Program Name : Masters of Computer Application

Semester : 2

Paper Title : Database Management System(DBMS)

Submitted by : Rishi Raj Chaurasia

Examination Roll Number : 19234757031

Assignment 4 : Normalization and Transactions

1. **Given a relation R (A, B, C, D, E) and F= (A->B, BC->D, D->BC, DE->Φ), synthesise a set of 3NF relation schemes**.

Sol.

Candidate Key of R is {A, D, E}.

Step 1: Find minimal cover

F = {A->B, BC->D, D->BC, DE->Φ}

Reduce RHS :-
F = {A->B, BC->D, D->B, D->C}

Reduce LHS :-
BC+ = {B,C,D}
B+ = {B,C,D}
C+ = {B,C,D}

Replace BC->D with B->D :-
F1 = {A->B, B->D, D->B, D->C}

Minimal Cover = {A->B, B->D, D->BC}

Step 2: Create relation for each FD

R1={A,B},
R2={B,D},
R3={D,B,C},

Step 3: Check if Key attributes are present in any relation, if not then make a new relation for it.

R1 = {A, B}
R2 = {B, D}
R3 = {B, C, D}
R4 = {A,D,E}

Step 4: Remove any redundant relations.

R2 is redundant, remove it.

Final Result: R1 (A,B), R3 (B, C, D), R4 (A, D, E) is the obtained 3NF relation schemes.

2. **Define BCNF. How does it differ from 3NF? Why is it considered a stronger from 3NF? Provide an example to illustrate.**

Sol.

Definition: A relation schema R is in BCNF if whenever a nontrivial functional dependency X → A holds in R, then X is a superkey of R.

BASIS FOR COMPARISON	**3NF**	**BCNF**

Concept	No non-prime attribute must be transitively dependent on the Candidate key.	For any trivial dependency in a relation R say X->Y, X should be a super key of relation R.
Dependency	3NF can be obtained without sacrificing all dependencies.	Dependencies may not be preserved in BCNF.
Decomposition	Lossless decomposition can be achieved in 3NF.	Lossless decomposition is hard to achieve in BCNF.

BCNF is stronger than 3NF because,

1. Every BCNF relation is in 3NF but not vice-versa. BCNF is stronger than 3NF.
2. In BCNF, any trivial dependency X -> Y in R holds if X is a super key in R. While in 3NF no non-prime attribute can be transitively dependent on the candidate key.

Example:
Consider the following universal relation:
R (Property_id#, County_name, Lot#, Area,Price,Tax_rate).

The following dependencies are present:
FD1: Property_id# -> {County_name, Lot#, Area,Price,Tax_rate}
FD2: County_name, Lot# -> {Property_id#, Area,Price,Tax_rate}
FD3: Area -> {Price}
FD4:County_name -> {Tax_rate}
FD5:Area -> {County_name}

After decomposition in 3NF
R1:{ **Property_id#,**County_name, Lot#, Area }
R2: { **Area**,Price}
R3:{**County_name**,Tax_rate}

After decomposition in BCNF:
R1:{ **Property_id#,**County_name, Lot#}
R2: { **Area**,Price}
R3:{**County_name**,Tax_rate}
R4:{**Area**,County_name}

3. Following relation is given:
STUDENT (COURSE, STUDENT, FACULTY, TERM, GRADE)
Each student receives only one grade in a course during a terminal examination. A student can take many courses and each course can have more than one faculty in a terminal.

a. Define the FDs and MVDs in the relation.

FD1: STUDENT, COURSE -> GRADE
FD2: COURSE -> TERM
MVD1: STUDENT ->> COURSE
MVD2: COURSE ->> FACULTY

b. Is the relation in 4NF? If not, decompose the relation.

No, the above relation is not in 4NF.

Decomposing the relation into 4NF

R1 = (**STUDENT, COURSE,** GRADE)

R2 = (**STUDENT, COURSE**)

R3 = (**COURSE**, **FACULTY**)

R4 = (**COURSE,** TERM)

4. Consider the relation SUPPLIES given as:

SUPPLIES (SUPPLIER, PART, CONTRACT, QTY)
CONTRACT->PART
PART->SUPPLIER
SUPPLIER, CONTRACT->QTY

Now the above relation is decomposed into the following two relations:
SUPPLIERS (SUPPLIER, PART, QTY)
CONTRACTS (CONTRACT, PART)

a. **Explain whether any information has been lost in this decomposition.**

Step 1: Create an initial matrix S for attributes (column) and relations (rows).

