UNIT I DISTRIBUTED DATABASES 9

Distributed Systems – Introduction – Architecture – Distributed Database Concepts – Distributed Data Storage – Distributed Transactions – Commit Protocols – Concurrency Control – Distributed Query Processing

DISTRIBUTED DATA STORAGE

Distributed database storage is managed in two ways: In database replication, the systems store copies of data on different sites. If an entire database is available on multiple sites, it is a fully redundant database.

Distributed databases are used for horizontal scaling, and they are designed to meet the workload requirements without having to make changes in the database application or vertically scale a single machine.

Distributed databases resolve various issues, such as availability, fault tolerance, throughput, latency, scalability, and many other problems that can arise from using a single machine and a single database.

Distributed Database Definition

A distributed database represents multiple interconnected databases spread out across several sites connected by a network. Since the databases are all connected, they appear as a single database to the users.

Distributed databases utilize multiple nodes. They scale horizontally and develop a distributed system. More nodes in the system provide more computing power, offer greater availability, and resolve the single point of failure issue.

Different parts of the distributed database are stored in **several physical locations**, and the processing requirements are distributed among processors on multiple database nodes.

A centralized distributed database management system (**DDBMS**) manages the distributed data as if it were stored in one physical location. DDBMS synchronizes all data operations among databases and ensures that the updates in one database automatically reflect on databases in other sites.

Distributed Database Features

Some general features of distributed databases are:

- **Location independency -** Data is physically stored at multiple sites and managed by an independent DDBMS.
- **Distributed query processing -** Distributed databases answer queries in a distributed environment that manages data at multiple sites. High-level queries are transformed into a query execution plan for simpler management.

- **Distributed transaction management -** Provides a consistent distributed database through commit protocols, distributed concurrency control techniques, and distributed recovery methods in case of many transactions and failures.
- **Seamless integration -** Databases in a collection usually represent a single logical database, and they are interconnected.
- **Network linking -** All databases in a collection are linked by a network and communicate with each other.
- **Transaction processing -** Distributed databases incorporate transaction processing, which is a program including a collection of one or more database operations. Transaction processing is an atomic process that is either entirely executed or not at all.

Distributed Database Types :

There are two types of distributed databases:

- **Homogenous**
- **Heterogenous**

Homogeneous :

A homogenous distributed database is a network of identical databases stored on multiple sites. The sites have the same operating system, DDBMS, and data structure, making them easily manageable.

Homogeneous databases allow users to access data from each of the databases seamlessly.

Heterogeneous :

A heterogeneous distributed database uses different schemas, operating systems, DDBMS, and different data models.

In the case of a heterogeneous distributed database, a particular site can be completely unaware of other sites causing limited cooperation in processing user requests. The limitation is why translations are required to establish communication between sites.

Distributed database storage is managed in two ways:

- **Replication**
- **Fragmentation**

Replication

In database replication, the systems store copies of data on different sites. If an entire database is available on multiple sites, it is a fully redundant database.

The advantage of database replication is that it increases data availability on different sites and allows for parallel query requests to be processed. However, database replication means that data requires constant updates and synchronization with other sites to maintain an exact database copy. Any changes made on one site must be recorded on other sites, or else inconsistencies occur. Constant updates cause a lot of server overhead and complicate concurrency control, as a lot of concurrent queries must be checked in all available sites.

Fragmentation

When it comes to fragmentation of distributed database storage, the relations are fragmented, which means they are **split into smaller parts**. Each of the fragments is stored on a different site, where it is required. The prerequisite for fragmentation is to make sure that the fragments can later be reconstructed into the original relation without losing data. The advantage of fragmentation is that there are **no data copies**, which prevents data inconsistency.

There are two types of fragmentation:

- Horizontal fragmentation - The relation schema is fragmented into groups of rows, and each group (tuple) is assigned to one fragment.
- Vertical fragmentation - The relation schema is fragmented into smaller schemas, and each fragment contains a common candidate key to guarantee a lossless join.