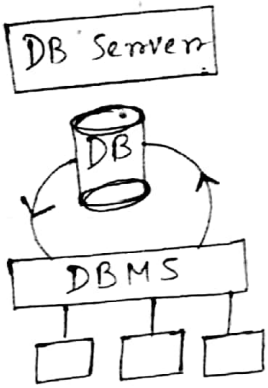
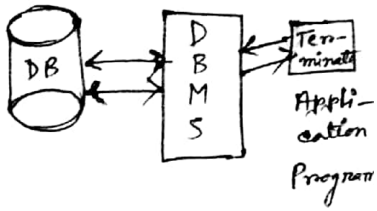
Database :- 1) Collection of Related Records Date → 2/8/18
2) " " Organized data

DBMS is the interface between the user and database.



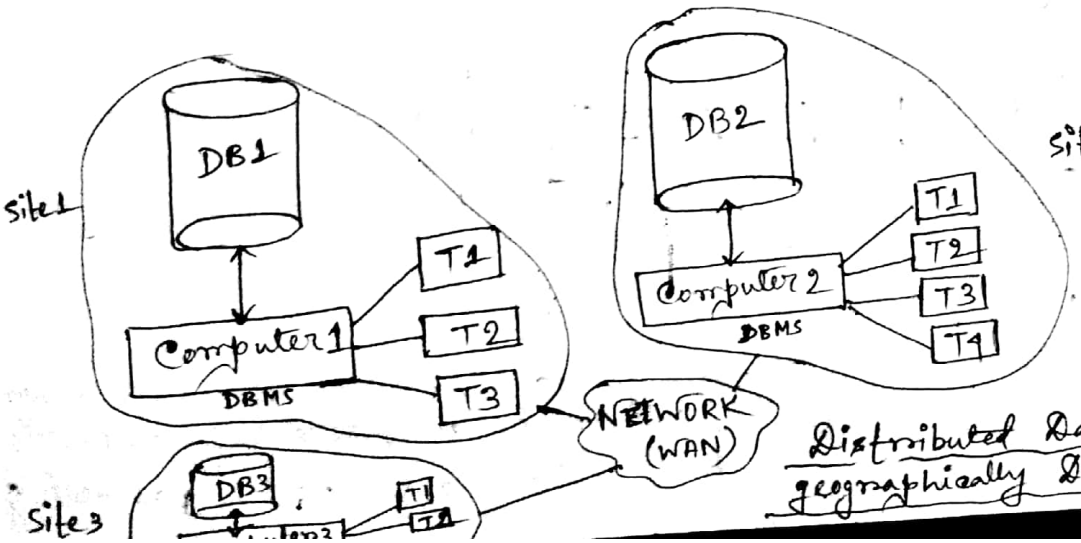


Definition :- A distributed database is a collection of data which belong logically to the same system but are spread over more than one side of a computer network. Two important aspects of a distributed database are →

i) No. of distribution → means that the data are not resident at the same side so that we can distinguish a distributed database from a single centralized database.

ii) Logical co-relation → The data have some properties which tie them together so that we can distinguish an distributed database from a set of local databases which are resident at different sides of a computer network.

