UNIT-1

Distributed Database Management System

Learning Objective

- Distributed DBMS features and needs
- Reference Architecture, Levels of Distribution
- Transparency, Replication, Distributed database design – Fragmentation, allocation criteria,
- Storage mechanisms, Translation of Global Queries / Global Query Optimization, Query
- Execution and access plan

A Centralized DBMS on a Network

Centralized DBMSs
in which all of the data is maintained at a single site given as in figure.
Disadvantage of Centralized Database

**Disadvantages:**
- Single Point of failure
- Performance Bottleneck
- Contention- Competition for resources

It is a situation where two or more nodes attempt to transmit a message across the same wire at the same time. Contention (term) is used especially in Networks.

Distributed Database System

- In a distributed database system, data is physically stored across several sites, and each site is typically managed by a DBMS capable of running independent of the other sites.
- The location of the data items and the degree of autonomy of the individual sites have a significant impact on all aspects of the system, including query processing and optimization, concurrency control, and recovery.

In contrast to parallel database systems, the distribution of data is governed by factors such as local ownership and increased availability, in addition to performance related issues.
A distributed database (DDB) is a collection of multiple, logically interrelated databases distributed over a computer network.

- Distributed Database - A logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network.

- DDBMS - Software system that permits the management of the distributed database and makes the distribution transparent to users.

- Processing logic - Processing logic/Processing elements are distributed
  - Inventory
  - Personnel
  - Sales

- Functions
  Functions of a system could be delegated to various pieces of hardware/software
  - Printing
  - Email
Distributed Database System Cont...

- **Data**
  Data used by a no. of applications may be distributed to a no. processing sites.

- **Control**

---

Features of Centralized Vs distributed Databases

- **Centralized Vs Hierarchical Control**
  Site Autonomy

- **Data Independence Vs Distribution Transparency**
  Actual Organization of data is transparent to the application programmer

- **Reduced Redundancy Vs Redundancy**
  Increased locality of Applications Site Failure

---

Features of Centralized Vs distributed Databases

- **Indexing, Chaining Vs Distributed Access Plan**
  The Execution of programs which are local at single site and the transmission of files between sites.
Advantages of DDBMS

1) Improved Performance - data located near site.
2) Improved Availability - node failure will not make system inoperable.
3) Improved Reliability - replicated data allows data accessibility.
4) Organisational structure - many organizations cover several sites.
5) Shareability and local autonomy - users at different sites can share.

Business Advantages

Business Advantages
1) Economics
   Several smaller computer may be cheaper than a Mainframe system
2) Modular Growth-easier expansion
3) Integration
   Allows for combining of several legacy databases into one DBMS.

Disadvantages of DDBMS

- **Complexity** - more complex than centralized
- **Cost** - added network and maintenance costs
- **Security** - network must be made secure
- **Integrity control more difficult**
- **Lack of standards**
- **Lack of experience** - no tools or methodologies
- **Database design more complex**
Distributed database facilitate distribution of data across vast geographical spread. Distributed database is a collection of various database sites which are mapped as a single global database.

Some levels may be missing, depending on levels of transparency supported.

Can be homogeneous or heterogeneous.

**Global schema** defines all the data which are contained in the distributed database as if the database were not distributed at all, or in short global schema defines data as a whole.

GlobalSchema:Employee(EmpNo,Ename,Dept)

The next layer is the **Fragmentation Schema** specifying the way in which the global
relations are fragmented to serve the purpose of distribution.

Fragmentation Schema:
Employee1=SLDept='Mgr' Employee
Employee2=SLDept='Sales' Employee

3. Below the fragmentation schema exists the allocation schema determining the sites on which any particular fragment is to be deployed.

Allocation Schema: Employee1 at site1,2
Employee2 at site3,4

4. The subsequent layers exists on the local database sites.

- The first layer at the local database site is the local mapping schema which helps in identifying the global relation schema for any local database relation schema. It is the local mapping schema which facilitates the integration of local database sites into one single global database.

- Below this layer is the local schema of the local DBMS. It is very much similar to the three schema architecture of the centralized data bases.
**Classification of DDBMS**

- Homogeneous - All servers use same DBMS
- Heterogeneous – All servers use different DBMS

**Examples of typical applications:**

<table>
<thead>
<tr>
<th>Type of DBMS</th>
<th>LAN network</th>
<th>WAN network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homogeneous</td>
<td>Data management and financial applications</td>
<td>Travel management and financial applications</td>
</tr>
<tr>
<td>Heterogeneous</td>
<td>Inter-divisional information systems</td>
<td>Integrated banking and inter-banking systems</td>
</tr>
</tbody>
</table>

**Types of DDBMS**

Homogeneous – Same DBMS is used at each site.
- Autonomous – Each DBMS works independently, passing messages back and forth to share data updates.
- Non-Autonomous – A central, or master, DBMS coordinates database access and updates across the sites.

