Distributed Databases

- Heterogeneous and Homogeneous Databases
- Distributed Data Storage
- Distributed Transactions
- Commit Protocols
- Concurrency Control in Distributed Databases
- Availability
- Distributed Query Processing
- Heterogeneous Distributed Databases
- Directory Systems

Distributed Database System

- A distributed database system consists of loosely coupled sites that share no physical component
- Database systems that run on each site are independent of each other
- Transactions may access data at one or more sites

Homogeneous Distributed Databases

- In a homogeneous distributed database
  - All sites have identical software
  - Are aware of each other and agree to cooperate in processing user requests.
  - Each site surrenders part of its autonomy in terms of right to change schemas or software
  - Appears to user as a single system
- In a heterogeneous distributed database
  - Different sites may use different schemas and software
    - Difference in schema is a major problem for query processing
    - Difference in software is a major problem for transaction processing
  - Sites may not be aware of each other and may provide only limited facilities for cooperation in transaction processing

Distributed Data Storage

- 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
- Replication and fragmentation can be combined
  - Relation is partitioned into several fragments: system maintains several identical replicas of each such fragment.

Data Replication

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

Data Replication (Cont.)

- Advantages of Replication
  - **Availability**: failure of site containing relation *r* does not result in unavailability of *r* is replicas exist.
  - **Parallelism**: queries on *r* may be processed by several nodes in parallel.
  - **Reduced data transfer**: relation *r* is available locally at each site containing a replica of *r*.
- Disadvantages of Replication
  - Increased cost of updates: each replica of relation *r* must be updated.
  - Increased complexity of concurrency control: concurrent updates to distinct replicas may lead to inconsistent data unless special concurrency control mechanisms are implemented.
    - One solution: choose one copy as **primary copy** and apply concurrency control operations on primary copy

Data Fragmentation

- Division of relation r into fragments $r_1, r_2, \ldots, r_n$ which contain sufficient information to reconstruct relation r.
- **Horizontal fragmentation**: each tuple of *r* is assigned to one or more fragments
- **Vertical fragmentation**: the schema for relation *r* is split into several smaller schemas
  - All schemas must contain a common candidate key (or superkey) to ensure lossless join property.
  - A special attribute, the tuple-id attribute may be added to each schema to serve as a candidate key.

Horizontal Fragmentation of *account* Relation

branch_name	*account_number*	*balance*
Hillside	A-305	500
Hillside	A-226	336
Hillside	A-155	62

$$account_1 = \sigma_{branch\_name=\text{“Hillside”}}(account)$$

branch_name	*account_number*	*balance*
Valleyview	A-177	205
Valleyview	A-402	10000
Valleyview	A-408	1123
Valleyview	A-639	750

$$account_2 = \sigma_{branch\_name=\text{“Valleyview”}}(account)$$

Vertical Fragmentation of *employee_info* Relation

branch_name	*customer_name*	
Hillside	Lowman	1
Hillside	Camp	2
Valleyview	Camp	3
Valleyview	Kahn	4
Hillside	Kahn	5
Valleyview	Kahn	6
Valleyview	Green	7

$deposit_1 = \Pi_{branch\_name,\ customer\_name,\ tuple\_id}(employee\_info)$

account_number	*balance*	*tuple_id*
A-305	500	1
A-226	336	2
A-177	205	3
A-402	10000	4
A-155	62	5
A-408	1123	6
A-639	750	7

$deposit_2 = \Pi_{account\_number,\ balance,\ tuple\_id}(employee\_info)$

Advantages of Fragmentation

- Horizontal:
  - allows parallel processing on fragments of a relation
  - allows a relation to be split so that tuples are located where they are most frequently accessed
- Vertical:
  - allows tuples to be split so that each part of the tuple is stored where it is most frequently accessed
  - tuple-id attribute allows efficient joining of vertical fragments
  - allows parallel processing on a relation
- Vertical and horizontal fragmentation can be mixed.
  - Fragments may be successively fragmented to an arbitrary depth.

Data Transparency

- **Data transparency**: Degree to which system user may remain unaware of the details of how and where the data items are stored in a distributed system
- Consider transparency issues in relation to:
  - Fragmentation transparency
  - Replication transparency
  - Location transparency

Naming of Data Items - Criteria

1. Every data item must have a system-wide unique name.
2. It should be possible to find the location of data items efficiently.
3. It should be possible to change the location of data items transparently.
4. Each site should be able to create new data items autonomously.

