SuperSQL を用いた HTML 生成における特定構造生成関数の実装

大山　直輝† 　五嶋　研人† 　遠山　元道†

† 慶應義塾大学理工学部情報工学科　〒 223–8522 神奈川県横浜市港北区日吉 3-14-1
E-mail: †{yama,goto}@db.ics.keio.ac.jp, ††toyama@ics.keio.ac.jp

あらまし　SuperSQL とは SQL の拡張言語であり，独自のクエリを記述して関係データベースの出力結果を構造化することで多彩なレイアウト表現の実現を可能とするものである．本論文では，SuperSQL を用いて関係データベースの出力結果から HTML を作成するにあたり，特定構造を生成することのできる関数機能を提案する．この関数機能は，プログラミングにおけるライブラリのように汎用性の高い特定の構造をひとまとまりとして簡単に利用できるようにしたものであり，先駆けとしてはクロスタブを生成する関数が実装されている．新たに本論文で提案する関数機能を利用することで，ユーザは jQuery を用いた動的なソートを行えるページや Google Charts を用いたグラフィカルなチャートが埋め込まれたページなどを SuperSQL を用いない場合に比べてはるかに少ないコード量で手軽に作成することが可能となる．

キーワード　データベース，SQL，SuperSQL，HTML

1　はじめに

SuperSQL は，独自のクエリ記述によって関係データベースの出力結果を構造化し，多彩なレイアウト表現を実現することのできる SQL の拡張言語である．関係データベースの値を用いた HTML を生成するような場合，SuperSQL は非常に便利なツールである．しかし，従来の SuperSQL で動的なソートを行えるページやグラフィカルなチャートを埋め込んだページを生成するには，SuperSQL で生成したページを手直ししたり，SuperSQL 以外のツールを併用することが必要であった．

本論文では，SuperSQL を用いて HTML を生成するにあたり，特定構造を生成することのできる関数機能の実装を提案する．これにより，SuperSQL の表現力および利便性が向上し，先述したような複雑なページを SuperSQL のみで生成することができるようになる．この機能を利用するとユーザは HTML 文書を直接記述せずに済むため，SuperSQL を用いない場合に比べてはるかに少ないコード量で複雑なページの生成が可能となる．また，HTML の記述に不慣れなユーザも手軽に複雑なページを生成できるようになる．

本論文の構成は以下の通りである．まず，第 2 章で SuperSQL の概要について述べる．次に，第 3 章で実装した特定構造生成関数について述べる．そして，第 4 章で特定構造生成関数の用例について述べる．第 5 章で評価について述べ，最後に第 6 章でまとめを記述する．

2　SuperSQL

本章では SuperSQL について簡単に述べる．SuperSQL のアーキテクチャを図 1 に示す．

SuperSQL は関係データベースの出力結果を構造化し，多様なレイアウト表現を可能とする SQL の拡張言語であり，慶應義塾大学遠山研究室で開発されている．[1] [2] そのクエリ

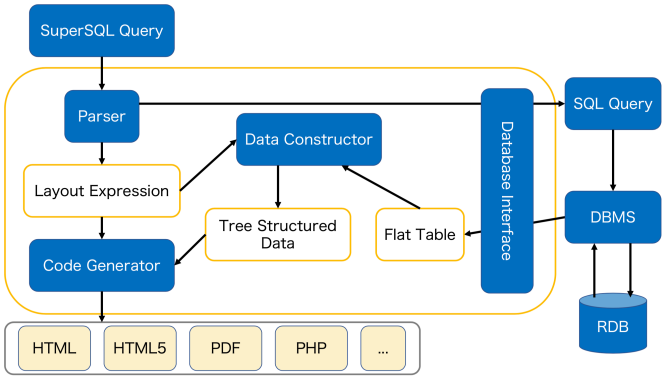


図 1　SuperSQL のアーキテクチャ

は SQL の SELECT 句を GENERATE `< media >< TFE >` の構文を持つ GENERATE 句で置き換えたものである．ここで `< media >` は出力媒体を示し，HTML，Mobile HTML5，PDF などの指定ができる．また `< TFE >` はターゲットリストの拡張である Target Form Expression を表し，結合子，反復子などのレイアウト指定演算子を持つ一種の式である．