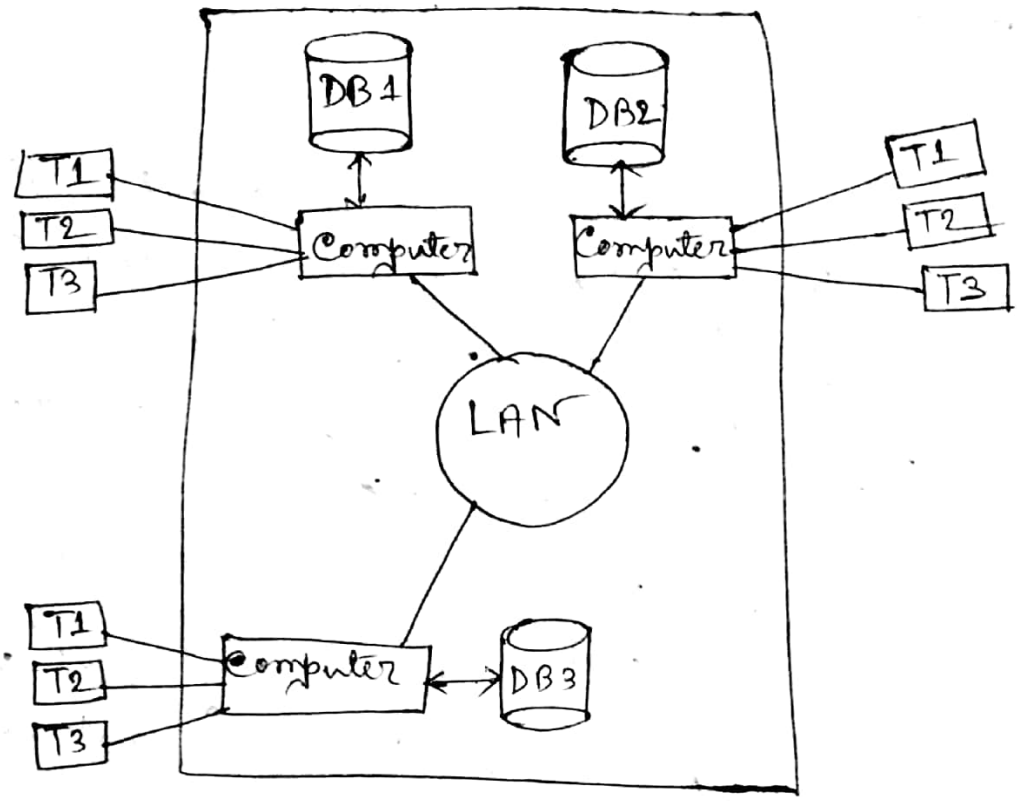
Local Application
Global "



Distributed Database in a geographically Dispersed Network

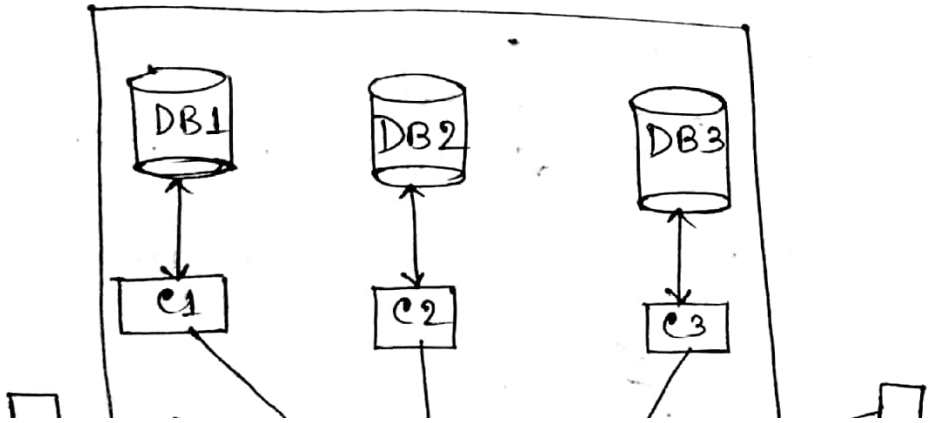
Local applications access the database of that particular site with the help of the DBMS of their own site where as Global applications access data from more than one site. These are also called Distributed applications.

Distributed Database in a local network



Multiprocessor Systems

(This is not a distributed database. There is no ~~two~~ global application



capability and can perform local applications.
Each site also participates in the execution of atleast one global application which requires accessing data at several sites using a communication subsystems.

Date → 7/8/18

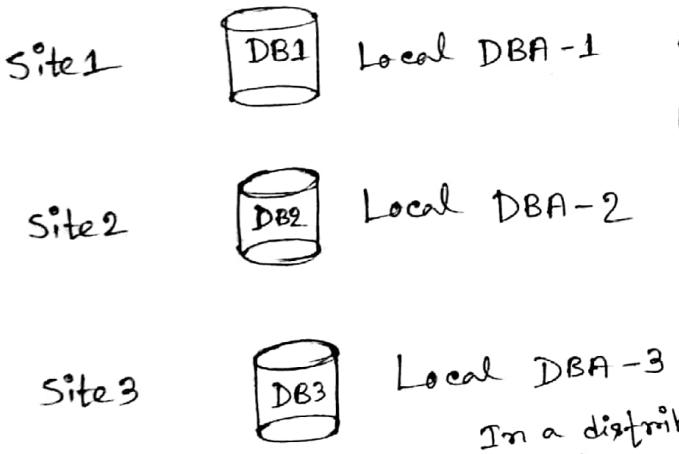
Features of Distributed Database :-

1) Centralised Control :-

Centralised → Central Control - DBA
Distributed → Site Autonomy - Local DBA

The fundamental function of a DBA, incase of a central database is to provide access to the data and garuntee the safety of data.