Centralized Scheme - Name Server

- Structure:
  - name server assigns all names
  - each site maintains a record of local data items
  - sites ask name server to locate non-local data items
- Advantages:
  - satisfies naming criteria 1-3
- Disadvantages:
  - does not satisfy naming criterion 4
  - name server is a potential performance bottleneck
  - name server is a single point of failure

Use of Aliases

- Alternative to centralized scheme: each site prefixes its own site identifier to any name that it generates i.e., *site* 17.*account.*
  - Fulfills having a unique identifier, and avoids problems associated with central control.
  - However, fails to achieve network transparency.
- Solution: Create a set of **aliases** for data items; Store the mapping of aliases to the real names at each site.
- The user can be unaware of the physical location of a data item, and is unaffected if the data item is moved from one site to another.

Extra Reference : 1

Distributed Database System

A distributed database is basically a database that is not limited to one system, it is spread over different sites, i.e, on multiple computers or over a network of computers. A distributed database system is located on various sites that don't share physical components. This may be required when a particular database needs to be accessed by various users globally. It needs to be managed such that for the users it looks like one single database.
control system.

Extra Reference : 1

Types:

1. **Homogeneous Database:**

   In a homogeneous database, all different sites store database identically. The operating system, database management system, and the data structures used – all are the same at all sites. Hence, they're easy to manage.

2. **Heterogeneous Database:**

   In a heterogeneous distributed database, different sites can use different schema and software that can lead to problems in query processing and transactions. Also, a particular site might be completely unaware of the other sites. Different computers may use a different operating system, different database application. They may even use different data models for the database. Hence, translations are required for different sites to communicate.

Extra Reference : 1

Distributed Data Storage :
There are 2 ways in which data can be stored on different sites. These are:
1. Replication –
In this approach, the entire relationship is stored redundantly at 2 or more sites. If the entire database is available at all sites, it is a fully redundant database. Hence, in replication, systems maintain copies of data.
This is advantageous as it increases the availability of data at different sites. Also, now query requests can be processed in parallel.
However, it has certain disadvantages as well. Data needs to be constantly updated. Any change made at one site needs to be recorded at every site that relation is stored or else it may lead to inconsistency. This is a lot of overhead. Also, concurrency control becomes way more complex as concurrent access now needs to be checked over a number of sites.

Extra Reference : 1

Distributed Data Storage :
There are 2 ways in which data can be stored on different sites. These are:
2. Fragmentation –
In this approach, the relations are fragmented (i.e., they're divided into smaller parts) and each of the fragments is stored in different sites where they're required. 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). Fragmentation is advantageous as it doesn't create copies of data, consistency is not a problem.

Extra Reference : 1

Horizontal fragmentation – Splitting by rows –
The relation is fragmented into groups of tuples so that each tuple is assigned to at least one fragment.
Vertical fragmentation – Splitting by columns –
The schema of the relation is divided into smaller schemas. Each fragment must contain a common candidate key so as to ensure a lossless join.
In certain cases, an approach that is hybrid of fragmentation and replication is used.
Applications of Distributed Database:
It is used in Corporate Management Information System.
It is used in multimedia applications.
Used in Military's control system, Hotel chains etc. It is also used in manufacturing control system.

Extra Reference : 1

Applications of Distributed Database:
It is used in Corporate Management Information System.
It is used in multimedia applications.
Used in Military's control system, Hotel chains etc. It is also used in manufacturing control system.

Extra Reference : 2

Applications of Distributed Database:
It is used in Corporate Management Information System.
It is used in multimedia applications.
Used in Military’s control system, Hotel chains etc. It is also used in manufacturing control system.

Extra Reference : 2

- A **distributed database** is a collection of multiple interconnected databases, which are spread physically across various locations that communicate via a computer network.
- **Features**
  - Databases in the collection are logically interrelated with each other. Often they represent a single logical database.
  - Data is physically stored across multiple sites. Data in each site can be managed by a DBMS independent of the other sites.
  - The processors in the sites are connected via a network. They do not have any multiprocessor configuration.
  - A distributed database is not a loosely connected file system.
  - A distributed database incorporates transaction processing, but it is not synonymous with a transaction processing system.

Extra Reference : 2

Factors Encouraging DDBMS

Distributed Nature of Organizational Units − Most organizations in the current times are subdivided into multiple units that are physically distributed over the globe. Each unit requires its own set of local data. Thus, the overall database of the organization becomes distributed.

