Parallel Database

By,
Harshil T. Kanakia

Outline

- Overview of Parallel Database Architecture
- Types of Parallelism
- Types of Partitioning

Parallel Database

Today everybody interested in storing the information they have got. Even small organizations collect data and maintain mega databases. Though the databases eat space, they really helpful in many ways. For example, they are helpful in taking decisions through a decision support system. To handle such a voluminous data through conventional centralized system is bit complex. It means, even simple queries are time consuming queries. The solution is to handle those databases through Parallel Database Systems, where a table / database is distributed among multiple processors possibly equally to perform the queries in parallel. Such a **system which share resources to handle massive data just to increase the performance of the whole system is called Parallel Database Systems.**

Need

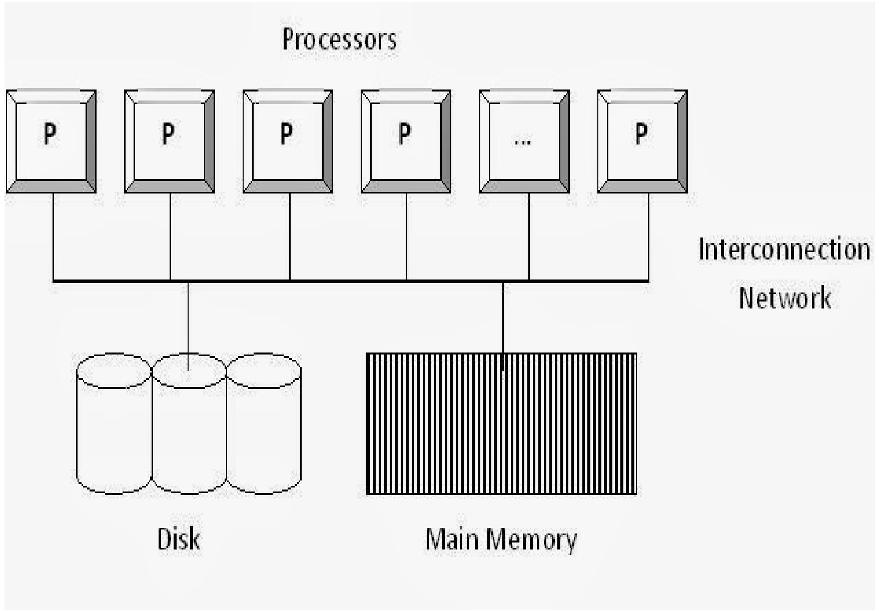
- Multiple resources like CPUs and Disks are used in parallel. The operations are performed simultaneously, as opposed to serial processing. A parallel server can allow access to a single database by users on multiple machines. It also performs many parallelization operations like data loading, query processing, building indexes, and evaluating queries.
- For example, if 50 users consume close to 100 percent of the CPU during normal processing, then adding more users would cause the system to slow down due to contention for limited CPU cycles. However, by adding more CPUs, we can support extra users without degrading performance.

Parallel Database Architecture

- Shared Memory Architecture
- Shared Disk Architecture
- Shared Nothing Architecture

Shared Memory Architecture

In Shared Memory architecture, single memory is shared among many processors as shown in Figure. As shown in the figure, several processors are connected through an interconnection network with Main memory and disk setup. Here interconnection network is usually a high speed network (may be Bus, Mesh) which makes data sharing (transporting) easy among the various components (Processor, Memory, and Disk).



Shared Memory Architecture

- **Advantages:**

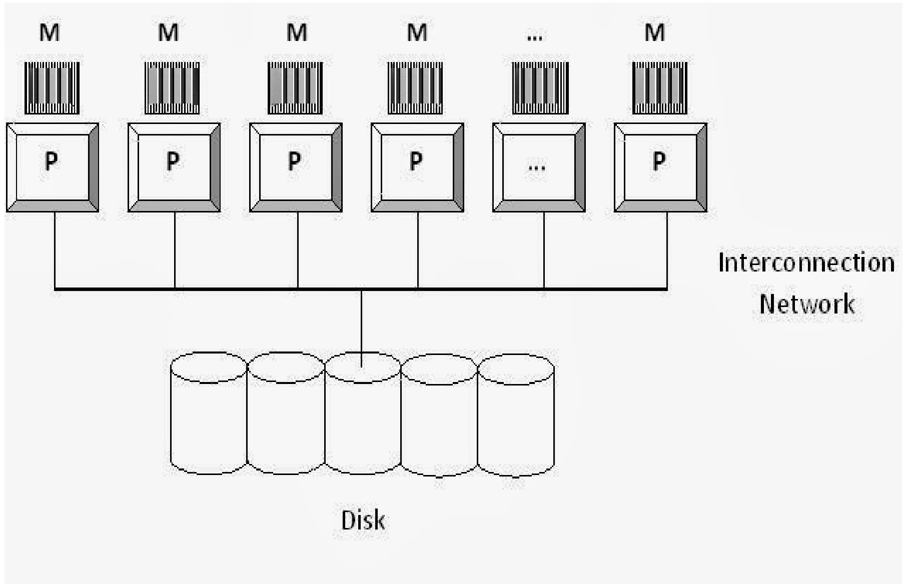
1. Implementation is simple.
2. Establishes effective communication between processors through single memory addresses space.
3. Above point leads to less communication overhead.