Heterogeneous – Potentially different DBMSs are used at each site.
- Systems – support some or all of the functionality of one logical database.
- Full DBMS functionality – supports all of the functionality of a distributed database.
A Homogenous Distributed Database

- A typical homogeneous distributed database environment is illustrated on the following page.
- This environment is typically defined by the following characteristics:
  - Data are distributed across all the nodes.
  - The same DBMS is used at each location.

A Homogenous Distributed Database Cont…

- All data are managed by the distributed DBMS. There is no exclusively local data.
- All users access the database through one global schema or database definition.
- The global schema is simply the union of all the local database schemas.
It is difficult in most organizations to force a homogeneous environment, yet heterogeneous environments are much more difficult to manage.

As the diagram illustrates, there are many variations of heterogeneous distributed database environments, however; a typical heterogeneous distributed database environment is defined by the following characteristics:

- Data are distributed across all the nodes.
- Different DBMSs may used at each location.
- Some users require only local access to databases, which can be accomplished using only the local DBMS and schema.
- A global schema exists, which allows local users to access "remote data".

In any distributed system transparency is the most central issue.

Base of distributed data base management system (DDBMS) emphasis's that a DDBMS should work like a non-Distributed DBMS.

The rule thus insists that the user should not be aware of the distribution of data.
1. **Fragmentation Transparency**: The user is not aware of the existence of fragments and work on global relations.

   Update emp set empno=10 where deptno=15; *(level1)*

2. **Location Transparency**: The user is aware of the fragments but it is not aware of the site of which they have been deployed.

   Insert into emp1 values(1,'Amit',20);
   Insert into emp2 values(2,'Ajeet',30);
   Delete emp1 where empno=10;
   Delete emp2 where empno=10; *(level2)*

3. **Local Mapping Transparency**: The user is aware of the fragments and the sites on which they have been deployed BUT he is insulated from the heterogeneity aspects.

   emp1: site1 and site 5(In case of update operation (replication))
   emp2: site2 and site 6
   emp3: site3 and site 7
   emp4: site4 and site 8
Levels of Distribution Transparency Cont…

Selecte name, esal, etax into $ename,$esal,$etax from emp1 at site1 where empno.=10;
- Insert into emp3(empno,ename,deptno) at site 3: (10,$ename,15);
- Insert into emp3(empno,ename,deptno) at site 7: (10,$ename,15);
- Insert into emp4(empno,esal,etax) at site 4: (10,$esal,$tax);

Data Replication

A relation or fragment of a relation is replicated if it is stored redundantly in two or more sites.

The two approach is given below:
- Full replication of a relation is the case where the relation is stored at all sites.
- Fully redundant databases are those in which every site contains a copy of the entire database.

Framework for Distributed Database Design

- Designing the conceptual schema- Which describes the integrated database. (i.e., all the data which are used by the database applications)
- Designing the physical database (i.e., mapping the conceptual schema to storage areas and determining appropriate access methods)
  - Designing the fragmentation
  - Designing the allocation of fragments
Objectives of data distribution design

• **Processing locality** - It ensures that those units of data which are most frequently accessed by any site are maintained locally as far as possible.

• **Availability** - A high degree of availability for read only applications is achieved by storing multiple copies of the same information; the system must be able to switch to an alternative copy when the one that should be accessed under normal condition is not available.

Objectives of data distribution design Cont…

• **Reliability** - Reliability is also achieved by storing multiple copies of the same information, since it is possible to recover from the crashes or from the physical destruction.

• **Distribution of workload** - Workload distribution is done in order to take advantage of the different powers or utilizations of computers at each site.

• **Storage costs** - Storage cost is directly depend on the how much information is locally required.

Fragmentation

• Fragmentation is the process of decomposition of global relations into fragments.

**Types of Fragmentation:**

• Horizontal Fragmentation
• Derived horizontal Fragmentation
• Vertical Fragmentation
• Hybrid/Mixed
Fragmentation Cont…

- **Horizontal** – Subset of rows
- **Vertical** – Subset of columns
  - Each fragment must contain primary key
  - Other columns can be replicated
- **Mixed (hybrid)** – both horizontal and vertical
- **Derived** – Derived from the horizontal fragmentation of another relation.

