D-038

# データ仮想化における効率的なクエリ処理手法の提案
# Proposal for a Method of Efficient Query Processing on Data Virtualization

米田 信之† 渡辺 泰之† 齋藤 和広† 村松 茂樹† 小林 亜令†
Nobuyuki Maita† Yasuyuki Watanabe† Kazuhiro Saito† Shigeki Muramatsu† Arei Kobayashi†

## 1. はじめに

近年，企業におけるビッグデータ活用の普及に伴い，更なる競争力向上のためのデータ統合が注目されている．データ統合は，組織内の様々なデータを組み合わせた分析業務を加速させ，新たな価値創造を可能とする．しかしながら，企業内データベース（以下 DB）は既存業務に利用されているものが大半であり，その影響やコストを考慮すると，物理的な統合や再構築は容易ではない．

こうした状況下において活躍の期待される技術の 1 つにデータ仮想化[1]がある．これは，複数 DB を仮想統合し，あたかも単一 DB であるかのようなアクセス性を提供するデータ統合技術である（図 1）．

しかしながら，データ仮想化システム（以下 DVS）におけるクエリ処理には，中間結果の肥大化により処理が著しく遅くなり，場合によっては異常終了に至る問題がある．ビッグデータを対象とする分析業務に適用すると，本問題は特に顕在化する．

本稿では，DVS における中間結果を削減した，効率的なクエリ処理手法を提案する．また TPC-H を用いた評価実験により，提案手法の有効性を検証する．

## 2. DVS 処理における課題

データ仮想化技術の現状を把握するため，既存のデータ仮想化製品について調査を実施した．本調査では，Teiid[2]を用いてデータ仮想化環境を構築し，TPC-H の問合せ処理における振る舞いを確認した．

調査の結果，クエリ処理が著しく遅くなる，あるいは異常終了するケースが見られた．これらの多くは，中間結果すなわち DVS から各 DB へ投稿されるサブクエリの結果が肥大化していることが分かった．

中間結果肥大化の原因の 1 つは，サブクエリへの直積や多対多の結合処理の混入と考えられる．MIND[3]の手法では，ユーザが投稿したクエリについて，各 DB スキーマに合わせサブクエリとして生成する．しかし，サブクエリ生成過程において，直積や多対多の結合条件を生み出しており，中間結果肥大化を招いている．Teiid の手法では，直積を避けたサブクエリ生成を実現しているものの，多対多の関係を残している．

そのほかの原因として，結合処理による絞り込みができていないことが挙げられる．データ仮想化における結合処理は，DVS 上で実行される．生成されるサブクエリは，選択・射影処理のプッシュダウンにとどまり，十分な絞り込みができていないサブクエリが多い．

以上のことから，DVS の処理効率に関する解決すべき課題は次のとおりである．

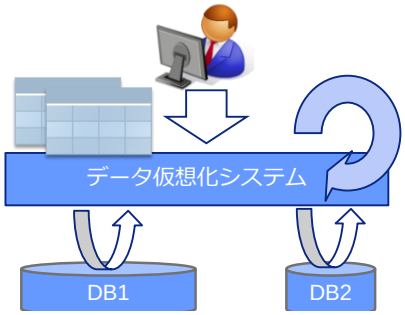


図 1　データ仮想化概要

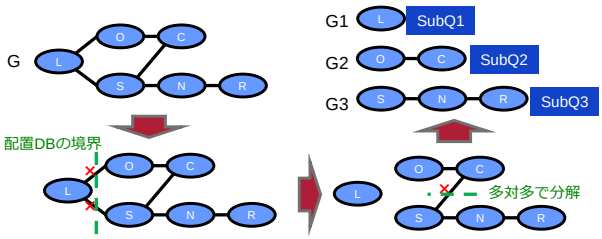


図 2　クエリ分解の例

課題 1：サブクエリ生成で直積・多対多を回避すること
課題 2：サブクエリ抽出件数を絞り込むこと

## 3. 提案手法

課題 1,2 を解決する手法として，クエリ分解手法と中間結果再配置手法の 2 つを提案する．TPC-H を例として，DB1 に Lineitem，DB2 にその他テーブルが配置されているものとして，次節から手法を述べる．

### 3.1 クエリ分解

課題 1 を解決する方法として，クエリグラフ[4]の分解によるサブクエリ生成単位の決定手法を提案する．

まず，ユーザが投稿したクエリを入力として，クエリグラフを生成する．グラフは，ノードがテーブル，エッジがテーブル間結合を表す．次に，各エッジのノード（テーブル）配置先 DB を確認し，DB が異なるエッジでグラフを分解する．図 2 では，Lineitem－Orders 間，Lineitem－Supplier 間で分解される．さらに分解されたグラフに対し，多対多の結合となるエッジにてグラフを分解する．図 2 では，例として Customer－Supplier 間を分解している．一連の操作により得られたグラフをサブクエリの生成単位とすることで，直積・多対多結合の回避が可能である．

† 株式会社 KDDI 研究所，KDDI R&D Laboratories, Inc.