Need for Sharing of Data − The multiple organizational units often need to communicate with each other and share their data and resources. This demands common databases or replicated databases that should be used in a synchronized manner.

Support for Both OLTP and OLAP − Online Transaction Processing (OLTP) and Online Analytical Processing (OLAP) work upon diversified systems which may have common data. Distributed database systems aid both these processing by providing synchronized data.

Extra Reference : 2

Factors Encouraging DDBMS

Database Recovery − One of the common techniques used in DDBMS is replication of data across different sites. Replication of data automatically helps in data recovery if database in any site is damaged. Users can access data from other sites while the damaged site is being reconstructed. Thus, database failure may become almost inconspicuous to users.

Support for Multiple Application Software − Most organizations use a variety of application software each with its specific database support. DDBMS provides a uniform functionality for using the same data among different platforms.

Extra Reference : 2

Advantages of Distributed Databases

Modular Development − If the system needs to be expanded to new locations or new units, in centralized database systems, the action requires substantial efforts and disruption in the existing functioning. However, in distributed databases, the work simply requires adding new computers and local data to the new site and finally connecting them to the distributed system, with no interruption in current functions.
More Reliable − In case of database failures, the total system of centralized databases comes to a halt. However, in distributed systems, when a component fails, the functioning of the system continues may be at a reduced performance. Hence DDBMS is more reliable.
Better Response − If data is distributed in an efficient manner, then user requests can be met from local data itself, thus providing faster response. On the other hand, in centralized systems, all queries have to pass through the central computer for processing, which increases the response time.

Extra Reference : 2

Advantages of Distributed Databases

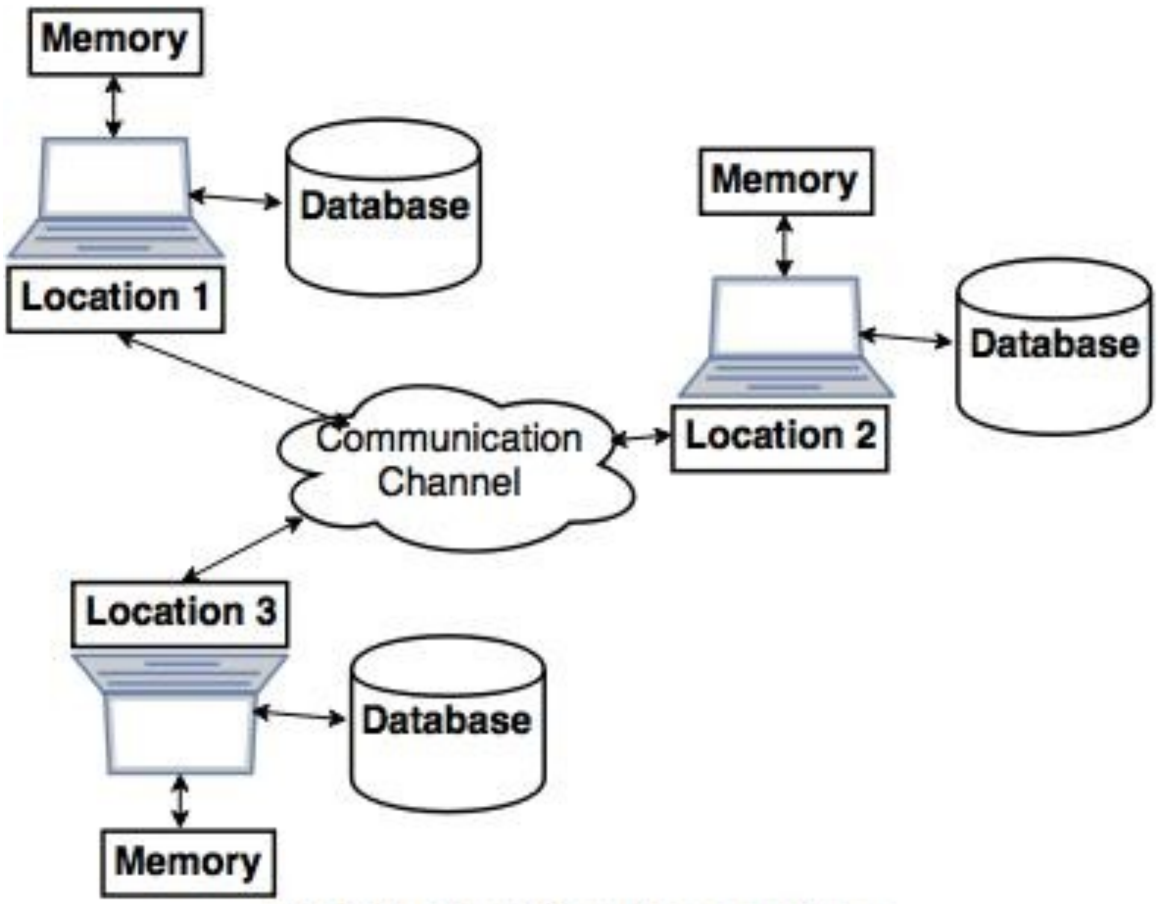
Lower Communication Cost − In distributed database systems, if data is located locally where it is mostly used, then the communication costs for data manipulation can be minimized. This is not feasible in centralized systems.