Example:

- Natural join first to get additional information required then fragment Must be able to reconstruct original table Can query and update through fragment

Horizontal Fragmentation

- **Horizontal fragmentation** is based on the selection operation. Some condition is chosen and against this condition the tuples are evaluated only those tuples which satisfied the condition become the part of that corresponding fragment.
- Example: If there is an organization it may fragment its global employees relation horizontally by keeping the records of the employee belonging to one particular country in a separate horizontal fragment.
Horizontal Fragmentation Cont…

Condition can be
C1=country_name="INDIA"
C2=country_name="United States"
.
.
.CN=country_name="Srilanka"

Example:
Let's a global relation (table) Supplier.
Supplier (SNum,Name,City)
Then the horizontal fragmentation can be defined as following:
Supplier1=SL city="sf" Supplier
Supplier2=SL city="la" Supplier

• Completeness
(The above fragmentation satisfies the completeness condition if "sf" and "la" are the only possible values of the City attribute, otherwise we would not know to which fragment the tuples with other City Values belong.)
Horizontal Fragmentation

- **Reconstruction** (It is always possible to reconstruct the Supplier global relation by using Union operation)
  
  Supplier=Supplier1 UN Supplier2

- **Disjointness** (Call the predicate which is used in the selection operation which define a fragment’s qualification and qualification be mutually exclusive)
  
  Q1=City="sf"
  Q2=City="la"

Derived Horizontal Fragmentation

- A Derived Horizontal fragmentation is based on conditions which are built on the output of some other Query. The Horizontal fragment thus define is a derived horizontal fragment.

Vertical Fragmentation

- It is based on Projection Operation
  
  The Predicate of the projection operation is a list of Attribute which are intended to constitute that corresponding vertical fragment.

- The Various predicate to carry out a vertical fragmentation are selected so as to meet the objectives of **disjointness**, **completeness** and **reconstruction**.
### Vertical Fragmentation Cont...

- Vertical Fragmentation can never be absolutely disjoint at least one column needs to be common, so as maintains referential integrity.

### Vertical Fragmentation Cont...

- Consists of a subset of attributes (column) of a relation.
- Defined using *Projection* operation of relational algebra:
  - \( \Pi_{a_1, \ldots, a_n}(R) \)
- For example:
  - \( S_1 = \Pi_{\text{staffNo, position, sex, DOB, salary}}(\text{Staff}) \)
  - \( S_2 = \Pi_{\text{staffNo, fName, lName, branchNo}}(\text{Staff}) \)
- Determined by establishing *affinity* of one attribute to another.

### Hybrid Fragmentation

- Applying vertical fragmentation to horizontal fragmentation.
### Fragment Allocation

- In determining the allocation of fragments, it is important to distinguish whether we design a final non-redundant or redundant allocation.

### Fragment Allocation

In case of non-redundant final allocation is easier. The simplest method is a "best-fit" approach; a measure is associated with each possible allocation, and the site with the best measure is selected.

### Fragment Allocation Cont...

**Replication introduces further complexity in the design, because:**

1. Degree of Replication is one problem
2. Maintaining Consistency is another issue.

For determining the redundant allocation of fragments, either of the following two methods can be used:
**Fragment Allocation Cont...**

- Determine the set of all sites where the “benefit of allocating one copy of fragment is higher than the cost”, and allocate a copy of the fragment to each element of this set; this method select “all beneficial sites”.

- Determine first the solution of the non replicated problem, and then progressively introduce replicated copies starting from the most beneficial; the process is terminated when no “additional replication” is beneficial.

**Assume relational data model**

- Replication
  - System maintains multiple copies of data, stored in different sites, for faster retrieval and fault tolerance.

- Fragmentation
  - Relation is partitioned into several fragments stored in distinct sites

**Fragment Allocation Cont...**

Replication and fragmentation can be combined

- Relation is partitioned into several fragments: system maintains several identical replicas of each such fragment.
In distributed Database management system a single global relation is sometimes fragmented and these fragments are deployed on various distinct sites, more over to ensure processing locality some times a relation or a fragment gets replicated even.

A query on the other hand is issued by a user or an application which is not aware of the existence of fragments, replicas and their respected allocations. This global query for its successful execution must get decomposed into fragment queries.

A global query can be

\[
\text{Select } * \text{ from student;}
\]

This Query must be decomposed into certain Queries which take as their operands the fragments into which student relation has been fragmented. These Queries are termed as fragment queries.