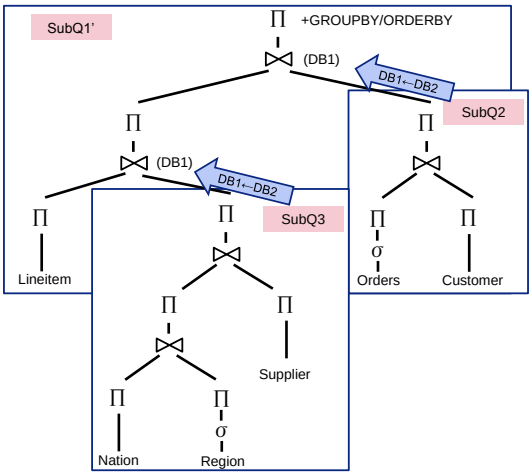


図 3　中間結果再配置と実行計画例

### 3.2 中間結果再配置

課題 2 に対しては，サブクエリの結果を他方の DB に一時的再配置し，結合・集計処理をプッシュダウンする手法を提案する．

3.1 節の手法により生成されたサブクエリ，および結合処理による中間結果サイズを推定し，DVS・DB 間の転送量が最小となるような実行計画を作成する．例として，推定結果から以下が成り立つとする．

・size(SubQ1) >> size(SubQ2), size(SubQ3)

・size(SubQ1) >> size(Join(SubQ3, Join(SubQ1, SubQ2)))

この場合，DB2 から DB1 へ，SubQ2/SubQ3 の結果を再配置し，DB1 に対するサブクエリで Lineitem との結合・集計処理をプッシュダウンすることで，問合せ処理全体の転送量削減が期待される．このときの実行計画例を図 3 に示す．

## 4. 評価実験

提案手法の有効性を検証するため，TPC-H を用いた評価実験を行った．実験では，提案手法にもとづき生成したサブクエリの処理時間と中間結果サイズを確認する．実験環境としてサーバを 3 台用意し，2 台を DBMS (PostgreSQL)，1 台を DVS とした．データセットは TPC-H Scale 1 とし，2 つの DBMS それぞれに，Lineitem とそれ以外のテーブルで分割配置した．提案手法におけるサブクエリと実行計画は，あらかじめ算出した中間結果情報をもとに作成している．作成した実行計画にしたがって，サブクエリの投稿および中間結果の LOAD について，DBMS に順次処理させる．比較対象として，Teiid による実行結果を示す．

実験結果を図 4 に示す．Teiid による Q5－SubQ2（DB2 向け）の中間結果サイズは，約 700MB（データセットの 7 割）に肥大化している．これは，多対多結合の混入が原因である．これについて提案手法では，DB2 向けのサブクエリを 2 つに分解することで，肥大化を回避している．次に，Teiid による SubQ1 を見ると，中間結果サイズがやや大きい．いずれも DB1 の Lineitem に対するサブクエリである．選択/射影によりカラム/件数を絞り込んでいるものの，取得件数の多さからサイズが大きくなる傾向がある．提案手法では，DB2 向けサブクエリの中間結果をそれぞれ DB1 へ再配置することで，DB1 での結合・集計処理をプッシュダウンしており，中間結果サイズを小さくできている．クエリ全体の実行時間を示したものが図 5 であり，提案手法を適用することで，DVS の処理効率化が期待できる．

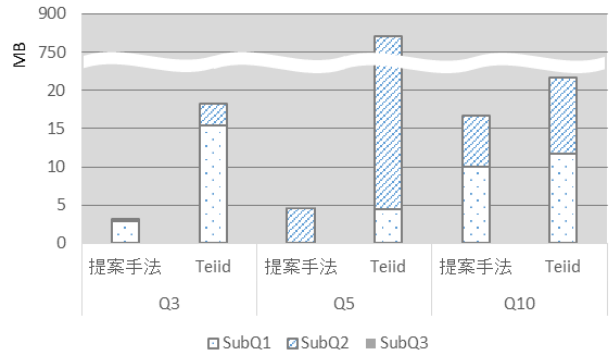


図 4　中間結果サイズの比較

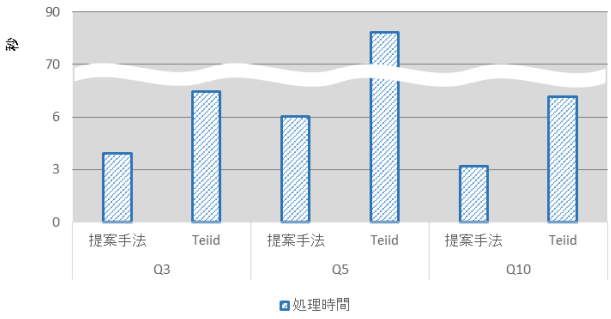


図 5　処理時間の比較

これらの結果から，提案手法を用いることで，Teiid の手法と比較すると，処理時間を最大 10 倍以上高速化できることが確認できた．一方，本手法は DB へのデータ投入も必要となることから，DB 側のリソースや制約事項を踏まえた実行計画の作成や，中間結果サイズ推定技術の精度向上が必要となる．

## 5. おわりに

本稿では，DVS におけるクエリ処理について，サブクエリ分解と中間結果再配置の 2 つの手法を用いて中間結果サイズを削減する，効率的な処理手法を提案した．また，TPC-H を用いた評価実験により，本手法を適用することで処理効率化が期待できることを確認した．

今後の課題は，提案手法を DVS の実行計画機能として実装し，より実用に近い形で評価することである．

### 参考文献

[1] Rick F.,van der Lans, “Data Virtualization for Business Intelligence Systems”, Morgan Kaufmann, 2012

[2] Teiid, http://teiid.jboss.org/

[3] S. Nural., et al., “Query Decomposition and Processing in Multidatabase Systems”, Object Oriented Database Symposium of the 3rd European Joint Conference on Engineering Systems Design and Analysis, pp. 41-52, France, 1996

[4] M. Tamer Özsu. and P. Valduriez., “Principles of Distributed Database Systems Third Edition”, Springer Science, 2011