2.1　結　合　子

結合子は取得したデータをどの方向 (次元) に結合するかを指定する演算子であり，以下の 3 種類がある．括弧内はクエリ中の演算子を示している．

- 水平結合子 (,)

データを横方向に結合して出力

例：Name, Score

Name	Score

- 垂直結合子 (!)

データを縦方向に結合して出力

例：Name! Score

- 深度結合子 (%)

データを 3 次元方向に結合して出力

例：Name % Score

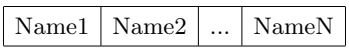
2.2 反 復 子

反復子は取得したデータをどの方向に繰り返し表示するかを指定する演算子である．括弧内はクエリ中の演算子を示している．

- 水平反復子 ([ ],)

データインスタンスがある限り，その属性のデータを横方向に繰り返し表示

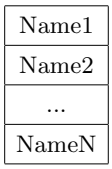
例：[Name],



- 垂直結合子 ([ ]!)

データインスタンスがある限り, その属性のデータを縦方向に繰り返し表示

例：[Name]!



2.3 装 飾 子

SuperSQL では，関係データベースから抽出された情報に文字のサイズや位置，横幅，縦幅などといった情報を付加することができる．これらの情報は属性名と装飾演算子 (@) を用いて指定し，

<属性名>@{<装飾指定>}

という形式で記述する．装飾指定は”装飾子の名称 =その内容”で指定する．また，”, ”を用いることで複数個の装飾を同時に行うことができる．

2.4 関 数

SuperSQL にはいくつかの関数が用意されている．image 関数では画像を表示することができたり，link 関数ではリンクを生成することができたりなど内容は様々である．近年ではクロス表を生成することのできる cross_tab 関数も実装されている．[3]

また，通常の関数とは異なる集約関数というものもある．これは，複数の入力行から 1 つの結果となる値を生成する関数である．例としては最大値を返す max，最小値を返す min, 平均値を返す avg などが挙げられる．集約関数では引数を丸括弧ではなく角括弧で囲う．

3 特定構造生成関数

本章では，本論文で実装を提案する特定構造生成関数について述べる．

SuperSQL では，クエリの検索結果を Data Constructor で構造情報に基づき構造化している．この構造化段階で通常の SSQL クエリの場合とは異なる構造化を行う関数を，本論文では特定構造生成関数と呼ぶ.

以下，特定構造生成関数の具体的な仕様について説明する．

3.1 sortable 関数

sortable 関数は，動的にソートを行うことのできるページを生成する関数である．本研究では，List.js という jQuery プラグインを取り入れることでこの sortable 関数を実装した．List.js は一般にテーブルのソートを行うような場合に用いられており，テーブルの頭の部分をクリックすることでページ遷移なしにソートをすることが可能である．この関数の一般的な記法を次に示す．

```
sortable(属性名 1, 属性名 2, ...)
```

sortable 関数では引数の個数に指定はなく，任意の個数の属性名を取ることができる．また，デフォルトではデータベース内の属性名がそのまま表頭部分の名前になる．ユーザが表頭部分の名前を指定したい時には次のように記述する．

```
sortable(属性名 1@{sortable-head=‘表頭 1’},
        属性名 2@{sortable-head=‘表頭 2’}, ...)
```

ここで，表頭 1 や表頭 2 の部分には任意の文字列を指定することができる．

以上が sortable 関数の基本的な使い方であるが，更に追加機能として，sortable 関数には動的に検索を行える機能も実装した．この機能を利用する時には次のように記述する．

```
sortable(属性名 1, 属性名 2, ...)@{search=‘on’}
```

