INTRODUCTION

The amount of data stored in databases have been growing exponentially over the recent years in both transactional and data warehouse environments. In addition to the enormous data growth users require faster processing of the data to meet business requirements.

Parallel execution is key for large scale data processing. Using parallelism, hundreds of terabytes of data can be processed in minutes, not hours or days. Parallel execution uses multiple processes to accomplish a single task. The more effectively the database can leverage all hardware resources - multiple CPUs, multiple IO channels, multiple storage units, multiple nodes in a cluster - the more efficiently queries and other database operations will be processed.

47

Large data warehouses should always use parallel execution to achieve good performance. Specific operations in OLTP applications, such as batch operations, can also significantly benefit from parallel execution. This paper covers three main topics:

- **Fundamental concepts of parallel execution** – why should you use parallel execution and what are the fundamental principles behind it.
- **Oracle's parallel execution implementation and enhancements –** here you will become familiar with Oracle's parallel architecture, learn Oracle-specific terminology around parallel execution, and understand the basics of how to control and identify parallel SQL processing.
- **Controlling parallel execution in the Oracle Database** – this last section shows how to enable and control parallelism within the Oracle environment, giving you an overview of what a DBA needs to think about.

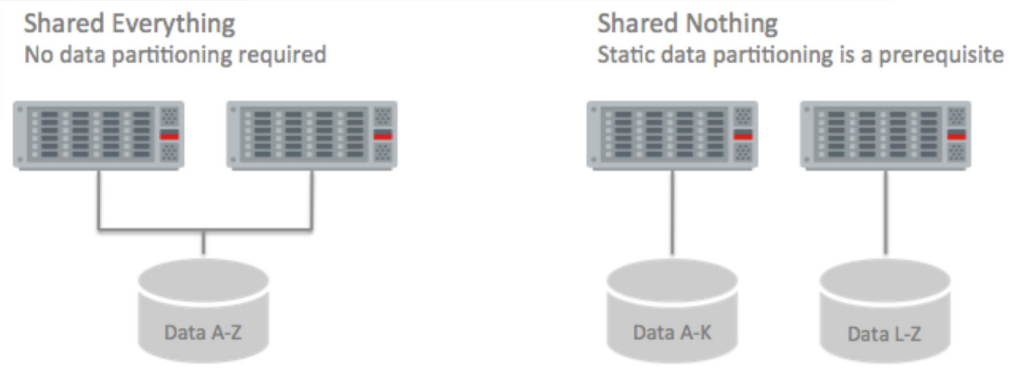

Figure 1: Shared everything versus shared nothing

In a **shared nothing system**, the system is physically divided into individual parallel processing units. Each processing unit has its own processing power (CPU cores) and its own storage component; its CPU cores are solely responsible for its individual data set on its own storage. The only way to access a specific piece of data is to use the processing unit that owns this subset of data. Such systems are also commonly known as Massively Parallel Processing (MPP) systems. Data partitioning is a fundamental prerequisite for these systems. In order to achieve a good workload distribution shared nothing systems have to use a hash algorithm to statically partition data evenly across all available processing units. The data partitioning strategy that controls the data placement has to be decided upon initial creation of the system.

47

As a result, shared nothing systems introduce mandatory, fixed minimal parallelism in their systems in order to perform operations that involve table scans; the fixed parallelism completely relies on the fixed static data partitioning at database or object creation time: the number of parallel processing units determines the minimal degree of parallelism to access all partitions of the data. Most non-Oracle data warehouse systems are shared nothing systems.

Oracle Database relies on a **shared everything architecture**. This architecture does not require any pre-defined data partitioning to enable parallelism; all of the data is accessible from all processing units without limitations; the degree of parallelism for an operation is decoupled from the actual data storage. However, by using Oracle Partitioning, Oracle Database can operate on the same processing paradigm, offering the exact same parallel processing capabilities as a shared nothing system. It is worth noting that it does so without the restrictions of the fixed parallel access encompassed in the data layout. Consequently, Oracle can parallelize almost all operations in various ways and degrees, independent of the underlying data layout, in addition to the parallel capabilities of a shared nothing system. By using a shared everything architecture Oracle allows flexible parallel execution and high concurrency without overloading the system, using a superset of parallel execution capabilities over shared nothing vendors.