6X-8 Java 版 SuperSQL エンジンの開発

佐藤 康裕† 赤堀 正剛§
有澤 達也‡ 遠山 元道¶

†慶應義塾大学 理工学部 情報工学科 §慶應義塾大学大学院 理工学研究科 管理工学専攻
‡慶應義塾大学大学院 理工学研究科 計算機科学専攻 ¶慶應義塾大学／ JST さきがけ研究２１

1 はじめに

SuperSQL は質問文の結果から様々な媒体への出力を可能にする SQL の拡張言語である。可能な出力媒体としては、XML、HTML、O2C、LATEX、SMIL 等が挙げられる。

SuperSQL は、SQL の SELECT 節に書かれるターゲットリストの概念を拡張し、タテ、ヨコ、深さ、時間方向 (次元) への結合とグルーピング (反復) の演算子をもつ TFE(TargetForm Expression) の概念を実装し、関係データベース内の正規化された情報から、各種媒体への高品質な出力を可能にしている。

2 SuperSQL

SuperSQL の質問文は、SQL の SELECT 節を拡張し 'GENERATE < medium >< TFE >' という構文を持つ GENERATE 節で置き換えたものである。< medium >に出力したい媒体を指定し、< TFE >に出力のレイアウトを記述する。TFE は、表の構造とレイアウトを規定する式の一種であり、オペランドと演算子から構成する。オペランドは SQL のターゲットリストの要素であり、演算子は結合子と反復子からなる。各方向の結合子及び反復子は以下のように表す。大カッコ内は任意の TFE 式を置くことが出来る。カッコ内に整形結果が複数生成されるとき、これらを指定方向へ連結する。

	結合子	反復子
ヨコ	,	[],
タテ	!	[]!
深さ	%	[]%
時間	#	[]#

図 1 に SuperSQL 質問文の例を示す。

```
GENERATE XML
  [a.value, [b.value,[c.value]!]!]!
FROM ex_a a, ex_b b, ex_c c
WHERE a.id=b.ex_a and b.id=c.ex_b and
c.id=1000;
```

図 1: 質問文 1

3 SuperSQL の実装

システムの概要を図 2 に示す。

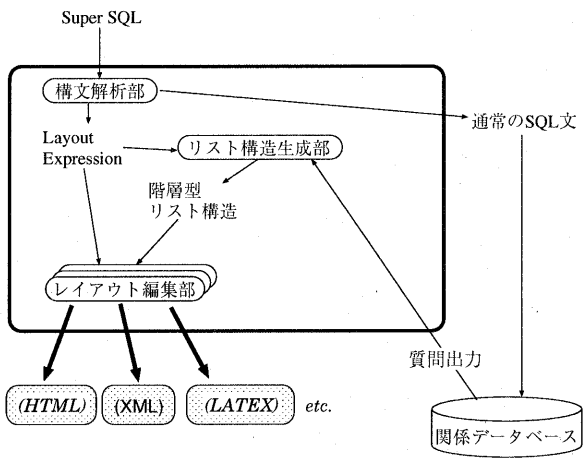

図 2: システム図

3.1 Lisp 版 SuperSQL

今まで SuperSQL の実装には Common Lisp を用いていた。Lisp はリストの扱いが容易であるためである。しかし、Common Lisp を使える環境はあまり多くなく、また JDBC のようなデータベースへの統合インターフェイスが無いため、汎用的な利用は望めない。

SATOU Yasuhiro†, AKAHORI Masatake§, ARISAWA Tatsuya‡, TOYAMA Motomichi¶
†Department of Information and Computer Science, Faculty of Science and Technology, Keio University.
§Department of Administration Engineering,Faculty of Science and Technology, Keio University.
‡Department of Computer Science, Faculty of Science and Technology, Keio University.
¶Keio University. PRESTO, JST.

