SQL Server Analysis Servicesのキューブ作成における集計パターンの最適化

Optimization of Aggregation Patterns in Cube Creation of SQL Server Analysis Services

高 橋 恭 之

要 約 SQL Server 2000がリリースされてから4年が経過し，Analysis Servicesを適用した多次元データ分析の事例も珍しくなくなってきた．また，大規模なデータを基にした多次元データ分析の構築事例も出てきている．

大規模なデータからキューブを構築する場合，キューブの処理時間とキューブの検索レスポンスを両立させることが課題となる．この課題解決の鍵は最適な集計パターンの作成であるが，現時点では作成の考え方，作成方法を記述した資料がほとんど存在しない．

本稿は飲料メーカA社の大規模データによる多次元データ分析システムの開発を通して得た，処理時間とレスポンスを両立させる有効な「集計パターン」の作成方法を記述する．

Abstract Four years have been passed since SQL Server 2000 was released, and the case example of the multi-dimensional data analysis using the Analysis Service has ceased to be novel recently. Several examples, such as a multi-dimensional data analysis based on a huge data volume, are recently reported.

In the case of building a cube from a huge data volume, it will be important issues for us to consider the following point of views : a cube processing time and a cube retrieving time. The creation of optimized aggregation patterns will be a key item for the solution of the problem, however there does not almost exist a documentation that shows any concept or method for the creation.

This paper describes the creation method of the optimal " aggregation patterns " available for achieving both an objective processing and an objective response time.

It has been acquired through the experience of the development for the multi-dimensional data analysis using the gigantic data volume at a certain beverage manufacturer.

1. はじめに

数年前までは多次元データ分析はハードウェア，ソフトウェアともに高価であったこともあり，会社内でもマーケティング部門などの限られたユーザを対象とした分析システムであった．しかし，近年ではハードウェアの性能向上と価格の下落，およびビジネス・インテリジェンス(BI)という情報戦略が浸透したことにより， 色々な会社からソフトウェアがリリースされて，多次元データ分析がポピュラーなデータ活用の形態となってきている．

SQL Serverは7.0で初めて多次元データ分析のエンジンであるOLAP Servicesをリリースし，SQL Server 2000（以下 SQL 2 K）では機能を拡張したAnalysis Services（以下 AS）としてリリースしている．

SQL 2 Kが2000年11月にリリースされてから4年が経過してASを適用した構築事例が増えてきているが，多次元キューブ（以下 キューブ）の設計の指針となるような書籍やホワイ

トペーパなどは限られている．

最近では数億件という大規模なデータからキューブを構築する要求も増えているが，データの規模が大きくなるにつれて検索のレスポンスとキューブを構築する時間の両立が難しくなる．検索のレスポンスを重視すると，色々な切り口に対応したより多くの集計値を事前に作成しておく必要があるが，逆に集計値を作成するためのバッチ処理の時間が増加してオンライン開始時間に間に合わなくなる．バッチ処理時間を重視すると有効な集計値を作成できず，レスポンスが悪化する．

このように，大規模なデータでキューブを構築する場合にはレスポンスと処理時間を両立させるための集計パターンの作成が不可欠となる．しかし，考え方，作成方法について記述されているものが，現在ほとんど存在していない．

本稿では，飲料メーカ A 社の開発を通して得た有効な「集計パターン」の作成方法について記述する．

2. Analysis Services の集計と作成方法

2.1 キューブ構造と集計

多次元分析を行うためにはキューブを構築する必要がある．キューブは大きくは分析の視点となるディメンションと金額や数量などの分析の対象となるメジャーで構成されている．メジャーはディメンションの階層構造に合わせて集計してあり，ユーザからの色々なディメンションを組み合わせた検索に応じて集計値を提供する（図 1）．