In distributed database, the idea of centralised control is much less emphasis. In general, in distributed database it is possible to identify a hierarchical control structure based on a global database administrator who has the central responsibility of the whole database whereas local DBAs have the the responsibility of the local respective databases.



2) Data Independence :-

In a distributed database, data independence is the same important as in traditional databases.

Distributed Transparency

However, a new aspects is added to the concept of data independence that is distribution transparency. By distributed transparency, we mean that programs can be written as if the database is not distributed.

Thus the correctness of a program by unaffected by movement of data from one side to another. However the speed of execution is affected.

3) Reduction of Redundency :- In traditional databases, redundency was reduced as far as possible for two reasons →

i) Inconsistency and

ii) Wastage of disk space

However, In distributed database, there are several reasons for considering data redundancy as the desirable features.

1) The locality of the application can be increased if the data is replicated at all the sites where the application need it.

2) The avalability of the system can be increased because the site failure does not stopped the execution of application at the other site if the data is replicated.

4) Integrity, Recovery and Concurrency Control :-

A transaction is an atomic unit of execution and hence maintaining transaction atomicity is an important aspects of database.

It ensures database integrity.

There are two important aspects of transaction atomicity, they are →

1) failures and

2) concurrency

In case of distributed database, the synchronization problem is harder than the centralized system because the recovery mechanism has to be implemented along with integrity and concurrency control.

5) Privacy and Security: [illegible] control can ensure that [illegible] authorized access to the data is performed. However, without specialized control procedures it is more exposed or vulnerable to privacy and security violations than based on separate files. In a distributed database with a very high degree of site autonomy the owners of local data feel more ~~sec~~ secure because they can enforce their own protection rather than depending on a single DBA.
For the security problems are natural to distributed system in general because communication networks are represent week point with respect to security.

* ~~control~~ Complex Physical Structure and efficient use/access:-

Complex accessing structures like secondary indexes, interfile ~~change~~ chains and so on are major aspects of traditional databases. The reason for provide a complex accessing structure is to obtain efficient access to the data. In distributed databases complex accessing structures are not the right tool for efficient access, therefore while efficient access is the main problem in distributed databases physical structures are not available technological issue.

* Reasons for Implementing Distributed Database :—

1) Organizational and Economic Reasons :— Many organizations are decentralized and a distributed database approach fits more naturally to the structure of the organization.

2) Interconnection of Existing Database :- Distributed Databases are the natural solution when several database are exists in an organization and the necessity of ~~the~~ global performing application arises.
In this case, the distributed database is created

new organization units [illegible] approach, supports a smooth incremental growth with a minimum degree of impact on the already existing units.

1) Reduce Communication Overload :- Compare it to a centralized database where the data access is at a central location, distributed database allows local applications and hence reducing the network overload.

5) Performance Consideration : The existence of several autonomous processors results and the increase of performance through high degree of parallelism.

6) Reliability and Availability :- Distributed databases approach can be used in order to obtain high reliability, however the autonomous processing capability of the different sites does not guarantee by itself a higher overall reliability of the system, but it ensures a graceful degradation which means failures in a distributed database can be more frequent than in a centralized one because of greater no. of components. However the failure at one site does not shut down the system completely, only the local application of that site and global application accessing that site will not be available.

Date → 14/8/1[illegible]

