Fragmentation in Distributed DBMS

Fragmentation is a process of dividing the whole or full database into various subtables or sub relations so that data can be stored in different systems. The small pieces or sub relations or subtables are called *fragments*. These fragments are called logical data units and are stored at various sites. It must be made sure that the fragments are such that they can be used to reconstruct the original relation (i.e, there isn't any loss of data).

In the fragmentation process, let's say, If a table T is fragmented and is divided into a number of fragments say T1, T2, T3….TN. The fragments contain sufficient information to allow the restoration of the original table T. This restoration can be done by the use of UNION or JOIN operation on various fragments. This process is called ***data fragmentation***. All of these fragments are independent which means these fragments can not be derived from others. The users needn't be logically concerned about fragmentation which means they should not concerned that the data is fragmented and this is called *fragmentation Independence* or we can say *fragmentation transparency*.

Advantages :

- As the data is stored close to the usage site, the efficiency of the database system will increase
- Local query optimization methods are sufficient for some queries as the data is available locally
- In order to maintain the security and privacy of the database system, fragmentation is advantageous

Disadvantages :

- Access speeds may be very high if data from different fragments are needed
- If we are using recursive fragmentation, then it will be very expensive

We have three methods for data fragmenting of a table:

- **Horizontal fragmentation**
- **Vertical fragmentation**
- **Mixed or Hybrid fragmentation**

Let's discuss them one by one.

Horizontal fragmentation –

Horizontal fragmentation refers to the process of dividing a table horizontally by assigning each row (or a group of rows) of relation to one or more fragments. These fragments can then be assigned to different sites in the distributed system. Some of the rows or tuples of the table are placed in one system and the rest are placed in other systems. The rows that belong to the horizontal fragments are specified by a condition on one or more attributes of the relation. In relational algebra horizontal fragmentation on table T, can be represented as follows:

$\sigma_p(T)$

where, σ is relational algebra operator for selection

p is the condition satisfied by a horizontal fragment

Note that a union operation can be performed on the fragments to construct table T. Such a fragment containing all the rows of table T is called a *complete horizontal fragment.*

For example, consider an EMPLOYEE table (T) :

Eno	Ename	Design	Salary	Dep
101	A	abc	3000	1
102	B	abc	4000	1
103	C	abc	5500	2
104	D	abc	5000	2
105	E	abc	2000	2

This EMPLOYEE table can be divided into different fragments like:

EMP 1 = $\sigma_{Dep = 1}$ EMPLOYEE

EMP 2 = $\sigma_{Dep = 2}$ EMPLOYEE

These two fragments are: T1 fragment of Dep = 1

Eno	Ename	Design	Salary	Dep
101	A	abc	3000	1
102	B	abc	4000	1

Similarly, the T2 fragment on the basis of Dep = 2 will be :

Eno	Ename	Design	Salary	Dep
103	C	abc	5500	2
104	D	abc	5000	2
105	E	abc	2000	2

Now, here it is possible to get back T as **T = T1 ∪ T2 ∪ …. ∪ TN**

Vertical Fragmentation

Vertical fragmentation refers to the process of decomposing a table vertically by attributes or columns. In this fragmentation, some of the attributes are stored in one system and the rest are

stored in other systems. This is because each site may not need all columns of a table. In order to take care of restoration, each fragment must contain the primary key field(s) in a table. The fragmentation should be in such a manner that we can rebuild a table from the fragment by taking the natural JOIN operation and to make it possible we need to include a special attribute called ***Tuple-id*** to the schema. For this purpose, a user can use any super key. And by this, the tuples or rows can be linked together. The projection is as follows:

$\pi_{a1, a2, ..., an}(T)$

where, π is relational algebra operator

a1 ...., an are the attributes of T

T is the table (relation)

For example, for the EMPLOYEE table we have T1 as :

Eno	Ename	Design	Tuple_id
101	A	abc	1
102	B	abc	2
103	C	abc	3
104	D	abc	4
105	E	abc	5

For the second. sub table of relation after vertical fragmentation is given as follows :

Salary	Dep	Tuple_id
3000	1	1
4000	2	2
5500	3	3
5000	1	4
2000	4	5

This is T2 and to get back to the original T, we join these two fragments T1 and T2 as π_{EMPLOYEE} **(T1 ⋈ T2)**

Mixed Fragmentation

The combination of vertical fragmentation of a table followed by further horizontal fragmentation of some fragments is called **mixed or hybrid fragmentation**. For defining this type of

fragmentation we use the SELECT and the PROJECT operations of relational algebra. In some situations, the horizontal and the vertical fragmentation isn't enough to distribute data for some applications and in that conditions, we need a fragmentation called a mixed fragmentation.

Mixed fragmentation can be done in two different ways:

1. The first method is to first create a set or group of horizontal fragments and then create vertical fragments from one or more of the horizontal fragments.
2. The second method is to first create a set or group of vertical fragments and then create horizontal fragments from one or more of the vertical fragments.
   The original relation can be obtained by the combination of JOIN and UNION operations which is given as follows:

$\sigma_P(\pi_{a1,\ a2..,an}(T))$

$\pi_{a1,a2....,an}\ (\sigma_p(T))$

For example, for our **EMPLOYEE** table, below is the implementation of mixed fragmentation is $\pi_{\text{Ename, Design}}\ (\sigma_{\text{Eno} < 104}(\text{EMPLOYEE}))$

The result of this fragmentation is:

Ename	Design
A	abc
B	abc
C	abc