Extra Reference : 2

Adversities of Distributed Databases

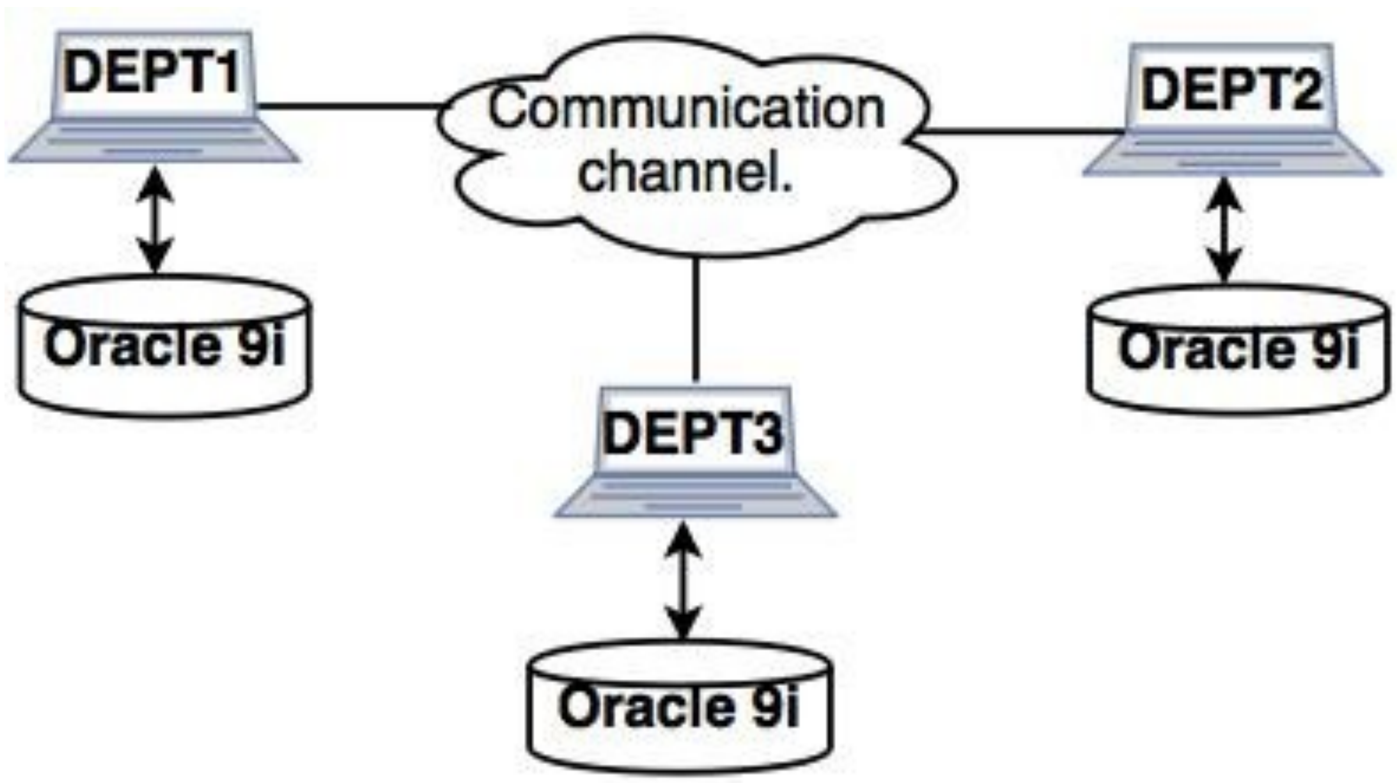
Need for complex and expensive software − DDBMS demands complex and often expensive software to provide data transparency and co-ordination across the several sites.
Processing overhead − Even simple operations may require a large number of communications and additional calculations to provide uniformity in data across the sites.
Data integrity − The need for updating data in multiple sites pose problems of data integrity.
Overheads for improper data distribution − Responsiveness of queries is largely dependent upon proper data distribution. Improper data distribution often leads to very slow response to user requests.