3.2 Java 版 SuperSQL の実装

Java 版 SuperSQL のプロトタイプは 1998 年に実装された。しかし、図 2 のリスト構造生成部で階層型リスト構造を生成する部分の処理が極端に遅かったため実用には至らなかった。今回はこの部分を改善することで処理の高速化に成功した。

(A B C) というリスト構造から (A B (C)) という入れ子のリスト構造を生成するとする。例えば、(A1 B1 C1),(A1 B1 C2) という結果があった場合、(A1 B1 (C1 C2)) というように (A1 B1) に関して C の値をグルーピングする。

プロトタイプでは質問結果の (A B) 部分の一致するものを結果全て検索してから次の候補を検索する。しかし、これでは処理が $O(n^2)$ になるため、タプル数が増えると処理が破綻してしまう。そのため、(A B) をキーとし C を値とするハッシュテーブルに格納することで処理を O(n) とした。これにより、ボトルネックであった階層型リスト構造生成の処理時間は大幅に短縮された。

Java を利用する一番大きな利点は、JDBC を利用してのデータベースアクセスと Java のプラットフォーム非依存性である。Java 版では JDBC ドライバ、使用するデータベース名、ユーザ名を各ユーザの設定ファイルに記述することで、JDBC の利用できるデータベースであれば SuperSQL で利用することが可能である。PostgreSQL の他、Microsoft Access での実行も確認している。

4 比較・検討

Lisp 版、プロトタイプ Java 版、改良 Java 版それぞれに対し前述の質問文 1 を適用した。また、質問文 1 の条件を変化させて出力タプル数を増やし、それぞれの実行時間を測定した。測定環境は、以下の通りである。

- SUN ULTRA SPARC2 248MHz
- Solaris2.6
- JDK1.2
- PostgreSQL7.0

実験結果を図 3 に示す。横軸は出力タプル数、縦軸は実行時間 [s] である。

結果から、プロトタイプの Java 版が n の 2 乗のオーダーなのに対し、改良した Java 版は n のオーダーになっており、大幅に実行時間を短縮できたことが分かる。Lisp 版と改良した Java 版では、実行時間に 3 倍程度の差が存在する。これは Lisp 版は C によりネイティブコードに変換されていることによるものである。しかし、JIT や Hotspot などの実行時最適化機能を利用すれば Java でもネイティブコードの利用は可能であり、さらなる実行時間の短縮は考えられる。以上を考慮に入れた上で、改良 Java 版は Lisp 版には及ばないものの十分に実用に耐えるレベルに達していると考える。

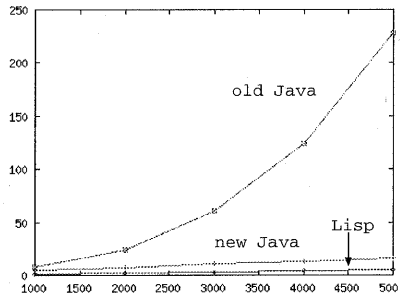

図 3: 実験結果

	Lisp	old Java	new Java
1000	1.3	7.7	4.7
2000	2.2	23.8	7.3
3000	3.0	60.9	10.6
4000	4.0	124.5	12.7
5000	4.7	227.8	15.8

表 1: 実行時間

5 まとめ

今回の実装で、Java による SuperSQL は実用可能なレベルに達したと言える。また、プラットフォームや使用する DBMS を選ばない点を考慮にいれると、処理速度は Lisp 版には及ばないが実用性は Java 版の方が高いと言える。

今後の課題としては、Java 版 SuperSQL はまだ Lisp 版で出力可能な媒体全てを出力できていないため、Lisp 版の機能の移行が必要である。

参考文献

[1] M. Toyama, SuperSQL: An Extended SQL for Database Publishing and Presentation, in *Proc. SIGMOD '98*, ACM(1998), pp.584-586.

[2] SuperSQL, http://www.db.ics.keio.ac.jp/ssql

[3] 遠山元道、軽部和幸、指田英雄、佐藤圭、実政宏幸、杉本晋司、瀬戸俊幸、中川裕二, レイアウト式 TFE の拡張, 情報処理学会データベースシステム研究会資料, 95-DBS-104(1995), pp.217-224.