このように記述することで，表の上に検索ウインドウを表示することができる．

3.2 gchart 関数

gchart 関数は，グラフィカルなチャートを埋め込んだページを生成する関数である．本研究では，Google Charts を取り入れることでこの gchart 関数を実装した．Google Charts とは Google がオンラインで提供するグラフ描画サービスであり，ユーザは JavaScript でデータとフォーマットを入力することで様々なチャートの画像を生成することができる．図 2 に Google

Charts を用いて生成したチャートの例を示す．Google Charts

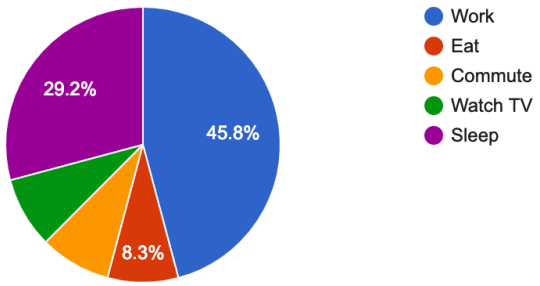


図 2　Google Charts を用いて生成したチャートの例

はこの例のようなチャートを約 30 種類描くことが可能である．実際にこれを利用した研究も行われており [5]，世界中で幅広く使われている．

gchart 関数の一般的な記法を次に示す．

gchart[属性名 1, 属性名 2, ...]@{gch-type=‘チャートの種類’}

gchart 関数ではチャートの種類によって引数の個数が変化する．例えば Pie Chart であれば引数は 2 個，Column Chart であれば引数は 2 個以上の任意の数といった具合である．チャートの種類は装飾子を用いて指定する．ここで指定できるチャートの種類名は Google Charts の提供するチャートの種類名に準じている．

また，Google Charts の機能としてチャートにオプションを指定することができる．指定可能なオプションはチャートの種類に応じて異なるため，ここでは Pie Chart を例に説明する．一般的な Pie Chart は図 2 のようなチャートである．しかし，オプションで 3D 指定をすると図 3 のようなチャートとなる．このオプション機能は gchart 関数でも使用可能である．Pie

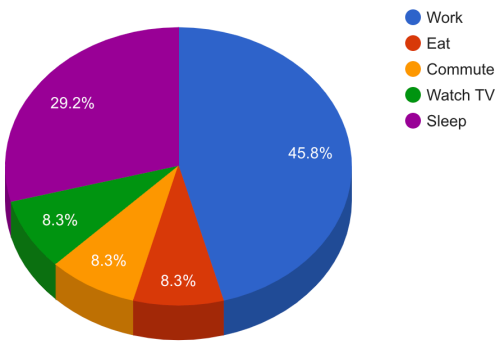


図 3　オプションで 3D 指定をした Pie Chart

Chart で先の例と同じオプションを使用する場合，次のように記述する．

gchart[属性名 1, 属性名 2, ...]@{gch-type=‘pie’, is3D=‘on’}

Pie Chart だけでも他に色の指定やしきい値の指定など数種類のオプションが存在し，これらも装飾子に記述することで設定することが可能である．他の種類のチャートに関しても，同様にオプションの設定を行うことができる．

4　用　　例

本章では特定構造生成関数の実際の用例を示す．用例では次に示すスキーマを用いる．

- prefecture(id, name, population, area, municipality)
- weather(year, month, prefecture, temperature, precipitation)

ここで，prefecture テーブルは 47 都道府県の id，都道府県名，人口，面積，市町村数といったデータを保有しており，weather テーブルは各都道府県の県庁所在地における毎月の気温及び降水量のデータを保有している．

4.1　sortable 関数

次に示す SSQL クエリを実行する．

クエリ 1

```
GENERATE HTML
  sortable(
    p.id@{width=50}, p.name@{width=150},
    p.population@{width=150}, p.area@{width=150},
    p.municipality@{width=150, sortable-head=‘市町村数’}
  )@{search=‘on’}
FROM prefecture p
```