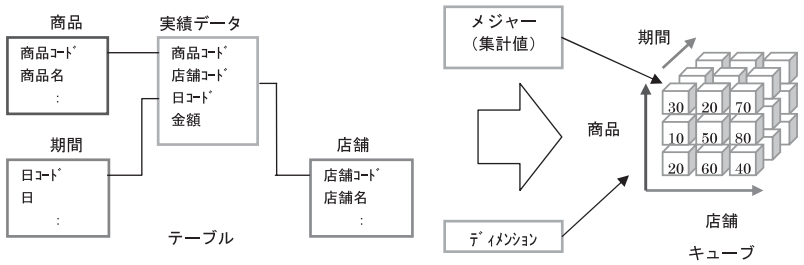

図 1　キューブ構成

AS はエンドユーザからの問い合わせに合致する集計値が存在すれば，その値を参照してユーザに結果を返す．しかし，合致する集計値が存在しない場合は実行時に集計を行い，ユーザに結果を返す．このため，事前に多くの集計値を作成しておけばレスポンスは向上することになる．

AS では集計値を作成するために事前に「集計パターン」を定義しておく．集計パターンはディメンションのレベルの組み合わせで定義を行い，キューブを処理する際にはこの定義に従って実績データを集計してキューブに格納する（図 2）．

2.2 集計の作成方法

AS ではディメンションとキューブの構成を定義した後に物理的なデータの格納方法を設定する「ストレージデザイン」をウィザードで行う．ストレージデザインは「データストレージ種類（MOLAP,ROLAP,HOLAP）の選択」と集計パターンの数を決定する「集計オプション

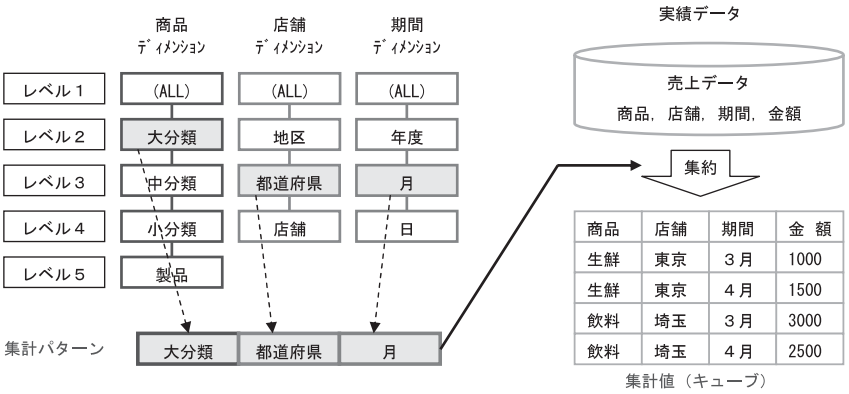

図 2　集計パターンと集計値

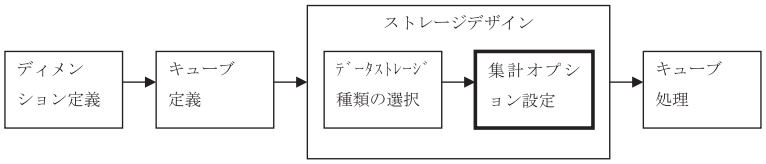

図 3　キューブ処理の流れ

の設定」を行う（図 3）.

集計オプションの設定ではウィザードを使用して,作成する集計パターンの数を決定する(図 4）. 集計パターン数を決定するためには二つの方法があり，どちらの方法を選択するかは任意である．実績データが多くないケースでは「パフォーマンスの到達率」を選択して，60% 以上を設定すれば，サーバのパフォーマンスで問題になることはほとんど無いと言える[1]．ただし，どちらの方式においても AS が任意の集計パターンを作成し，ユーザが組み合わせを指定することはできない.

① ストレージの到達見積もり

集計値を格納するファイルサイズを指定して，そのサイズに納まるように集計パターン数を計算する.

⇒集計パターン毎にディメンションのレベルのメンバ数を積算して集計値の数を見積もり，レコードサイズを積算してファイルのサイズを求める.

