65

EXPERIMENT - 2

AIM: To study lossy and lossless decomposition in DBMS.

THEORY: Decomposition in Database Management System is to break a relation into multiple relations to bring it into an appropriate normal form. It helps to remove redundancy, inconsistencies, and anomalies from a database. The decomposition of a relation R in a relational scheme is the process of replacing the original relation R with two or more relations in a ~~re~~ relational schema. Each of these relations contains a subset of the attributes of R and together they include all attributes of R.

⇒ Hierarchial decomposition: Hierarchial decomposition in database management system (DBMS) is an approach to organizing and structuring data based on a hierarchical model. It involves breaking down complex data into smaller, more managable units using a parent - child relationship.
In a hierarchical ~~or~~ decomposition, data is organized in a tree-like structure, where each node represents a record or entity, and the links between nodes represent the parent-child relationship. The top-level node is called the root and subsequent nodes are children of their respective parent nodes.

⇒ Lossy Decomposition :

When a relation is decomposed into two or more relational schema, the loss of information is unavoidable when the original relation is retrieved. Let us see an example—

Emp_ID	Emp_Name	Emp_Age	Emp_Location	Dept_ID	Dept_Name
E001	James	28	Paris	Dept1	operations
E002	Smith	31	Ukraine	Dept 2	HR
E003	Alex	33	Texas	Dept3	Finance

Decompose the above table into two-tables:

i) <Emp_Details>

Emp_ID	Emp Name	Emp_Age	Emp_location
E001	James	28	Paris
E002	Smith	31	Ukraine
E003	Alex	33	Texas

ii) <Dept_Details>

Dept_ID	Dept_Name
Dept1	operations
Dept2	HR
Dept3	Finance

Now you won't be able to join the above tables, since "Emp-ID" isn't part of the "Dept Details" relation. Therefore, the above relation has lossy decomposition.
Therefore, the above relation has lossy decomposition.

⇒ Lossless Decomposition:
Decomposition is lossless if it is feasible to reconstruct relation R decomposed tables using joins. This is the preferred choice. The information will not lose from the relation when decomposed. The join would result in the same original relation.
Let us see an example:
<Emp Info>

Emp_ID	Emp_Name	Emp_Age	Emp_location	Dept_ID	Dept_Name
E001	James	28	Paris	Dept1	Operations
E002	Smith	31	Ukraine	Dept2	HR
E003	Alex	33	Texas	Dept3	Finance

Decompose the above table into two tables:

i) <EmpDetails>

EmpID	EmpName	Emp_Age	Emp_location
E001	James	28	Paris
E002	Smith	31	Ukraine
E003	Alex	33	Texas

ii) <DeptDetails>

Dept_ID	Emp_ID	Dept_Name
Dept1	E001	operations
Dept2	E002	HR
Dept3	E003	Finance

Now, natural join is applied on the above two tables, the result will be the original table.
Therefore, the above relation had lossless decomposition i.e no loss of information.

⇒ Identification between lossless and lossy decomposition:

Lossless	Lossy
1) The decomposition $R_1, R_2, R_3 \cdots R_n$ for a relation schema R is lossless if these natural join results the original relation R. $R_1 \bowtie R_2 \bowtie R_n = R$	1) The decomposition $R_1, R_2, R_3 \cdots R_n$ for a relation schema R is lossy if these natural join results into addition of extraneous tuples with original relation R. $R \subset R_1 \bowtie R_2 \bowtie R_3 \cdots \bowtie R_n$
2) There is no loss of information as the relation obtained after natural join of decomposition is equilateral to original relation. Thus, it is also referred as non-additive join decomposition.	2) There is loss of information as extraneous tuples are added into the relation after natural join of decomposition. Thus, it is also referred as careless decomposition.

3) The common attribute of the sub relations is a superkey of any one of the relation.	3) The common attribute of the sub relation is not a superkey of any of the sub-relation.

⇒ Lossy and Lossless decomposition for BCNF:

Let there be a relation M (A, B, C, D, E) with functional dependencies A→C, D ; B→E ; A, E→F
Every functional dependency in relation M violates BCNF because none of the ~~fet~~ left hand sides are a super key of M.
So, we start by splitting M on the function dependency A→C, D to obtain:

M1 (A, C, D) where A→C, D
M2 (A, B, E, F) where B→E, and A, E→F and A, B→EF
(this is a desired functional dependency since B→E and A, E →F)

Now, M1 is BCNF because A is a super key and there are other functional dependencies in that relation violating BCNF. M2 is not BCNF because B→E & A, E→F violate BCNF. We can break M2 on B→E to obtain M3 and M4.

M1 (A, C, D) where A→C, D
M3 (B, E) where B→E
M4 (A, B, F) where only trivial functional dependency remain

Now, M_1, M_3 and M_4 are lossless BCNF decomposition of M
If the relation were $M_1(A, C, D)$, $M_2(B, E)$, $M_3(A, E, F)$
then it would be a lossy decomposition.

⇒ ## Hoolman's Algorithm:

It is used to check whether a decomposition of a relation is lossy or lossless. It is a test for lossless (non-additive) join property.
Here, a universal relation R, a decomposition $D = \{R_1, R_2, R_3, \ldots R_n\}$ of R, and of set F of functional dependencies are defined. where
$R = \{A_1, A_2, A_3, \ldots A_n\}$

Algorithm:

1) Create an initial matrix S with one row i for each relation in R_i in D, and one column j for each attribute A_j in R.

ex – Let $R = (A, B, C, D)$, $FD = (A \to B, B \to C, C \to D)$
and R is decomposed into $R_1(A, B)$, $R_2(B, C)$ and $R_3(C, D)$

		1 A	2 B	3 C	4 D
1	R_1				
2	R_2				
3	R_3				

2) Set $S(i,j) := b_{ij}$ for all matrix entries, if relation R_i includes attribute A_j then set $S(i,j) = a_j$

		1	2	3	4
		A	B	C	D
1	R_1	a_1	a_2	b_{13}	b_{14}
2	R_2	b_{21}	a_2	a_3	b_{24}
3	R_3	b_{31}	b_{32}	a_3	a_4

3) For each functional dependency $X \to Y$ in F, for all rows in S which have the same symbol in the columns corresponding to attribute in X make the symbols in each column that correspond to an attribute is Y be same is all these rows as follows:

i) If any of the rows have an "a" symbol for the column, set the other rows to that same "a" symbol in the column.

ii) If no "a" symbol exists for the attribute in any of the rows, choose one of the "b" symbols that appear in one of the rows for the attribute and set other rows to that same "b" symbol in the column.

4) Repeat the following loop until a complete loop execution results in no change to S.

5) If a row is made up entirely of "a" symbol, then the decomposition has the lossless join property otherwise the decomposition is lossy.

		1	1	3	4
		A	B	C	D
1	R_1	a_1	a_2	b_{13} a_3	b_{14}
2	R_2	b_{21}	a_2	a_3	b_{24}
3	R_3	b_{31}	b_{32}	a_3	a_4

$\Downarrow$

		1	2	3	4
		A	B	C	D
1	R_1	a_1	a_2	a_3	b_{14} a_4
2	R_2	b_{21}	a_2	a_3	b_{24} a_4
3	R_3	b_{31}	b_{32}	a_3	a_4

} Row 1 contains all a's
Therefore, the decomposed relations are lossless.

Conclusion: Decomposition, lossy and lossless decomposition and Ullman's decomposition algorithm has been studing in Database Management System (DBMS)