このクエリでは，prefecture テーブルの全属性を sortable 関数の引数としている．属性 municipality の表頭部分の名前が市町村名となるように指定し，検索ウインドウも表示されるようにしている．クエリ 1 の出力結果は図 4 のようになる．

ここで，例として表頭の population の部分をクリックすると，図 5 のようにタプルが人口の昇順に並ぶようになる．もう一度 population をクリックするとタプルは人口の降順に並び変わる．同様に，表頭の id, name, population, area, 市町村名の部分をクリックするとタプルがそれぞれの昇順及び降順に並び変わるようになっている．

4.2　gchart 関数

次に示す SSQL クエリを実行する．

Search

id▼	name	population	area	市町村名
1	Hokkaido	5320000	8342384	179
2	Aomori	1278000	964564	40
3	Iwate	1255000	1527501	33
4	Miyagi	2323000	728222	35
5	Akita	996000	1163752	25
6	Yamagata	1102000	932315	35
7	Fukushima	1882000	1378390	59
8	Ibaraki	2892000	609719	44
9	Tochigi	1957000	640809	25
10	Gunma	1960000	636228	35
11	Saitama	7310000	379775	63

図 4　クエリ 1 の生成結果

Search

id	name	population▼	area	市町村名
31	Tottori	565000	350713	19
32	Shimane	685000	670826	19
39	Kouchi	714000	710386	34
36	Tokushima	743000	414680	24
18	Fukui	779000	419051	17
19	Yamanashi	823000	446527	27
41	Saga	824000	244068	20
30	Wakayama	945000	472464	30
37	Kagawa	967000	187677	17
5	Akita	996000	1163752	25
16	Toyama	1056000	424761	15

図 5　クエリ 1 の生成結果をソートしたもの

クエリ 2

```
GENERATE HTML
  [
    null((asc)p.id),
    p.name@{width=150, align='center'},
    gchart[w.month, w.temperature]
                        @{gch-type='column'}
  ],2!
FROM weather w, prefecture p
WHERE w.city=p.id AND w.year=2018
```

このクエリでは，都道府県名に 2018 年の各月ごとの降水量を Column Chart にしたものを横結合している．そして，それを prefecture テーブルの id の順に 2 つずつ横に表示したものを縦に繰り返し反復している．クエリ 2 の出力結果は図 6 のようになる．

5　評　　価

本章では，評価として様々な観点から提案システムと既存のシステムの比較を行う．

5.1　sortable 関数

テーブルや表のソートを行う場合，一般的には Microsoft Excel が多く用いられる．本節では，表現力及び利便性の観点から sortable 関数と Microsoft Excel の比較を行う．また，コード量の観点から従来の SuperSQL でのページ遷移によるソーティングとの比較を行う．

5.1.1　表　現　力

sortable 関数と Microsoft Excel を表現力の観点から比較する．

表 1　sortable 関数と Microsoft Excel の比較

機能	sortable 関数	Microsoft Excel
動的なソート	○	○
HTML の生成	○	△
データベースとの接続	○	×

表 1 に示した通り，sortable 関数と Microsoft Excel はどちらも動的なソートを行うことができる．しかし，HTML への出力という点では大きな違いがある．sortable 関数は SuperSQL 内の関数であるため HTML の出力が可能である．一方，Microsoft Excel では，HTML 内に表を記述するためには表を画像ファイルとして出力する必要がある．そのため，生成した HTML 内では表のソートを行うことができない．この点では sortable 関数の方が優れていると言える．また，データベースとの接続という点においては，Microsoft Excel ではデータベースから直接データを取ってくることはできないため，sortable 関数の方が優れていると言える．

