Caching and Materialization for Web Databases

Alexandros Labrinidis
University of Pittsburgh
USA
labrinid@cs.pitt.edu

Qiong Luo
Hong Kong University of Science and Technology
Hong Kong
luo@cse.ust.hk

Jie Xu
University of Pittsburgh
USA
xujie@cs.pitt.edu

Wenwei Xue
Hong Kong University of Science and Technology
Hong Kong
wwxue@cse.ust.hk

Full text available at: http://dx.doi.org/10.1561/1900000005
Editorial Scope

Foundations and Trends® in Databases covers a breadth of topics relating to the management of large volumes of data. The journal targets the full scope of issues in data management, from theoretical foundations, to languages and modeling, to algorithms, system architecture, and applications. The list of topics below illustrates some of the intended coverage, though it is by no means exhaustive:

- Data Models and Query Languages
- Query Processing and Optimization
- Storage, Access Methods, and Indexing
- Transaction Management, Concurrency Control and Recovery
- Deductive Databases
- Parallel and Distributed Database Systems
- Database Design and Tuning
- Metadata Management
- Object Management
- Trigger Processing and Active Databases
- Data Mining and OLAP
- Approximate and Interactive Query Processing
- Data Warehousing
- Adaptive Query Processing
- Data Stream Management
- Search and Query Integration
- XML and Semi-Structured Data
- Web Services and Middleware
- Data Integration and Exchange
- Private and Secure Data Management
- Peer-to-Peer, Sensors and Mobile Data Management
- Scientific and Spatial Data Management
- Data Brokering and Publish/Subscribe
- Data Cleaning and Information Extraction
- Probabilistic Data Management

Information for Librarians
Foundations and Trends® in Databases, 2009, Volume 2, 4 issues. ISSN paper version 1931-7883. ISSN online version 1931-7891. Also available as a combined paper and online subscription.
Caching and Materialization for Web Databases

Alexandros Labrinidis\textsuperscript{1}, Qiong Luo\textsuperscript{2}, Jie Xu\textsuperscript{3} and Wenwei Xue\textsuperscript{4}

\textsuperscript{1} University of Pittsburgh, USA, labrinid@cs.pitt.edu
\textsuperscript{2} Hong Kong University of Science and Technology, Hong Kong, luo@cse.ust.hk
\textsuperscript{3} University of Pittsburgh, USA, xujie@cs.pitt.edu
\textsuperscript{4} Hong Kong University of Science and Technology, Hong Kong, wwxue@cse.ust.hk

Abstract

Database systems have been driving dynamic websites since the early 1990s; nowadays, even seemingly static websites employ a database back-end for personalization and advertising purposes. In order to keep up with the high demand fuelled by the rapid growth of the Internet, a number of caching and materialization techniques have been proposed for web databases over the years. The main goal of these techniques is to improve performance, scalability, and manageability of database-driven dynamic websites, in a way that the quality of data is not compromised. Although caching and materialization are well-understood concepts in the traditional database and networking/operating systems literature, the Web and web databases bring forth unique characteristics that warrant new techniques and approaches.
In this monograph, we adopt a data management point of view to describe the system architectures of web databases, and analyze the research issues related to caching and materialization in such architectures. We also present the state-of-the-art in caching and materialization for web databases and organize current approaches according to the fundamental questions, namely how to store, how to use, and how to maintain cached/materialized web data. Finally, we associate work in caching and materialization for web databases to similar techniques in other related areas, such as data warehousing, distributed systems, and distributed databases.
Contents

1 Introduction 1

2 Typical Architecture 5

3 Store How are Data Cached/Materialized? 9
   3.1 Store Location of Caching/Materialization 9
   3.2 Store Unit of Caching/Materialization 12
   3.3 Store Selecting Content for Caching/Materialization 12

4 Use Using Cached Content 15
   4.1 Use Query Scheduling 15
   4.2 Use Query Processing 17

5 Maintain Cache Maintenance 23
   5.1 Maintain Cache Initialization: Proactive versus Reactive 23
   5.2 Maintain Timing for Updates 25
   5.3 Maintain Processing of Updates 31
   5.4 Maintain Scheduling of Updates 34
   5.5 Maintain Cache Replacement Policies 35
1

Introduction

Database systems have been driving dynamic websites since the early 1990s, and caching and materialization have been the major techniques to improve the performance, scalability, and manageability of such web databases. Different from a traditional database environment, the software components of a web database, including web servers, database servers, application servers, and possibly additional middleware, are largely independent from one another, even though they work together as a holistic system (Figure 1.1). Caching and materialization techniques for such web databases consider a number of issues at different parts of the system and they bring interesting challenges and opportunities.