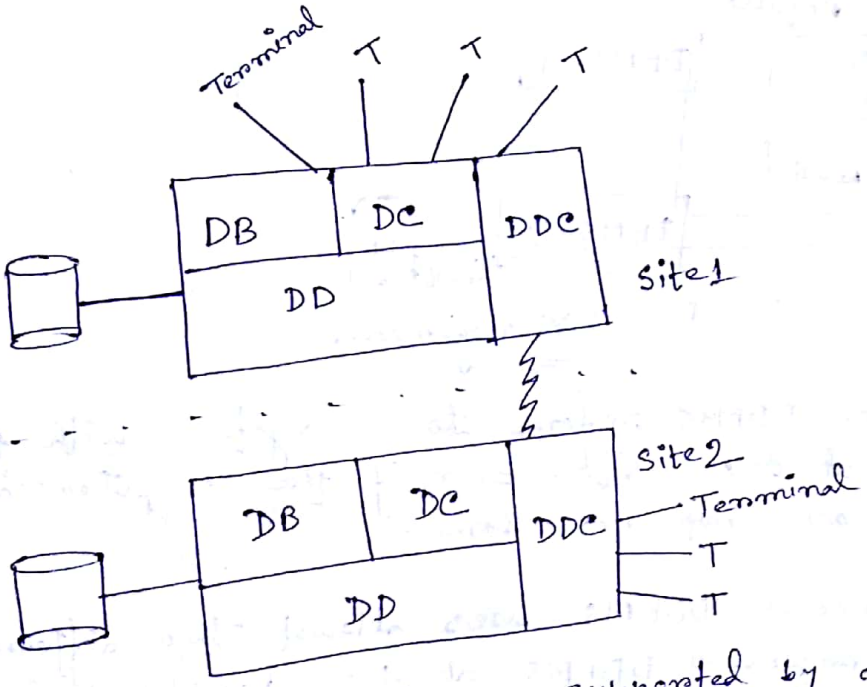
Distributed Database Management System (DDBMS) :-

A DDBMS supports the creation and maintainance of distributed databases. They contain additional components which extend the capabilities of centralized DBMS by supporting communication and cooperation

tion between several instances of DBMS which are installed at different sites of computer network.

The various software components of distributed database are →

i) Database Management component (DB).
ii) Data communication component (DC).
iii) Data Dictionary (DD) :- The data dictionary is extended to represent the information about the distribution of data over a network.
iv) Distributed Database component (DDC).



The services that are supported by a DDBMS are →

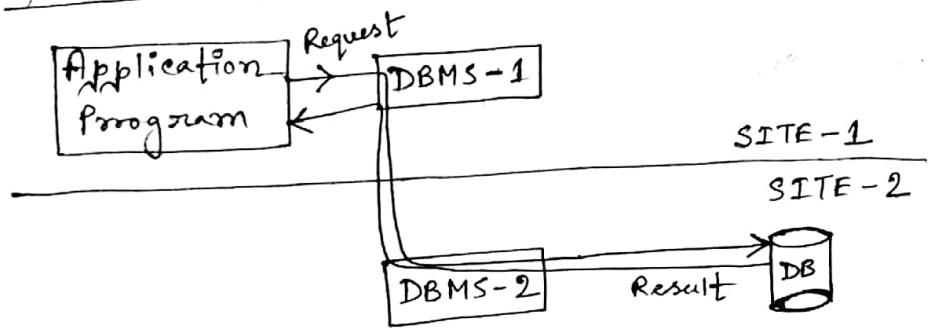
1) Remote Database Access by an application program:- This feature is a most important one and provided by all systems which have distributed database component.

2) Some degree of distributed transparency :- this feature is supported to a different aspects by different systems

because there is a strong contradiction between distribution transparency and performance.

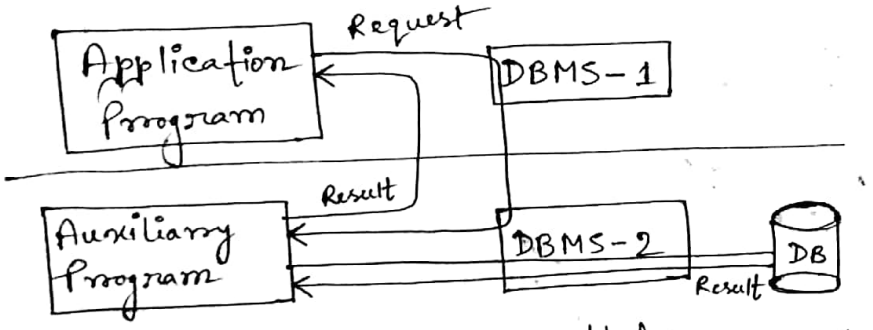
3) Support for database administration and control: this feature includes tools for monitoring the database, gathering information about the database, utilization and providing a global view of the data files existing at different sites.

4) Support for concurrency control and recovery of distributed transactions.



Remote Access

Homogeneous



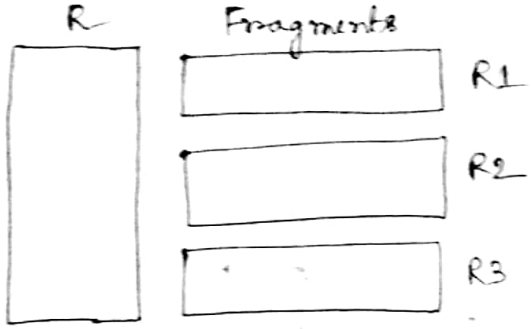
Heterogeneous

A homogeneous DDBMS refers to a system with the same DBMS at each site. Even if the computers AND/OR the OS are not the same.