5.1.2　利　便　性

次に，sortable 関数と Microsoft Excel を利便性の観点から比較する．Microsoft Excel では，ソートを行う時には正しく範囲の指定を行わないと一部の属性のみでソートしてしまい，正しい表が保てなくなってしまう場合がある．これに対して sortable 関数を利用して生成した表では，範囲の指定を行わずとも全ての属性を同時にソートするため，表が崩れるということはない．また，範囲の指定を行う必要がないため，昇順であれば 1 回のクリック，降順であれば 2 回のクリックでソートを行うことが可能である．ユーザが簡単に正しいデータを得ることができるという点において，提案手法は Microsoft Excel よりも優れていると言える．

5.1.3　コ　ー　ド　量

従来の SuperSQL では動的なソートを行うページを生成する

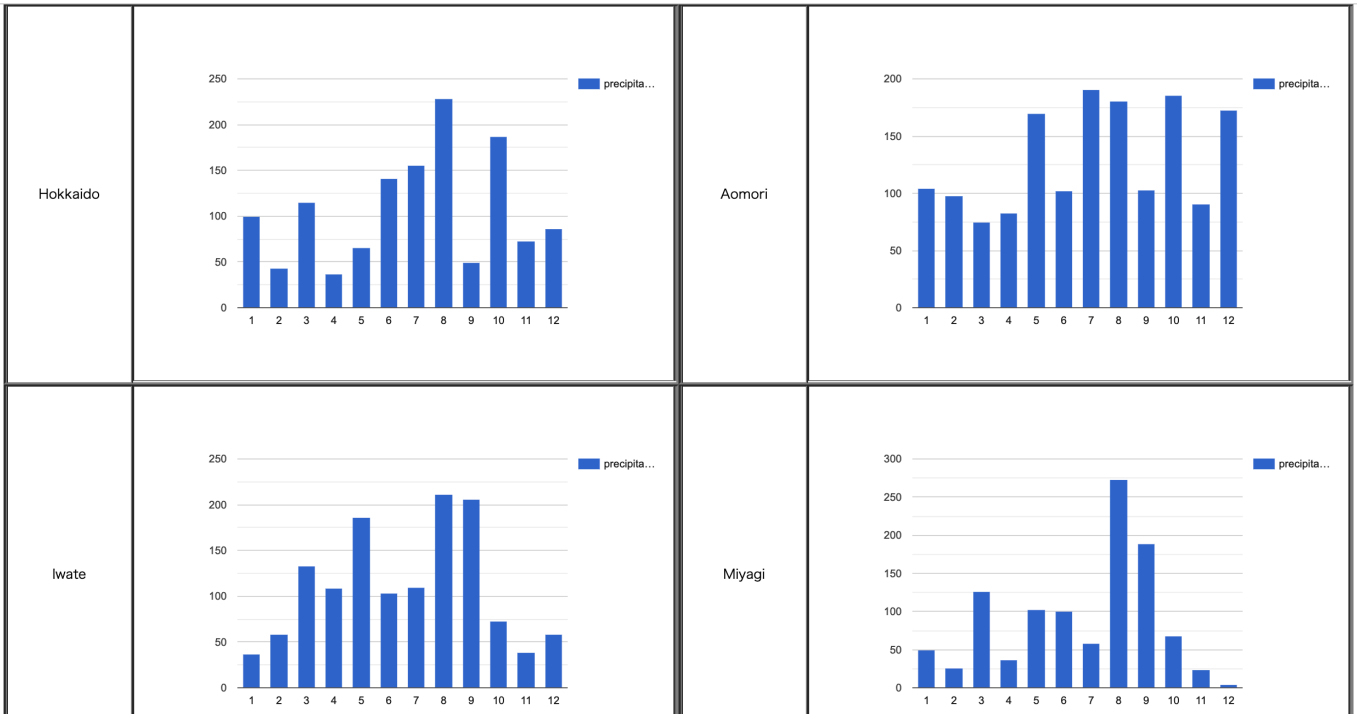


図 6　クエリ 2 の生成結果

ことはできなかったが，属性の昇順または降順にただ並べるということは可能であった．これとリンク生成機能を組み合わせることで，ページ遷移によるソートを行うことはできた．本項では，sortable 関数と従来のページ遷移によるソートをコード量の観点から比較する．