- **Disadvantages:**

1. Addition of processor would slow down the existing processors.
2. Degree of Parallelism is limited. More number of parallel processes might degrade the performance.

Shared Disk Architecture

In Shared Disk architecture, single disk or single disk setup is shared among all the available processors and also all the processors have their own private memories as shown in Figure.



Shared Disk Architecture

- Advantages:

1. Failure of any processors would not stop the entire system (Fault tolerance)
2. Interconnection to the memory is not a bottleneck.
3. Support larger number of processors (when compared to Shared Memory architecture)

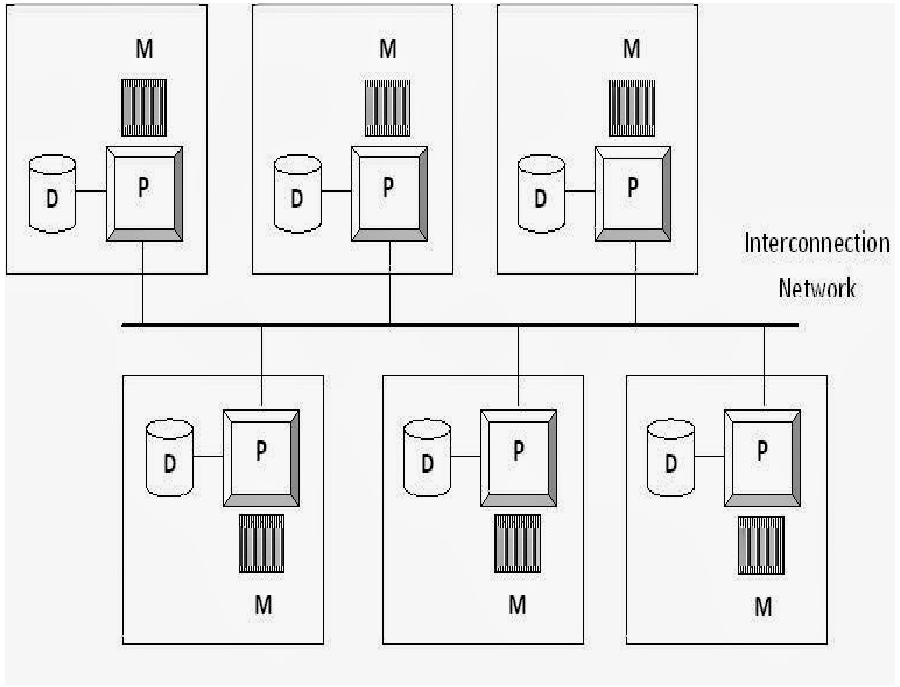
Example Real Time Shared Disk Implementation:
DEC clusters (VMScluster) running Rdb

- Disadvantages:

1. Interconnection to the disk is bottleneck as all processors share common disk setup.
2. Inter-processor communication is slow. The reason is, all the processors have their own memory. Hence, the communication between processors need reading of data from other processors' memory which needs additional software support.

Shared Nothing Architecture

In Shared Nothing architecture, every processor has its own memory and disk setup. This setup may be considered as set of individual computers connected through high speed interconnection network using regular network protocols and switches for example to share data between computers. (This architecture is used in the Distributed Database System). In Shared Nothing parallel database system implementation, we insist the use of similar nodes that are Homogenous systems. (In distributed database System we may use Heterogeneous nodes)



Shared Nothing Architecture

- Advantages:

1. Number of processors used here is scalable. That is, the design is flexible to add more number of computers.
2. Unlike in other two architectures, only the data request which cannot be answered by local processors need to be forwarded through interconnection network.

