ABSTRACT

With the explosion of data, several applications are designed around analytical aspects, with data warehousing technology at the heart of the construction chain. The exploitation of this data warehouse is usually performed by the use of complex queries involving selections, joins and aggregations. These queries bring the following characteristics: (1) their routinely aspects, (2) their large number, and (3) the high operation sharing between queries. This interaction has been largely identified in the context of multi-query optimization, where graph data structures were proposed to capture it. Also during the physical design, the structures have been used to select redundant optimization structures such as materialized views and indexes. Horizontal data partitioning (HDP) is another non-redundant optimization structure that can be selected in the physical design phase. It is a pre-condition for designing extremely large databases in several environments: centralized, distributed, parallel and cloud. It aims to reduce the cost of the above operations. In HDP, the optimization space of potential candidates for partitioning grows exponentially with the problem size making the problem NP-hard. This paper proposes a new approach based on query interactions to select a partitioning schema of a data warehouse in a divide and conquer manner to achieve an improved trade-off between the optimization algorithm’s speed and the quality of the solution. The effectiveness of our approach is proven through a validation using the Star Schema Benchmark (100 GB) on Oracle11g.

Keywords: Database, Horizontal Data Partitioning, Multi-Query Optimization, Optimization Structure, Query Interaction

1. INTRODUCTION

Multiple Query Optimization (M/QO) is one of the most important characteristics of query processing in the context of database field (see Figure 1). M/QO was clearly identified and studied by Timos Sellis in 1988 in traditional relational databases (Sellis, 1988). Two years later, he formalized the problem of M/QO (PMQO) and studied its complexity (NP-complete problem) (Sellis & Ghosh, 1990). The basic idea of the PMQO is: given a set of queries to evaluate, how to exploit the common sub-expressions by either sharing or materializing them. Note that database workloads consist of mixes of queries that run concurrently and interact with each other (Ahmad, Aboulnaga & Babu, 2009). Due to the characteristics of relational...
algebraic operations, many equivalent forms of a query exist. Therefore, various algorithms were proposed to deal with the PMQO (Sellis & Ghosh, 1990).

The particularity of the MQO is that it spans the different generations of databases: traditional centralized databases (Ahmad, Aboulnaga & Babu, 2009), object oriented databases (Banerjee, Kim & Kim, 1988; Mitchell, Zdonik & Dayal, 1993), data warehousing (Kerkad, Bellatreche & Geniet, 2013), distributed databases (Furtado, 2009), semantic databases (Goasdoué et al. 2011; Le et al. 2012), map-reduce databases (Wang & Chan 2013), etc.

In object oriented databases, MQO has been used for instance in O2 system, (Mitchell, Zdonik & Dayal, 1993) where common sub-expressions (path expression representing object queries) were factorized to guide the query process and contribute on determining access plans of object queries. With the spectacular development of data warehouses, the MQO re-emerged due to characteristics of star join queries. Indeed, all individual join operations of a star-join query reference the fact table that is usually the largest table. Evaluating a star-join query is expensive because the fact table participates in every join operation. Star join queries increase the probability of sharing sub-expressions. This characteristic has been exploited to optimize these queries (Microsoft) as well as to perform the physical design phase. To be more precise, let us give a formalization of the physical design, which has gain importance as query optimizers became sophisticated to cope with complex decision support applications (Chaudhuri & Narasayya, 2007). Let $Q$, OS and CS be respectively a set of queries and optimization structures classes supported by a given DBMS, and constraints associated to each OS. Then the physical design problem consists in selecting instances of OS so that the execution cost of $Q$ is reduced under the corresponding constraints. The connection between MQO and the physical design phase was mainly focused on the selection of two optimization structures which are materialized view selection (Yang, Karlapalem & Li, 1997) and indexes (Labio et al., 1997).

The PMQO has also been studied in the context of distributed databases (DDB). Particularly, the work of (Kementsietsidis et al. 2012) discusses the PMQO for exploratory “trial-and-error” queries in E-commerce applications. As a consequence, MQO is revisited since evaluating exploratory queries involves the evaluation of a large number of distributed queries that have to be combined.

Semantic databases (SDB) and SPARQL and RDF queries also benefit from the MQO, and research efforts were focused on two directions: (i) the proposition of algorithms for the MQO problem taking into account the characteristic of semantic databases (Goasdoué et al. 2011) and the fact that SPARQL typically involve many joins and increases the probability that some joins may be shared and (ii) the exploitation of MQO in selecting materialized views for semantic databases modeled by RDF triples as in the data warehouse context.

With the development of the paradigm of MapReduce databases for large-scale data analysis that brings the notion of job sharing, MQO was a serious candidate to exploit the

Figure 1. The role of MQO and its connection with physical design phase
A Survey on Temporal Data Warehousing
www.igi-global.com/article/survey-temporal-data-warehousing/1820?camid=4v1a