従来のページ遷移によるソートでは，テーブル上の全ての属性それぞれに対して昇順に並んだページと降順に並んだページを生成し，それらをリンクで結ぶこととなる．例として図**??**を生成することを考えると，それぞれのページを生成するのに必要な TFE のコード量は 5 行である．これを全ての属性において昇順と降順で 2 つずつ生成することを考えると，合計で TFE のコード量は 50 行となる．それに対し，sortable 関数を用いた手法では TFE の記述は全て合わせて 5 行で済んでいる．一般に，表示する属性数を n 個とすると，従来のページ遷移によるソーティングでは記述する TFE のコード量は $2n^2$ 行で表され，sortable 関数を用いた手法では n 行で表される．表示する属性数が増えるにつれ記述するコード量が増えるため，多くの属性を表示したい場合においては提案手法は非常に有用であると考えられる．

5.2　gchart 関数

本節では，gchart 関数と Google Charts 単体，及び過去に研究された SuperSQL と R のライブラリである ggplot2 を連携する手法 [4] の比較を行う．SuperSQL と ggplot2 を連携する手法により生成された HTML の例を図 7 に示す．

5.2.1　表　現　力

gchart 関数と Google Charts 単体及び SuperSQL と ggplot2 を連携したものを表現力の観点から比較する．

表 2　gchart 関数と各システムの比較

機能	gchart 関数	Google Charts	SSQL+ggplot2
HTML の生成	○	○	○
構造的な可視化	○	×	○
データベースとの接続	○	×	○
グラフの種類	○	○	△

表 2 に示した通り，これらの手法はどれも HTML を生成することができる．しかし，構造的な可視化が行えるかどうかという点で，Google Charts 単体に比べ他の手法は優れている．また，データベースとの接続という点においても，Google Charts 単体ではデータベースからデータを得ることはできないため，他の手法の方が優れていると言える．グラフの種類の多さという点では，ggplot2 に比べ Google Charts の方が多種類のグラフを描くことができるため，前 2 つの手法の方が優れていると言える．

5.2.2　利　便　性

次に，gchart 関数と Google Charts 単体及び SuperSQL と ggplot2 を連携したものを利便性の観点から比較する．Google Charts 単体では，HTML を作成するにあたりデータを手動で入力する必要がある．これに対して他の 2 種類の手法では，データベースからデータを得ることができるため，比較的容易に HTML を生成することができる．また，SuperSQL と ggplot2 を連携する手法では R を利用しているため，R 言語のパッケージである JRI を導入する必要がある．これに対して gchart 関数を使用する手法では，gchart 関数自体が SuperSQL の内部に備わっているため，新たに他のツールを使用することなく手軽にチャートを埋め込んだ HTML を生成可能である．また，

