Concept-Oriented Query Language

Alexandr Savinov
Technische Universität Dresden, Germany

INTRODUCTION

With the explosion of data volume and the variety of data sources (Cohen et al., 2009) – two aspects of the *big data* problem - we observe quite significant difficulties in applying conventional *data analysis* methodologies to real world problems. The existing technologies for data management and analytics were pushed to the limits of their ability to solve more and more complex analysis tasks:

- **Agile Analytics:** Perhaps the most widely used methodology for data analysis during several decades is based on the multidimensional metaphor where data is viewed as existing in a multidimensional space. A problem of this approach is that it is based on application-specific scenarios with predefined roles of dimensions, measures, cubes and facts. Changing such scenarios is a quite difficult task because they are embedded in both database systems and client software. The goal of agile analytics consists in going beyond standard OLAP analysis by facilitating exploratory ad-hoc analytics where the user can freely vary all data processing and visualization parameters.
- **Self-Service Analytics:** The conventional approach to analysis is to approach IT department which however has several drawbacks: business frequently does not trust data provided by IT, IT is unable to understand the needs of the user (and this leads to frustration and low motivation), IT might not be able to respond to user requests as quickly as is desirable (and the requirements may well change during the response time), existing BI tools are not intended for non-professional users. Self-service analytics is one of the most significant trends in the BI industry over the last few years and these tools aim to give non-professional users the ability to solve analytical tasks with little or no help from IT.
- **Near Real Time Analytics:** It may take days to generate a BI report in a typical enterprise system and there is strong demand in reducing the time between data acquisition and making a business decision. One of the core problems is that traditional systems are based on two separate technology stacks: for transactional workload and for analytical workload. The design principles and techniques of these two subsystems are quite different and they cannot provide the necessary response time and agility of decision making on large volumes of data (Chaudhuri, Dayal, & Narasayya, 2011; Thiele & Lehner 2012). Although modern hardware provides a basis for a new generation of in-memory, columnar databases (Boncz, 2012; Larson, 2013) with potentially higher query performance on analytical workloads, it is important to understand that real time analytics is not a hardware problem - new data models, new query languages, new analysis scenarios, new analysis algorithms are needed.
- **Semantic Analysis:** The conventional approach is that it is the task of the human analyst to understand the meaning of data while the system has to only execute precise queries. However, a typical enterprise system can contain tens of thousands data tables and open systems can involve nu-

DOI: 10.4018/978-1-4666-5202-6.ch046

merous external data sources. In this situation it is extremely difficult to get meaningful results manually. Existing solutions add semantics via a separate layer which is based on quite different data modeling and analysis techniques. This leads to complex mappings and translations at all levels of the system architecture.

- **Reasoning about Data:** The goal of this type of analysis is to answer questions by automatically deriving them from the available data. This task has been a prerogative of the systems based on formal logic which have several drawbacks: formal logic is not natural for expressing analysis tasks, formal logic is not very suitable for numeric analysis, formal logic requires a separate system because it is not directly compatible with available data storage, queries in formal logic are computationally expensive.
- **Analytical Computations:** Analysis is not limited by the operations of grouping and aggregation. Now analysts need to embed arbitrary computations in their analysis tasks. Such tasks are normally expressed as batch jobs where data is exported from one or many databases and then processed using an analysis program. Executing arbitrary analysis tasks close to the data (ideally directly where data resides) is still a big problem. It is actually a new incarnation of the old problem of incompatibility between programming and data modeling (impedance mismatch) because data is modeled and manipulated differently in programming languages and databases.

These fundamental challenges require a principled solution rather than yet another specific technique. Many of the above problems can be solved at the level of a unified data model which should be general enough to cover major analytical patterns of thought, and at the same time should it be simple and natural. In this article we describe a novel query language, called the concept-oriented query language (COQL), which addresses the above issues and is aimed at radically *simplifying* typical data analysis and data modeling tasks. COQL is a syntactic description of the concept-oriented model (COM) (Savinov, 2009, 2011) and it has the following distinguishing features:

- COQL replaces joins as a means of connectivity by a novel *arrow notation* which can be viewed as a set-oriented analog of dot notation.
- COQL replaces group-by operation by a novel operation of de-projection.
- COQL introduces a novel mechanism of inference based on the multidimensional structure of data instead of using logical inference.
- COQL inherently supports dimensions as a basic construct rather than treating them as something optional that is added for specific kinds of analysis.
- COM and COQL support several data modeling and analysis paradigms (relational, multidimensional, entity-relationship, semantic and conceptual, object-oriented) by resolving many incompatibilities and controversies as well as increasing semantic integrity of data models and analysis tasks.
- COQL relies on a novel data typing construct, called concept, and two relations: inclusion and partial order.

BACKGROUND

A language reflects main principles of the underlying model or paradigm using syntactic constructs and rules. It is valid for programming languages, query language, conceptual languages and for other areas where a theory can be described syntactically. Below we describe major language categories used for data querying with the focus on data analysis, connectivity and set operations.

Join-based languages:. It is probably the most wide spread class of query languages which rely