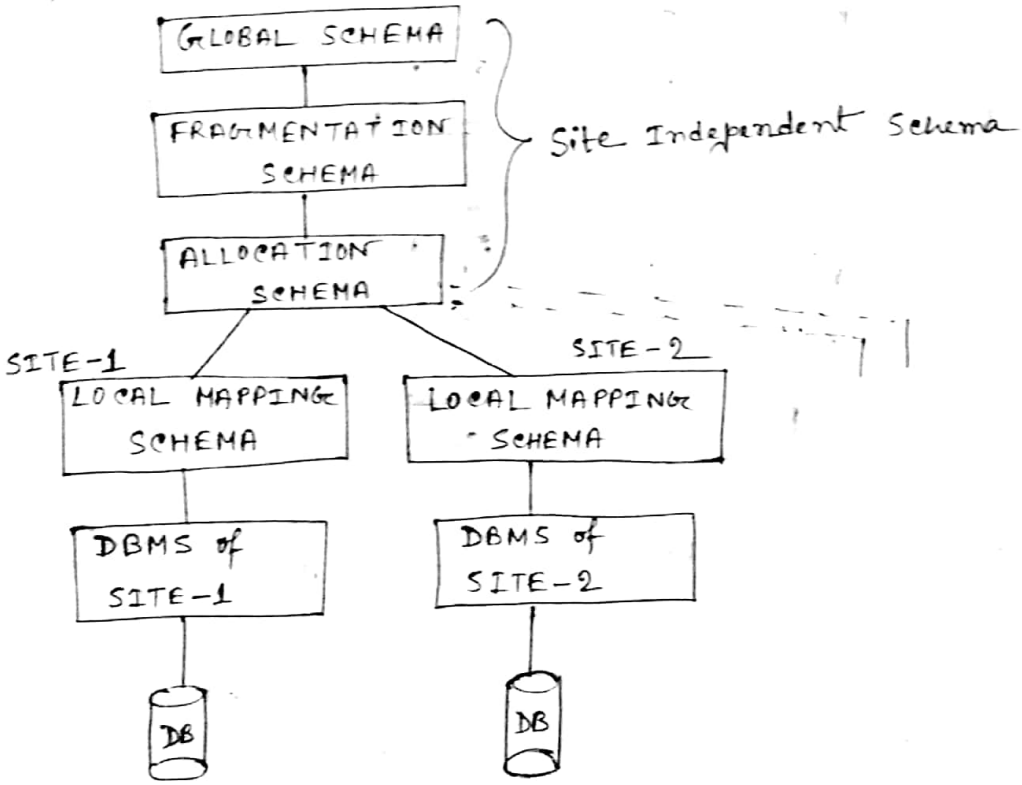
A heterogeneous DDBMS uses atleast two different DBMSs. Heterogeneous DDBMS at the problem of translating between the different data models of the different local DBMS to the complexity of the homogeneous DBMS. This requires building a global view of the database which is stacked by an auxiliary program present at each site.