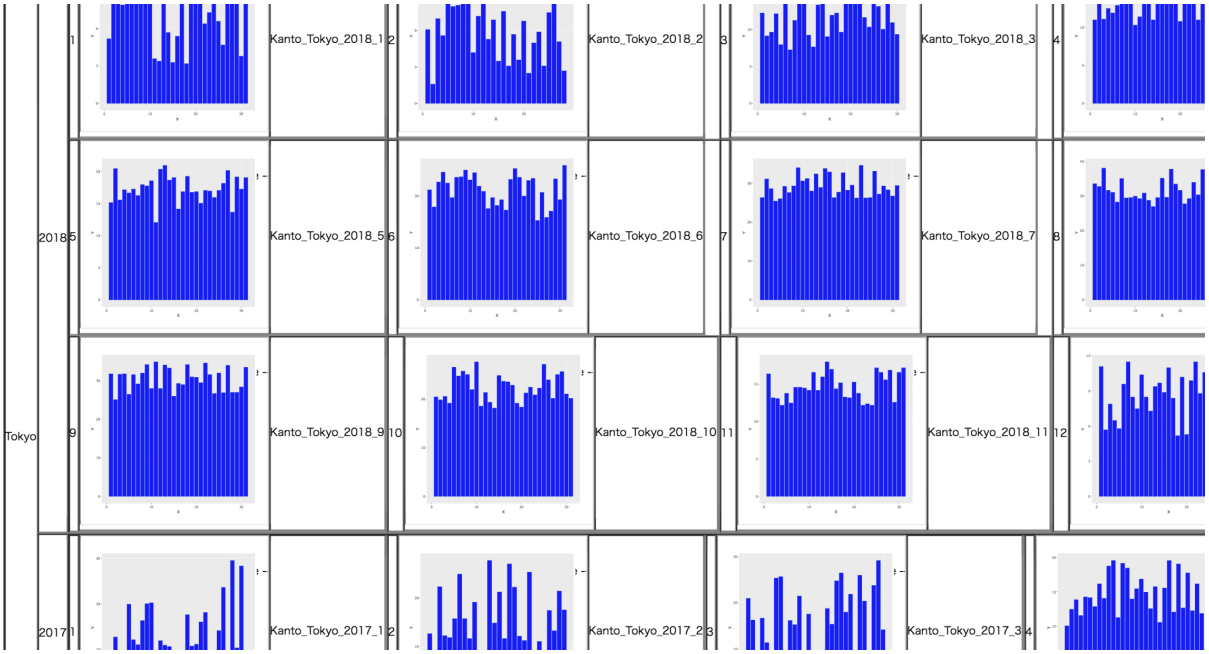


図 7　SuperSQL と ggplot2 を連携する手法により生成された HTML の例

Google Charts 単体ではそもそもツールを使用しないため，事前準備は一切必要ない．以上のことを考慮すると，チャートを埋め込んだ HTML を素早く生成したい場合には，提案手法が最も手軽な手法であると考えられる．

6　ま　と　め

6.1　結　　論

本研究では，SuperSQL を用いて HTML を生成するにあたり，特定構造生成関数を実装した．これにより，動的ソートが行えるページやグラフィカルなチャートが埋め込まれたページを SuperSQL のみで手軽に生成できるようになり，SuperSQL の表現力及び利便性が向上した．

6.2　課題・展望

本研究では sortable 関数と gchart 関数を実装したが，現段階ではどちらも HTML への出力にしか対応していない．他の出力媒体に対応していくことで，本研究がより有用なものになると考えられる．また，gchart 関数に関しては，現時点で Google Charts で利用可能な全てのオプションに対応している訳ではない．gchart 関数で利用できるオプションを増やしていくことは今後の課題となる．

他にも，特定構造生成関数としては sortable 関数と gchart 関数の他にも様々なものが考えられるため，更に特定構造生成関数の種類を増やしていくことで，表現力の更なる充実が期待される．

文　　献

[1] SuperSQL: http://ssql.db.ics.keio.ac.jp
[2] M. Toyama: “SuperSQL: An Extended SQL for Database Publishing and Presentation”. Proceedings of ACM SIGMOD’98 International Conference on Management of Data, pp. 584-586, 1998.
[3] A. Tabata, K. Goto, M. Toyama: “Generating Arbitrary Cross Table Layout in SuperSQL”. Proceedings of ACIIDS 2018: Intelligent Information and Database Systems, pp. 13-24, 2018.
[4] 大多和俊介, 五嶋研人, 遠山元道. SuperSQL による構造化データの二次元可視化. DEIM2019. 2019.
[5] Y. Sakamoto, S. Matsumoto, M. Nakamura: “Integrating Service Oriented MSR Framework and Google Chart Tools for Visualizing Software Evolution”. Proceedings of IWESEP 2012: International Workshop on Empirical Software Engineering in Practice, pp. 35-39, 2012.
[6] List.js: https://listjs.com/
[7] Google Charts: https://developers.google.com/chart