PARALLEL DATABASE SERVERS.

This Chapter first introduces the basic types of architectures for parallel database servers, in respect of the distribution strategies for main memory and disks. Then we briefly review four state-of-the-art parallel database servers. The first three systems covered are the Teradata system, the ICL Goldrush MegaSERVER running Oracle Parallel Server and the IBM SP2 running DB2 Parallel Edition. These three systems are commercially available. The fourth parallel database server reviewed in this Chapter is Monet, a research system designed to support advanced database applications.

8.1 ARCHITECTURES OF PARALLEL DATABASE SERVERS.

Parallel Database Servers can be divided into three categories, according to the degree of distribution of main memory and disks [DeWitt & Gray 92], [Bergsten et al. 93], [Valduriez 93], [Stonebraker 86]. Hybrid systems also occur in practice, as we will see later. These categories are illustrated in Figure 8-1 - where P stands for processor and M stands for memory - and discussed in the following Subsections.

8.1.1 Shared Everything (or Shared Memory).

In this architecture all processors have access to a common, large main memory (RAM) and to all disks, as shown in Figure 8-1(a). This is also known as a symmetric shared-memory multiprocessor (SMP) architecture. Load balancing can be automatically done and parallelization of database operations is easy, since shared memory is an inter-processor communication model more flexible than distributed memory (Section 6.3). The major drawback of shared-memory machines is that they are more difficult to scale up as the number of processors increases, because it is expensive to build hardware for providing simultaneous access to a shared memory by a large number of processors. If the processors each have significant cache, then cache coherency may also become an issue. Shared-memory machines nowadays have scalability limited to a few tens of processors.
8.1.2 Shared Disks.

In this architecture the main memory is distributed among the processors, but each processor has access to all disks, as shown in Figure 8-1(b). All inter-processor communication and I/O occurs via message passing through the interconnection network. Any processor can request data on any disk. The transmission of the requested data from disks to processors characterizes the data-shipping paradigm for parallel database systems. Unlike shared-nothing architectures (see below), there is no static assignment of data to specific processors. Hence, the number and the identity of the processors allocated to perform a given relational operation can be dynamically decided, depending on the processor's current workload [Rahm 96]. That is, this architecture facilitates dynamic load balancing, in comparison with shared-nothing. (Of course, the architecture that most facilitates load balancing is still shared everything, as discussed above.) This architecture has better scalability, in comparison with shared-memory systems. Its major drawback is the high traffic of data in the interconnection network.

8.1.3 Shared Nothing.

In this architecture the main memory is distributed among the processors, and each processor manages its own disk(s), as shown in Figure 8-1(c). Hence, all processors can access their corresponding disks in parallel, minimizing the classical I/O bottleneck in database systems. Each processor can independently process its own data, and the processors communicate with each other - via the interconnection network - only to send requests and receive results. This characterizes the function-