Date → 16/8/18

Levels of Distribution Transparency



* Reference Architecture of the ~~DDBMS~~ Distributed Database :-



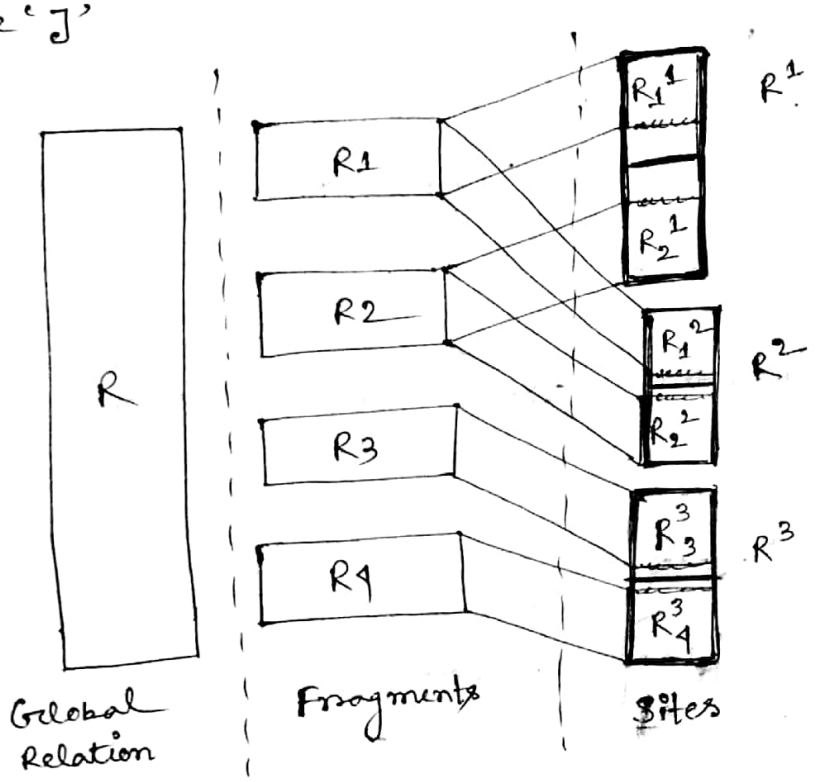
The global schema defines all the data which are contained in distributed database as if the database ~~are dis~~ were not distributed at all. The global schema consist of the definition of set of global relations

Each global relation can be split into several non-overlapping portions which are called fragments. The mapping between the global relations and the fragments is defined in the fragmentation schema. This relation is one to many because several fragments are mapped to one global relation.

Fragments are logical portions of global ~~schema~~ relations which are physically located at one or several sites of the network. The allocation schema

defines at which site(s) a fragment be located. The type of mapping defined in the allocation schema determines whether distributed database is redundent or nonredundent.

All the fragments which correspond to the same global relation 'R' and are located at the same site 'J' constitute the physical image of global relation 'R' at the site 'J'



Fragments and Physical Images for a Global Relation

At a lower level, it is necessary to map the physical images to the objects which are manupulated by the local DBAs. This mapping is called local mapping schema and depends on the type of local DBMS. Therefore, in a heterogeneous system we have different types of local mapping at different sites.

Date → 21/8/18

There are three more important features of this architecture part →

1) Seperation of data fragmentation and allocation.
2) Control of redundency.
3) Independence from local database.

1) Seperating the concept of data fragmentation from the concept of data allocation. The seperation allows us to

distinguish two different levels of distinguish and transparency namely fragmentation transparency and location transparency. Fragmentation transparency is the highest degree of transparency and consists of the fact that the user and application programmer works on global relations. Location transparency is the lower degree of transparency and requires the user and application programmer to work on the fragments instead of global relations. However, the user doesn't know, where the fragments are located. The seperation between the concept of fragmentation and allocation is very convenient in the distributed database design because the determination of the relevant portion of data in distinguished from the problem of optimal allocation where optimal allocation means that database is fragmented only upto the level that is necessary.

2) The reference architecture provides explicit control of redundancy at the fragment level. In the previous figure the two images are available R2 and R3 which are overlapped in each other. Although the fragments of physical images allows us to refer to this overlapping parts, the explicit control over redundency is useful in several aspects of distributed database management.

3) Independence from local database feature calls the local mapping transparency allows us to develop several applications of distributed DBMS without having to take into account the specific data models of local DBMS. Clearly in a homogeneous system, it is possible that the size independent schema are defined using the same data model as the local DBMS. This reduces complexity of the mapping.

Another type of transparency is strictly related to location transparency is replication transparency, it means that the user is unaware the replication of fragments.

Date → 30/8/18

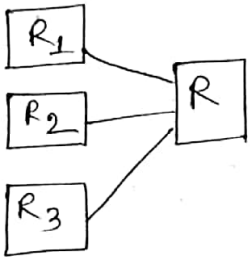
Fragmentation

Rules which must be followed when defining fragments:

1) Completeness condition :- All the data of global relation must be mapped into the

fragments that means it must not happen that the data item which belongs to a global relation does not belong to any fragment.

2) <u>Reconstruction Condition</u> :- It must always be posible to reconstruct each global relation from its fragments in fact only fragments are stored in the distributed database and global relation have to be built ~~through~~ with the reconstruction operation if ~~need~~ necessary.



3) <u>Disjoint Condition</u> :- It is convenient that the fragments be disjoint so that the replication of data ~~so that~~ can be controlled explicitly at the allocate level. However this condition is useful mainly with horizontal fragmentation while for vertical fragmentation, this condition needs to be violated sometimes.