	Supplier	**Part**	**Contract**	**Qty**
R1	b11	b12	b13	b14
R2	b21	b22	b23	b24

Step 2: SUPPLIERS (SUPPLIER, PART, QTY) , CONTRACTS (CONTRACT, PART)

	Supplier	**Part**	**Contract**	**Qty**
R1	a	a	b13	a
R2	b21	a	a	b24

Step 3: CONTRACT->PART, PART->SUPPLIER, SUPPLIER, CONTRACT->QTY

	Supplier	**Part**	**Contract**	**Qty**
R1	a	a	b13	a
R2	~~b21~~ a	a	a	b24

Since, No row contains all “a”, this is not lossless decomposition**.**

b. **Explain whether any information has been lost if the decomposition is changed to SUPPLIER(CONTRACT,PART,QTY)**
CONTRACTS (CONTRACT, SUPPLIER)

Step 1:

	Supplier	Part	Contract	Qty
R1	b11	b12	b13	b14
R2	b21	b22	b23	b24

Step 2:

	Supplier	Part	Contract	Qty
R1	b12	a	a	a
R2	a	b22	a	b24

Step 3:

	Supplier	Part	Contract	Qty
R1	~~b12~~ a	a	a	a
R2	a	~~b22~~ a	a	~~b24~~ a

Here, all row contains "a", this decomposition is lossless.

5. Let us assume that the following is given:
Attribute set R = ABCDEFGH
FD set of F={AB->C, AC->B, AD->E, B->D, BC->A, E->G}
Which of the following decompositions of R=ABCDEG, with the same set of dependencies F, is dependency-preserving and lossless-join?

a. {AB, BC, ABDE, EG}

For Dependency preservation:

R1: (AB) covers nothing.

R2: (BC) covers nothing.

R3: (ABDE) covers AD -> E and B->D.

R4 (EG) covers E->G

This decomposition doesn't preserve all the FDs.

For lossless-join test:

Step1:

	A	B	C	D	E	F	G
R1	b11	b12	b13	b14	b15	b16	b17
R2	b21	b22	b23	b24	b25	b26	b27
R3	b31	b32	b33	b34	b35	b36	b37
R4	b41	b42	b43	b44	b45	b46	b47

Step2:

	A	B	C	D	E	F	G
R1	a	a	b13	b14	b15	b16	b17
R2	b21	a	a	b24	b25	b26	b27
R3	a	a	b33	a	a	b36	b37
R4	b41	b42	b43	b44	a	a	b47

Step 3:

	A	B	C	D	E	F	G
R1	a	a	~~b13~~ a	~~b14~~ a	~~b15~~ a	b16	~~b17~~ a
R2	~~b21~~ a	a	a	~~b24~~ a	~~b25~~ a	b26	~~b27~~ a
R3	a	a	~~b33~~ a	a	a	b36	~~b37~~ a
R4	b41	b42	b43	b44	a	b46	a

This is not a lossless decomposition because no row contain all "a".

b. {ABC, ACDE, ADG}

Dependency Preservation Test:
R1 (ABC) covers AB-> C, AC-> B, BC->A
R2 (ACDE) covers AD -> E
R3 (ADG) covers nothing.

This Decomposition does not preserve all dependencies.

Lossless-Join Test:
Step 1:

	A	B	C	D	E	F	G
R1	b11	b12	b13	b14	b15	b16	b17
R2	b21	b22	b23	b24	b25	b26	b27
R3	b31	b32	b33	b34	b35	b36	b37

Step 2:

	A	B	C	D	E	F	G
R1	a	a	a	b14	b15	b16	b17
R2	a	b22	a	a	a	b26	b27
R3	a	b32	b33	a	b35	b36	a

Step 3

	A	B	C	D	E	F	G
R1	a	a	a	~~b14~~ a	~~b15~~ a	b16	~~b17~~ a
R2	a	~~b22~~ a	a	a	a	b26	~~b27~~ a
R3	a	b32	b33	a	~~b35~~ a	b36	a

It is not a lossless join(no row contains all "a").