- Disadvantages:

1. Non-local disk accesses are costly. That is, if one server receives the request. If the required data not available, it must be routed to the server where the data is available. It is slightly complex.
2. Communication cost involved in transporting data among computers.

Example Real Time Shared Nothing Implementation
Teradata
Oracle nCUBE

Data Partitioning Strategies in Parallel Database

There are various partitioning strategies proposed to manage the data distribution into multiple processors evenly.
Let us assume that in our parallel database system we have n processors P0, P1, P2, …, Pn-1 and n disks D0, D1, D2, …, Dn-1 where we partition our data. The value of n is chosen according to the degree of parallelism required. The **partitioning strategies** are,

- Round Robin Partitioning
- Hash Partitioning
- Range Partitioning

Round Robin Partitioning

It is the simplest form of partitioning strategy. Here, the data are distributed into various disks in the fashion, first record into first disk, second record into second disk, and so on. If the number of available disks n is 10, then first record goes to D1 (1 mod 10 = 1), second record goes to D2 (2 mod 10 =2), and so on and 10th record goes to D0 (10 mod 10 = 0), 11th record goes to D1 (11 mod 10 = 1). This scheme distributes data evenly in all the disks.

Hash Partitioning

This strategy identifies one or more attributes as partitioning attributes. We need a hash function which is carefully chosen which takes the identified partitioning attributes as input to hash function.
For **example**, consider the following table;
    EMPLOYEE(ENo, EName, DeptNo, Salary, Age)
If we choose DeptNo attribute as the partitioning attribute and if we have 10 disks to distribute the data, then the following would be a hash function;
    h(DeptNo) = DeptNo mod 10
If we have 10 departments, then according to the hash function, all the employees of department 1 will go into disk 1, department 2 to disk 2 and so on.
As another example, if we choose the EName of the employees as partitioning attribute, then we could have the following hash function;
    h(EName) = (Sum of ASCII value of every character in the name) mod n,
where n is the number of disks/partitions needed.

Range Partitioning

In Range Partitioning we identify one or more attributes as partitioning attributes. Then we choose a range partition vector to partition the table into n disks. The vector is the values present in the partitioning attribute.
For example, for the EMPLOYEE relation given above, if the partitioning attribute is Salary, then the vector would be one as follows;
[5000, 15000, 30000],
where every value means the individual range of salaries. That is, 5000 represents the first range (0 – 5000), 15000 represents the range (5001 – 15000), 30000 represents the third range (15001 – 30000), and it includes the final range which is (30001 – rest). Hence, the vector with 3 values represents 4 disks/partitions.

Problem Solving on Round Robin, Hash and Range Partitioning

Types of Parallelism

- Inter-query Parallelism

- Intra-Query Parallelism

Inter-query Parallelism

- It is a form of parallelism where many different Queries or Transactions are executed in parallel with one another on many processors.
- **Advantages:**

1. It increases Transaction Throughput. That is, number of transactions executed in a given time can be increased.
2. It scales up the Transaction processing system. Hence, best suited for On-Line Transaction Processing (OLTP) systems.

- **Supported Parallel Database Architectures**
- It is easy to implement in Shared Memory Parallel System. Lock tables and Log information are maintained in the same memory. Hence, it is easy to handle those transactions which shares locks with other transactions. Locking and logging can be done efficiently.
- In other parallel architectures like Shared Disk and Shared Nothing, the locking and logging must be done through message passing between processors, which is considered as costly operation when compared Shared Memory Parallel architecture. Cache coherency problem would occur.
- **Example Database systems which support Inter-query Parallelism**

  Oracle 8 and Oracle Rdb

Intra-Query Parallelism

- It is the form of parallelism where Single Query is executed in parallel on many processors.
- **Advantages:**

1. To speed up a single complex long running queries.
2. Best suited for complex scientific calculations (queries).

- **Supported Parallel Database Architectures**

SharedMemory, Shared Disk and Shared Nothing parallel architectures are supported. We need not worry about locking and logging as because it involves parallelizing single query.

- Types

1. Intra-operation parallelism – the process of speeding up a query through parallelizing the execution of individual operations. The operations which can be parallelized are Sort, Join, Projection, Selection and so on.
2. Inter-operation parallelism – the process of speeding up a query through parallelizing various operations which are part of the query. For example, a query which involves join of 4 tables can be executed in parallel in two processors in such a way that each processor shall join two relations locally and the result1 and result2 can be joined further to produce the final result.

- **Example Database systems which support Intra-query Parallelism**

Informix, Terradata