Bernhard Jaecksch, Franz Faerber, Frank Rosenthal, Wolfgang Lehner

Hybrid Data-Flow Graphs for Procedural Domain-Specific Query Languages

Erstveröffentlichung in / First published in:

Scientific and Statistical Database Management 23rd International Conference. Portland, 20.-22.07.2011. Springer, S. 577–578. ISBN 978-3-642-22351-8.

DOI: http://dx.doi.org/10.1007/978-3-642-22351-8_44

Final edited form was published in "Scientific and Statistical Database Management: 23rd International Conference. Portland 2011", S. 577–578, ISBN 978-3-642-22351-8
http://dx.doi.org/10.1007/978-3-642-22351-8_44

Hybrid Data-Flow Graphs for Procedural Domain-Specific Query Languages

Bernhard Jaecksch[1], Franz Faerber[1], Frank Rosenthal[2], and Wolfgang Lehner[2]

[1] SAP AG, Dietmar-Hopp-Allee 16, 69190 Walldorf, Germany
{b.jaecksch,franz.faerber,wolfgang.lehner}@sap.com
[2] TU Dresden, Institute for System Architecture,
Database Technology Group, 01062 Dresden, Germany
frank.rosenthal@tu-dresden.de

Abstract. Domain-specific query languages (DSQL) let users express custom business logic. Relational databases provide a limited set of options to execute business logic. Usually, stored procedures or a series of queries with some glue code. Both methods have drawbacks and often business logic is still executed on application side transferring large amounts of data between application and database, which is expensive. We translate a DSQL into a hybrid data-flow execution plan, containing relational operators mixed with procedural ones. A cost model is used to drive the translation towards an optimal mixture of relational and procedural plan operators.

1 Introduction

Relational databases provide with SQL a standardized and powerful query language. Although, SQL can be considered as a domain-specific language (DSL), its scope is broad and generic. To keep the user within confined boundaries of specific application domains a domain-specific query language (DSQL) is better suited to the task. Stored procedures incorporate procedural and declarative elements and can be used to express business logic. But they are pre-compiled into native C programs with embedded SQL commands. Hence, execution is driven by procedural C-code interspersed with SQL statements. This makes it difficult to optimize the entire procedure.

2 Contribution

We propose a mechanism to translate a procedural DSQL into the data-flow execution model of the underlying database. Normally, a database execution plan contains relational operators and the graph describes the execution in a declarative way. To overcome the mismatch between an imperative language and the declarative plan, we introduce a hybrid execution plan incorporating both aspects in one plan. Different, to typical stored procedure translation, we translate into a data-flow graph interspersed with procedural elements.

Final edited form was published in "Scientific and Statistical Database Management: 23rd International Conference. Portland 2011", S. 577–578, ISBN 978-3-642-22351-8
http://dx.doi.org/10.1007/978-3-642-22351-8_44

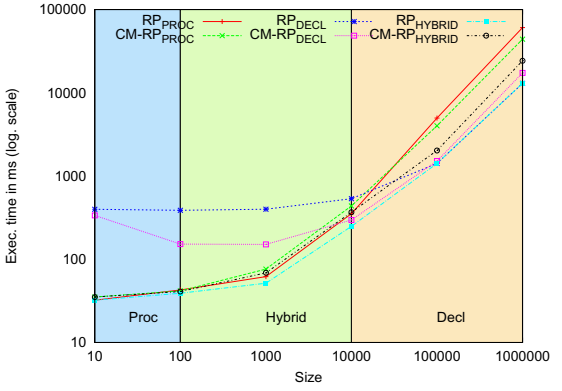

Fig. 1. Actual execution times vs. cost model (CM) predictions for procedural (PROC), declarative (DECL) and hybrid (HYBR)

Translation Type	Percent
Declarative	58%
Hybrid	38%
Procedural	4%

Fig. 2. Translation types of different DSQL scripts

The goal is to translate into a plan with as few procedural elements as possible. Because we express procedural logic by declarative means, we benefit from well-known optimizations and the data-flow graph representation allows the easy exploitation of parallelism. Although, there are many cases where procedural statements can be expressed entirely by relational operators, for others it is impossible. In other cases, both translation variants are possible and we provide a cost model that drives the translation process towards an optimal plan in terms of execution time. We identified procedural patterns that can be expressed in a purely declarative way using a combination of relational operators.

Figure 1 shows evaluation results for an example script translated into a hybrid plan based on our cost model and depending on the input size. The prediction by the cost model are included as well. The shaded areas specify the type of plan that was selected. Our setting is an industrial setup, where an existing DSQL for business planning is used and most complex data processing is done on application side. We see a demand, that the database layer has to provide means to handle business logic from various domains in order to process the tasks closer to the data. To provide an insight, how how typical customer scripts can be translated, we did a survey of custom scripts from over 50 customers, classifying them into translatable to a purely declarative plan, a hybrid plan or only procedural. The results shown in Table 2 suggest that a large percentage of typical scripts can be translated into a hybrid plan with only relational operators.

3 Conclusion

We proposed a translation of a domain-specific query language (DSQL) into a data-flow execution graph that contains relational and procedural operators. Furthermore, we devised a cost model that guides the selection between procedural and relational operators to find an optimal hybrid plan based on the size of the input data.