6. Explain the concepts of serial, non-serial and serializable schedules. State the rules for equivalence of schedules.

Serial Schedule: In serial schedule,

a. The operations of each transaction are executed consecutively, without any interleaved operations from the other transaction.
b. Entire transactions are performed in serial order.
c. In a serial schedule, only one transaction at a time is active—the commit of the active transaction initiates execution of the next transaction.

Serial schedule involving T1 and T2 transactions is given below:

T1	T2

read_item(X);
X := X – N
write_item(X);
	read_item(X);
	X := X + M
	write_item(X);

In the above example, first T1 gets executed, updates value of X and then T2 starts its execution.

Non-Serial Schedule: In a non-serial schedule,

a. Transactions execute in a non-serial order
b. In non-serial schedule, concurrency issues may occur.
c. Some transaction operations may interfere with the operations of other transactions.

T1	T2

read_item(X);
X := X – N
write_item(X);
read_item(Y);
Y = Y+N;
write_item(Y);
	read_item(X);
	X := X + M
	write_item(X);

The above example is a non-serial schedule with interleaving of operations.

Serializable Schedule: A schedule S is serializable if it is equivalent to some serial schedule of the same n transactions.

Following are the three rules of equivalence of schedules:-

• If two transactions only read a data item, they do not conflict and order is not important.

• If two transactions either read or write completely separate data items, they do not conflict and order is not important.
• If one transaction writes a data item and another either reads or writes the dame data item, the order of execution is important.

7. Consider the following decompositions for the relation schema R = {A, B, C, D, E, F, G, H, I, J} and the set of functional dependencies.
F = {AB ->C, A -> DE, B ->F, F->GH, D ->IJ}.
What is the key for R? Decompose R into 2NF, then 3NF relations.
Determine whether each decomposition has
(i) The dependency preservation property,
(ii) The lossless join property, with respect to F. Also determine which normal form each relation in the decomposition is in.

Solution:

Finding Key of the Relation:

Since there is no incoming edge on A and B,
$(AB)^+$ = **{A, B, C, D, E, F, G, H, I, J} = R**
Thus, AB is Key of the relation.

Decomposing R into 2NF:

R_1 = {**A**, D, E, I, J}
R_2 = {**B**, F, G, H}
R_3 = {**A, B**, C}

Decomposing R into 3NF:

R_1 = {**A**, D, E}
R_2 = {**B**, F}
R_3 = {**A, B**, C}
R_4 = {**F**, G, H}
R_5 = {**D**, I, J}

(i). F = {AB ->C, A -> DE, B ->F, F->GH, D ->IJ}

R1 covers A->DE
R2 covers B->F
R3 covers AB->C
R4 covers F->GH
R5 covers D->IJ

Thus, all FDs are preserved.

(ii). Lossless join Test

Step 1: Create an initial matrix S for attributes (column) and relations (rows).

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
R1	b	b	b	b	b	b	b	b	b	b
R2	b	b	b	b	b	b	b	b	b	b
R3	b	b	b	b	b	b	b	b	b	b
R4	b	b	b	b	b	b	b	b	b	b
R5	b	b	b	b	b	b	b	b	b	b

Step 2: Mark “a” instead of “b” in all rows where:-

R_1 = {**A**, D, E}
R_2 = {**B**, F}
R_3 = {**A, B**, C}
R_4 = {**F**, G, H}
R_5 = {**D**, I, J}

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
R1	a	b	b	a	a	b	b	b	b	b
R2	b	a	b	b	b	a	b	b	b	b
R3	a	a	a	b	b	b	b	b	b	b
R4	b	b	b	b	b	a	a	a	b	b
R5	b	b	b	a	b	b	b	b	a	a

Step 3: F = {AB ->C, A -> DE, B ->F, F->GH, D->IJ}.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**
R1	a	b	b	a	a	b	b	b	a	a
R2	b	a	b	b	b	a	a	a	b	B
R3	a	a	a	a	a	a	a	a	a	a
R4	b	b	b	b	b	a	a	a	b	b
R5	b	b	b	a	b	b	b	b	a	a

This is a lossless join decomposition because R3 is all filled with “a”.
Relation is 3NF.