement;

PNO^+={PNO, PNAME}

R21(PNO, PNAME);
R22(SSN,PNO,HOURS)

R22

SSN	PNO	HOURS
101	1	38
101	2	20
102	1	64
103	2	58

FDs: {SSN, PNO} → HOURS

R21

PNO	PNAME
1	P-1
2	P-2

FDs: PNO → PNAME

Final BCNF decomposition

Given:

R(SSN,FNAME,PNO,PNAME,HOURS)

F={ {SSN, PNO} → HOURS,

SSN → FNAME, PNO → PNAME}

R1

SSN	FNAME
101	Sumit
102	Vipul
103	Ajay

FDs: SSN → FNAME

R22

SSN	PNO	HOURS
101	1	38
101	2	20
102	1	64
103	2	58

FDs: {SSN, PNO} → HOURS

R21

PNO	PNAME
1	P-1
2	P-2

FDs: PNO → PNAME

BCNF Decomposition– Example #2

- Consider relation **Medicine**(TradeName, GenericName, BatchNo, Stock, MRP, TaxRate, Manufacturer), and FDs
  TradeName → GenericName
  TradeName → Manufacturer
  BatchNo → TradeName
  BatchNo → Stock
  BatchNo → MRP
  GenericName → TaxRate
- Key: BatchNo
- FD TradeName → GenericName violates BCNF requirement, and $TradeName^+$ = {TradeName, GenericName, Manufacturer, TaxRate};
- Therefore, the relation **Medicine**(TradeName, GenericName, BatchNo, Stock, MRP, TaxRate, Manufacturer) decomposed to:

  Medicine1($TradeName^+$), and
  Medicine2({Medicine – $TradeName^+$} U TradeName)

BCNF Decomposition– Example #2

- New relations and their projected FDs you have as following.
- **Medicine1**(TradeName, GenericName, Manufacturer, TaxRate)
  - FDs

    TradeName → GenericName

    TradeName → Manufacturer

    GenericName → TaxRate
- **Medicine2**(BatchNo, TradeName, Stock, MRP)
  - FDs:

    BatchNo → TradeName

    BatchNo → Stock

    BatchNo → MRP
- Are both of these in BCNF ?

BCNF Decomposition– Example #2

- **Medicine1** is not in BCNF (highlighted FD violates BCNF requirement)
- **Medicine1**(<u>TradeName</u>, GenericName, Manufacturer, TaxRate)
  - FDs

    TradeName → GenericName

    TradeName → Manufacturer

    GenericName → TaxRate

- Repeat the process.

BCNF Decomposition– Example #2

- Compute GenericName+, and do decomposition
- **Medicine11**(<u>GenericName</u>, TaxRate)
  - FDs

    GenericName → TaxRate
- **Medicine12**(<u>TradeName</u>, GenericName, Manufacturer)
  - FDs

    TradeName → GenericName

    TradeName → Manufacturer
- Both are in BCNF ?

BCNF Decomposition– Example #2

- Here is final BCNF decomposition

Medicine11(GenericName, TaxRate)
Medicine12(TradeName, GenericName, Manufacturer)
Medicine2(BatchNo, TradeName, Stock, MRP)

- Based on semantics of attributes captured in each relation, we can name the final relations as following-

MedicineBatch(BatchNo, TradeName, Stock, MRP)

Medicine(TradeName, GenericName, Manufacturer)

GenericMedicine(GenericName, TaxRate)

BCNF Decomposition– Example #3

- Determine which Normal Form following relation is?
- Try bringing it in to BCNF. Decomposition should be lossless and should preserve all FDs

Book(ISBN, Title, Author, Publisher, Price, AccessonNo);

FDs:

AccessonNo → ISBN

ISBN → Title, Publisher, Price

Key: ?

BCNF Decomposition– Example #3

Book(ISBN,Title,Author,Publisher,Price, AccessonNo);

FDs:

AccessonNo → ISBN
ISBN → Title, Publisher, Price

- Key: **AccessonNo, Author**
- Relation is not in BCNF. Let us decompose using BCNF decomposition algo

BCNF Decomposition– Example #3

Book(ISBN,Title,Author,Publisher,Price, AccessonNo);

FDs:

AccessonNo → ISBN
ISBN → Title, Publisher, Price

Key: **AccessonNo, Author**

FD ISBN → Title violates BCNF requirement, and we decompose as following -

R1(ISBN,Title,Publisher, Price)

R2(AccessonNo,Author,ISBN)

BCNF Decomposition– Example #3

Book(ISBN,Title,Author,Publisher,Price,AccessonNo);

Decomposed to:

R1(ISBN,Title,Publisher, Price)
R2(AccessonNo,Author,ISBN)

FDs on R1

ISBN → Title, Publisher, Price

ISBN is the key and the relation is in BCNF.

