Chapter 3.21
Database Benchmarks

Jérôme Darmont
ERIC, University of Lyon 2, France

INTRODUCTION

Performance measurement tools are very important, both for designers and users of Database Management Systems (DBMSs). *Performance evaluation* is useful to designers to determine elements of architecture, and, more generally, to validate or refute hypotheses regarding the actual behavior of a DBMS. Thus, *performance evaluation* is an essential component in the development process of well-designed and efficient systems. Users may also employ *performance evaluation*, either to compare the efficiency of different technologies before selecting a DBMS, or to tune a system.

Performance evaluation by experimentation on a real system is generally referred to as benchmarking. It consists of performing a series of tests on a given DBMS to estimate its performance in a given setting. Typically, a *benchmark* is constituted of two main elements: a database model (conceptual schema and extension), and a workload model (set of read and write operations) to apply on this database, following a predefined protocol. Most *benchmarks* also include a set of simple or composite performance metrics such as response time, throughput, number of input/output, disk or memory usage, and so forth.

The aim of this article is to present an overview of the major families of state-of-the-art database benchmarks, namely, relational benchmarks, object and object-relational benchmarks, XML benchmarks, and decision-support benchmarks; and to discuss the issues, tradeoffs, and future trends in database benchmarking. We particularly focus on XML and decision-support benchmarks, which are currently the most innovative tools that are developed in this area.

BACKGROUND

Relational Benchmarks

In the world of relational DBMS benchmarking, the *Transaction Processing Performance Council (TPC)* plays a preponderant role. The mission of this non-profit organization is to issue standard benchmarks, to verify their correct application by users, and to regularly publish performance tests results. Its benchmarks all share variants of a classical business database (*customer-order-*

product-supplier) and are only parameterized by a scale factor that determines the database size (*e.g.*, from 1 to 100,000 GB).

The *TPC* benchmark for transactional databases, TPC-C (TPC, 2005a), has been in use since 1992. It is specifically dedicated to On-Line Transactional Processing (OLTP) applications, and features a complex database (nine types of tables bearing various structures and sizes), and a workload of diversely complex transactions that are executed concurrently. The metric in TPC-C is throughput, in terms of transactions.

There are currently few credible alternatives to *TPC-C*. Although, we can cite the Open Source Database Benchmark (OSDB), which is the result of a project from the free software community (SourceForge, 2005). OSDB extends and clarifies the specifications of an older benchmark, AS^{3}AP. It is available as free C source code, which helps eliminate any ambiguity relative to the use of natural language in the specifications. However, it is still an ongoing project and the benchmark's documentation is very basic. AS^{3}AP's database is simple: it is composed of four relations whose size may vary from 1 GB to 100 GB. The workload is made of various queries that are executed concurrently. OSDB's metrics are response time and throughput.

Object-Oriented and Object-Relational Benchmarks

There is no standard benchmark for object-oriented DBMSs. However, the most frequently cited and used, OO1 (Cattel, 1991), HyperModel (Anderson, Berre, Mallison, Porter, & Schneider, 1990), and chiefly *OO7* (Carey, DeWitt, & Naughton, 1993), are *de facto* standards. These benchmarks mainly focus on engineering applications (*e.g.*, computer-aided design, software engineering). They range from OO1, which bears a very simple schema (two classes) and only three operations, to *OO7,* which is more generic and proposes a complex and tunable schema (ten classes), as well as fifteen complex operations. However, even *OO7,* the more elaborate of these benchmarks, is not generic enough to model other types of applications, such as financial, multimedia, or telecommunication applications (Tiwary, Narasayya, & Levy, 1995). Furthermore, its complexity makes it hard to understand and implement. To circumvent these limitations, the *OCB* benchmark has been proposed (Darmont & Schneider, 2000). Wholly tunable, this tool aims at being truly generic. Still, the benchmark's code is short, reasonably easy to implement, and easily portable. Finally, *OCB* has been extended into the Dynamic Evaluation Framework (DEF), which introduces a dynamic component in the workload, by simulating access pattern changes using configurable styles of changes (He & Darmont, 2005).

Object-relational benchmarks such as BUCKY (Carey, DeWitt, & Naughton, 1997) and BORD (Lee, Kim, & Kim, 2000) are query-oriented and solely dedicated to object-relational systems. For instance, BUCKY only proposes operations that are specific to these systems, considering that typical object navigation is already addressed by object-oriented benchmarks. Hence, these benchmarks focus on queries implying object identifiers, inheritance, joins, class and object references, multivalued attributes, query unnesting, object methods, and abstract data types.

XML Benchmarks

Since there is no standard model, the storage solutions for XML (eXtended Markup Language) documents that have been developed since the late nineties bear significant differences, both at the conceptual and the functionality levels. The need to compare these solutions, especially in terms of performance, has lead to the design of several benchmarks with diverse objectives.

X-Mach1 (Böhme & Rahm, 2001), XMark (Schmidt, Waas, Kersten, Carey, Manolescu, & Busse, 2002), XOO7 (an extension of OO7) (Bressan, Lee, Li, Lacroix, & Nambiar, 2002) and

Related Content