* <u>Horizontal Fragmentation</u> :- It consists of partitioning ~~the~~ the tuples of a global relation into subsets. It can be defined by expressing each fragment as a selection operation of the global relation.

<u>Example</u> :-

SUPPLIER (SNUM, NAME, CITY)

$SUPPLIER1 = SL_{CITY="MUMBAI"}\ SUPPLIER$

$SUPPLIER2 = SL_{CITY="KOLKATA"}\ SUPPLIER$

qualification → q_1 : CITY = "MUMBAI"
q_2 : CITY = "KOLKATA"

$SUPPLIER = SUPPLIER_1\ UN\ SUPPLIER_2$ [UN stands for Union]

a global relation

always be posible
ruct each global
act only fragments
ted database
built ~~through~~
ation if ~~need~~ necessary.

t that the frag-
t so that the
controlled
However this
horizontal frag-
nagmentation,
ed sometimes.

sists of partitioning
the tuples
It can be defined
selection opera-

UPPLIER
SUPPLIER

IER$_2$ [UN stands for Union]

* Derived Fragmentation :- The horizontal fragmentation of a relation can not be based on the properties on its own attributes but it derived from the horizontal fragmentation of another relation.

Example :-

SUPPLY (SNUM, PNUM, DEPTNUM, QTY)

$SUPPLY_1 = SUPPLY \ SJ_{SNUM=SNUM} \ SUPPLIER_1$ [SJ stands for Semi Join]

$SUPPLY_2 = SUPPLY \ SJ_{SNUM=SNUM} \ SUPPLIER_2$

q_1 = SUPPLY.SNUM=SUPPLIER.SNUM AND SUPPLIER.CITY="MUMBAI"

q_2 = SUPPLY.SNUM=SUPPLIER.SNUM AND SUPPLIER.CITY="KOLKATA"

Referential Integrity

Date → 4/9/18

* Vertical Fragmentation

EMP (EMPNUM, NAME, SAL, TAX, MGRNUM, DEPTNUM)

$EMP_1 = PJ_{EMPNUM, NAME, MGRNUM, DEPTNUM} \ EMP$

$EMP_2 = PJ_{EMPNUM, SAL, TAX} \ EMP$

EMP_1 (EMPNUM, NAME, MGRNUM, DEPTNUM)

EMP_2 (EMPNUM, SAL, TAX)

The Vertical fragmentation of a global relation is the subdivision of its ~~the~~ attributes into groups. Fragments are obtained by projecting the global relation over each group. The fragmentation is correct if each attribute is mapped into atleast one attribute of the fragments. Moreover, it must be possible to reconstruct the original relation by joining the fragments together.

$EMP = EMP_1 \ JN_{EMPNUM=EMPNUM} \ EMP_2$

In vertical fragmentation, the main motivation of having disjoint fragments is not as important as in horizontal fragmentation.

EMP_1 (EMPNUM, NAME, MGRNUM, DEPTNUM)

EMP_2 (EMPNUM, NAME, MGRNUM, DEPTNO)

$EMP = EMP_1 \; JN_{EMPNUM=EMPNUM}$

$PJ_{EMPNUM, SAL, TAX} \; EMP_2$

SL → Select
JN → Join
PJ → Projection

* Mixed Fragmentation :-

EMP(EMPNUM, NAME, SAL, TAX, MGRNUM, DEPTNUM)

horizontal:

$EMP_1 = SL \; DEPTNAME <= 10 \; PJ_{EMPNUM, NAME, MGRNUM, DEPTNUM} \; EMP$

$EMP_2 = SL \; > 10 \; DEPTNAME <= 20 \; PJ_{EMPNUM, NAME, MGRNUM, DEPTNUM} \; EMP$

$EMP_3 = SL \; DEPTNAME > 20 \; PJ_{EMPNUM, NAME, MGRNUM, DEPTNUM} \; EMP$

Pure vertical:

$EMP_4 = PJ_{EMPNUM, SAL, TAX} \; EMP$

$EMP = UN(EMP_1, EMP_2, EMP_3) \; JN_{EMPNUM=EMPNUM} \; PJ_{EMPNUM, SAL, TAX} \; EMP$

95 records

6 column = Attributes

T_1 = 4 columns

T_2 = 3 columns

T_{11} = 45 records

T_{12} = 30 records

T_{13} = 20 records