FDs for R2:

AccessonNo → ISBN

Key for R2: AccessonNo, Author and therefore R2 is not in BCNF.

BCNF Decomposition– Example #3

Book(ISBN,Title,Author,Publisher,Price, AccessonNo);

FDs:

AccessonNo → ISBN
ISBN → Title, Publisher, Price

Key: AccessonNo, Author

- Final BCNF decomposition-

R1(<u>ISBN</u>,Title,Publisher, Price)

R21(<u>AccessonNo</u>, ISBN)

R22(<u>Author, AccessonNo</u>)

BCNF Decomposition Algorithm

- BCNF decomposition algorithm that we have seen here-
  - Guarantees loss-less decomposition.
  - However this does not guarantee Dependency Preservations

- Here are few such example (following slides)-

BCNF Decomposition Algorithm

- Consider our example #1 again. We have given relation R(PNO, PNAME, ESSN, HOURS), and FDs

  PNO → PNAME //FD1

  PNAME → PNO //FD2

  {PNO, ESSN} → HOURS //FD3

  {PNAME, ESSN} → HOURS //FD4

- Both of following are loss-less BCNF decompositions but are not FD preserving-

  R1(PNO, ESSN, HOURS), R2(PNO, PNAME) //Loose FD4

  R1(PNAME, ESSN, HOURS), R2(PNO, PNAME) //Loose FD3

BCNF Decomposition– Example #4

Given:

R(ABC)

AB→C

C→B

Key: AB, AC

- FD C → B violates BCNF requirement

  C^+=BC; R1(BC); R2(AC); F1{C→B}; F2{}.

- Though both relations are in BCNF, we loose FD AB → C

3NF Synthesis Algorithm and worked Examples

3NF Synthesis Algorithm

- “3-NF Synthesis” algorithm that guarantees lossless and preserves FDs.

- however its resultant relations may not be in BCNF, but surely in 3NF.

Algorithm – 3NF synthesis

- Find a minimal FD* set on R.

- For each set of FDs where X is determinant; say $X \rightarrow A1$, $X \rightarrow A2$, … $X \rightarrow An$; create a relation schema R'(X U A1 U A2 U … U An).

- If none of the relation created in above step that contains key of R; then create one more relation schema for key of R.

- *Note: also referred as “minimal base”, “minimal cover”, “canonical cover”

3NF Synthesis – Example#1

- Given relation R(ABCDEF), and FDs

  A → B

  A → C

  C → D

  C → E
- Key: AF
- 3NF Decomposition

  R1(ABC)

  R2(CDE)

  R(AF)

3NF Synthesis – Example#2

Book(ISBN,Title,Author,Publisher,Price, AccessonNo)

FDs:

AccessonNo → ISBN
ISBN → Title, Publisher, Price

Key: AccessonNo, Author

Decomposition:

R1(<u>AccessonNo</u>,ISBN)
R2(<u>ISBN</u>,Title,Publisher,Price)
R3(AccessonNo,Author)

3NF Synthesis – Example#3

- Given relation R(A,B,C,D,E), and FDs

  $\{A,B\} \rightarrow C$

  $C \rightarrow B$

  $A \rightarrow D$

Key: ABE

- Using 3NF synthesis algo, we have following decomposition-

  R1(ABC), ~~R2(CB)~~; R3(AD)

- Since, R2 we do not need because it is subset of R1, and none of determined relation contains the key, we add a relation R4(ABE). Finally we have following 3NF relations –

  R1(ABC), R3(AD), R4(ABE)

Requirements for relational decomposition (conclude)

- Should be attribute preserving
- Should be lossless
- Should be FD preserving

Notion of De-normalization

- We accept less normal form than BCNF for efficiency reasons.

Exercises

Exercises

- For following relations, find out FDs (use your understanding of the attribute semantics); find out NF, decompose to best possible normal form without any information and FD loss.
- LibMember(ID, Name,Type, NoOfBooksCanBeIssued,IssueDuration)
- R (StudentID, SPI, CPI_UptoDate, CPI_UptoASem, AcadYr, Sem, ProgCode, CourseNo, Grade)
- IssueLog(IssueDate, MemberID, AccessonNo, DueDate, ReturnDate)
- R(A,B,C,D,E); FDs {AB $\rightarrow$ C, C $\rightarrow$ B, A $\rightarrow$ D}

Sources/References

- Database Systems: The Complete Book, by
  Hector G. Molina, Jeff Ullman, and Jennifer Widom
  Pearson Education, India
  Home Page:
  http://infolab.stanford.edu/~ullman/dscb.html

- Elmasir/Navathe