Select * from student1;
Select * from student2;
Select * from student3;

Select * from studentN;
The above fragment Queries are then executed at the respective sites and the result of these Queries are combined using a union operation and the result is to initiator site.

For the execution of these fragment Queries it is imperative that the information about the fragmentation, replication, and allocation must be obtained. This information can be derived from the fragmentation and the allocation schema and is kept in the system catalog.

To Decompose a global Queries into fragment Queries this information is procured from the catalog.

Catalog Management

Catalog Management OR Distribute Data Dictionary Management

System catalog Constitutes the data dictionary. It is the meta data i.e. it holds data about data.
Approaches for Catalog Management

- Centralized Approach
- Distributed Approach.

Centralized Approach for Catalog Management.

- Under this approach the system catalog is maintained at one of the participating sites in the distributed database.
- This site acts as the central coordinator of the distributed database management system.
- The basic advantage of this approach is that it is simple and consistency is not a concern. BUT the approach suffers from a major drawback:
  1) A single point of failure
  2) A performance bottleneck.

Centralized Approach Cont...

In case the coordinator site fails no site could progress as the catalog is maintained at only one place and added to that Query processing for all the sides can only be as efficient as the coordinator site is.
• Full replication Approach
• Partial replication Approach

Full Replication Approach

Under this approach the complete catalog is maintained on all the sites this allows processing locality to all the sites in a manner that the system catalog being locally available each site has a greater degree of Autonomy.

Draw Backs

BUT this approach has its own set of Drawbacks. One of the draw back is the storage overhead owing to greater redundancy and the other draw back is the consistency problem that is how to keep the replicated copies of the system catalog on various sites synchronized.
Partial Replication Approach

- Under this approach each site maintains a local catalog where information about the database objects for which the corresponding site is the birth site is store.

- Additionally it also holds the information of the replicas and each site maintains a set of links to database objects on the other site.

Partial Replication Approach Cont...

When ever any site submits a Query if it can be handled using the local catalog its OK else the links are evaluated. If the information is not available in the set of links then hunt for the database object is made and the set of links is accordingly updated.

Query Optimization

Global Query Optimization, Query Execution and access plan
Find all courses that "Mary" takes

```
SELECT C.name
FROM Students S, Takes T, Courses C
WHERE S.name = "Mary" and
      S.ssn = T.ssn and T.cid = C.cid
```

What happens behind the scene?
- Query processor figures out how to answer the query efficiently.

Declarative SQL query — Imperative query execution plan:

- Preprocess the relation or table.
- Perform selection as early as possible.
- Compute common expression only once.
- Translate an expression involving a Cartesian product followed by a subsequent selection into natural join.

The optimizer chooses the best execution plan for a query.
• Operator/Query Tree provides a more practical representation of queries, in which expression manipulation is easier.

• The leaves of the tree represent the relations and that each node represents an operation.

• Example: select snum from supply, dept where supply. deptnum=dept.deptnum and area='North';

Operator Tree Representation
Case1-Global Relation

Case2-Fragments
Query Optimization Cont…

Case 3: Fragments with Optimized result.

JN deptnum=deptnum

UN UN

Supply1 Supply2 Supply N

Dept1 Dept2 Dept N

Execution and Access Plan

- In order to execute and access query a plan is prepared by the programmer.
- This plan determines how to navigate in the complete database as well as how the database must be accessed.
- In order to implement these plan, this requires to implement optimization both at global as well as locally.

Global Optimization

- **Global Optimization** consists of determining which data must be accessed at which sites and which data files must consequently be transmitted between sites.
- The main optimization parameter for global optimization is communication cost. While Local Optimization consists of deciding how to perform the local database accesses at each site.
Example of Access plan

At site 1
Send site 2 and 3 the supplier number SN

2) At sites 2 and 3
Execute in parallel, upon receipt of the supplier number,
the following program:
Select part_no where supp_no=SN;
Send result to site 1

3) At site 1
Merge results from sites 2 and 3;
Output the result.

Short Questions

• Explain the use of distributed DBMS over Centralized DBMS?
• Discuss the transparency in terms of transaction.
• Describe various fragmentation techniques with examples?
• Explain the distribution of a Database on various sites.
• What is distributed DBMS and write its features?

Long Questions

• What are Global Optimization, Execution and Access Plan, give an example for access plan?
• Differentiate between homogeneous and heterogeneous DDBMS?
• Advantage and disadvantage of DDBMS, Explain?
• Describe Distributed approach for catalog management?
• What is fragmentation explain different type of fragmentation?
References