In addition to the inherent architectural uniqueness, web databases also come with stringent demands for near-real-time performance (at the speed of thought) and ability to withstand high request volumes (e.g., due to flash crowds), giving caching and materialization techniques a pivotal role in such environments.

In this monograph, we look at the entire process of caching and materialization for web databases as a sequence of actions (or verbs). For each action, we identify the possible options and present
representative techniques and papers for each one. Specifically, we look into the following actions, as they relate to caching and materialization for web databases:

- **Store**: How is data cached/materialized, including where and at what granularity to cache/materialize data, and how to select the data items to cache/materialize.
- **Use**: How do we use cached/materialized data items to answer new requests?
- **Maintain**: How are caches maintained up-to-date, including when and how to handle updates, how to organize the cache to answer new requests efficiently, and how to perform cache replacement as necessary.

When presenting existing techniques on these three actions, we focus on the performance and quality aspect (in Section 6). In brief, we examine the following three types of evaluation metrics: (1) Quality of Service (QoS) metrics, including response time, throughput, and availability; (2) Quality of Data (QoD) metrics, mainly about data freshness and accuracy, and (3) user-centric metrics, usually in the form of quality contracts.

Among the above evaluation metrics, throughput is a major one to consider for web databases, and one way to achieve a high throughput is through high-performance hardware configurations. However, enhancing system throughput using high-end configurations is not
always desirable; a major factor is the Total Cost of Ownership (TCO) \[43\]. TCO is the direct and hidden lifetime IT cost of purchasing, operating, managing, and maintaining a computing system \[31\]. Example cost items include hardware, software, network communication, administration, personnel training, and technical support. For a database-backed website, an increase in the throughput through high-end configurations may result in the increase of all those TCO cost items and in turn the TCO itself. Therefore, system throughput should be considered together with TCO. In particular, transparent and adaptive caches are often an ideal candidate to increase system throughput automatically with little increase on TCO.

Note that an important factor to consider in any caching and materialization technique for web databases is whether the scheme supports database transaction semantics, even though many web applications today work well with a weaker consistency guarantee. Eric Brewer proposed the CAP Theorem \[23\], which states that, for any distributed computing system, only two out of the following three key requirements can be satisfied: Consistency, Availability, and Partition tolerance. The theorem was later formally proved by Gilbert and Lynch \[61\] for both asynchronous and partially synchronous network models. As such, in real-world distributed web databases, where availability and scalability are crucial, it is unavoidable to make compromises on consistency. Therefore, caching and materialization are a good match to such systems in that they can help on both availability and partition tolerance and can benefit from the relaxed consistency requirement. In practice, many commercial database-backed websites today have focused their technical development, including deploying caches throughout all levels of the system, on ensuring service availability and tolerance upon data partitioning/distribution, while sacrificing consistency to various degrees. Well-known examples of such websites include Amazon, EBay, and Twitter.

**Roadmap:** The rest of this monograph is structured into four separate components as follows:

- **Architecture** — this part describes a typical web-database architecture and discusses some of the complexities intrinsic in such setups (Section\[2\]).
Introduction

- **Taxonomy** — this part presents the three “verbs” in detail, along with the possible alternatives. We also give a detailed description of a typical web-database architecture (Sections 3–5), followed by a discussion on metrics, in Section 6.
- **Projects** — this part describes some representative web-database projects, while explaining the choices each project adopted under the presented taxonomy (Section 7).
- **Related work** — this section presents related work from other areas (Section 8).

We conclude this monograph with a short discussion on some open problems and future directions.
References

References


References

References


References


References

References


References

References