Extra reference : 3



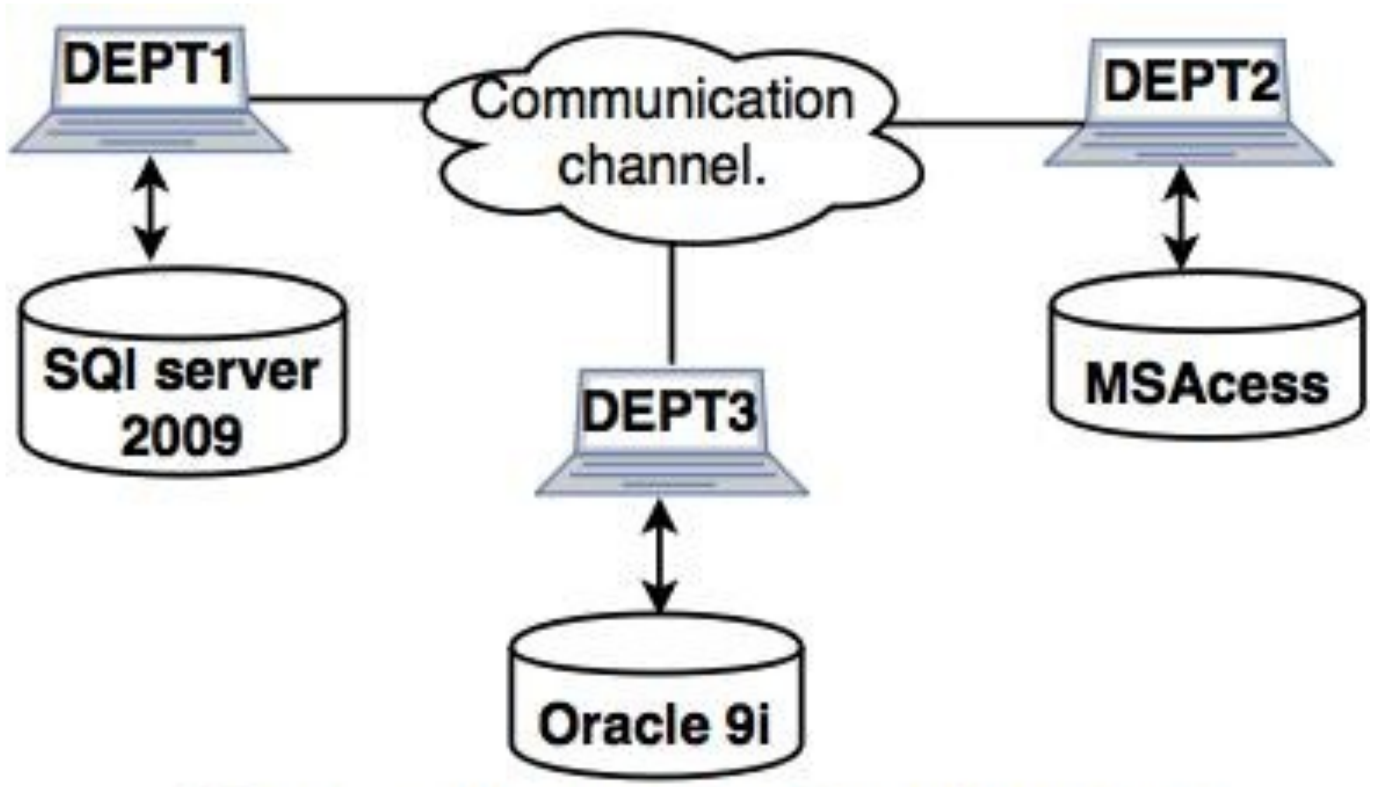
Distributted Dtatabase system

Extra reference : 3



Homogeneous distributed system

Extra reference : 3



Heterogeneous distributed system