② パフォーマンスの到達率

全ディメンションのレベル数を積算した数（全集計パターン数）を 100% として，指定された到達率で集計パターン数を決定する.

⇒ただし，100% の集計パターン数が 136 個のキューブで，99% を指定すると 69 個となるため，割合だけで決定しているわけではない.

上記①，②のどちらの場合でも，グラフの下に表示されるキューブサイズは実際よりも大きな値となる．これは，集計パターン数の見積もりにおいてディメンションのメンバ数を基にしているのと，メンバ数を積算して求めた集計値が実際の実績データには存在しない組み合わせも含むからである．このため，グラフ下のキューブサイズはあくまでも目安と考えるべきである.

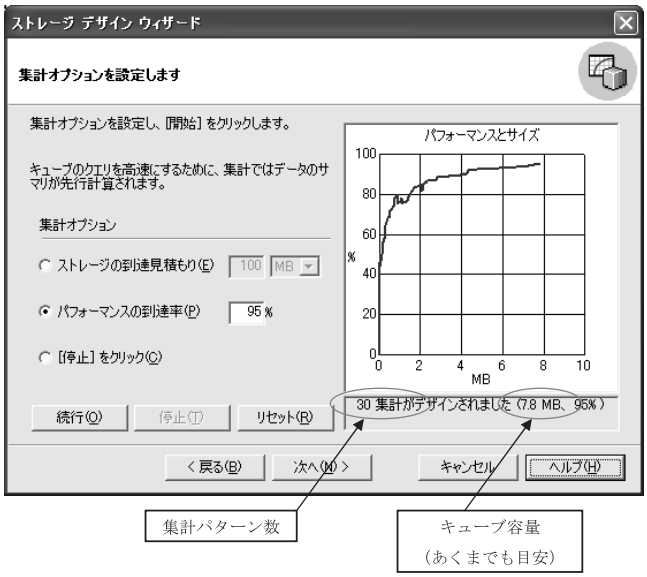

図 4　集計オプション設定画面

3.　Analysis Services が作成する集計パターンの問題点

AS の集計オプションの設定ではユーザがディメンションのレベルの組み合わせを設定できないことは前章で述べたが，これにより集計パターンの過不足が生じることになる．特に，AS の集計パターン作成のロジックの特長により，パラメータを意図的に変更しなければ集計オプションで作成できない集計パターンも存在する．

3.1　Analysis Services が作成しない集計パターン

AS が集計パターンを作成するロジックに「3 分の 1 ルール[2]」が存在する．これは，ディメンションのレベルのメンバ数を積算した値がキューブの基となる実績データのレコード件数の 3 分の 1 以下の場合に集計パターンとして作成するというルールである．

図 5 にあるように商品，店舗，期間の各ディメンションをモデルに説明すると，集計パターンである集計 1〜集計 4 までは括弧内のメンバ数を積算した値が売上データ 90 万件の 3 分の 1 以下である 30 万件以内であるため，集計オプションで集計パターンを作成できるが，集計 5 についてはメンバ数を積算した値が 30 万件を超過しているため，集計オプションで集計パターンを作成することはできない．

エンドユーザが月別，都道府県別，小分類別の集計値で分析を行う場合，集計パターンが作成できないことにより，レスポンスが悪化することが考えられる．

ただし，3 分の 1 を超えるようなディメンションのレベルの組み合わせは，下位のレベル（メンバ数が多い）同士の組み合わせで発生することが多く，図 5 のようにデータ件数が 90 万件と少ない場合には問題とはならない．また，上位レベルで分析するケースが多い場合にも同様のことが言える．

この 3 分の 1 ルールを適用している理由は標準機能の範囲で処理時間とレスポンスを両立させるためと考えられる．実績データの 3 分の 1 を超えるような集計値を作成するには CPU およびメモリなどのリソースを大量に消費して処理時間もかかる．これは SQL のグループ化処

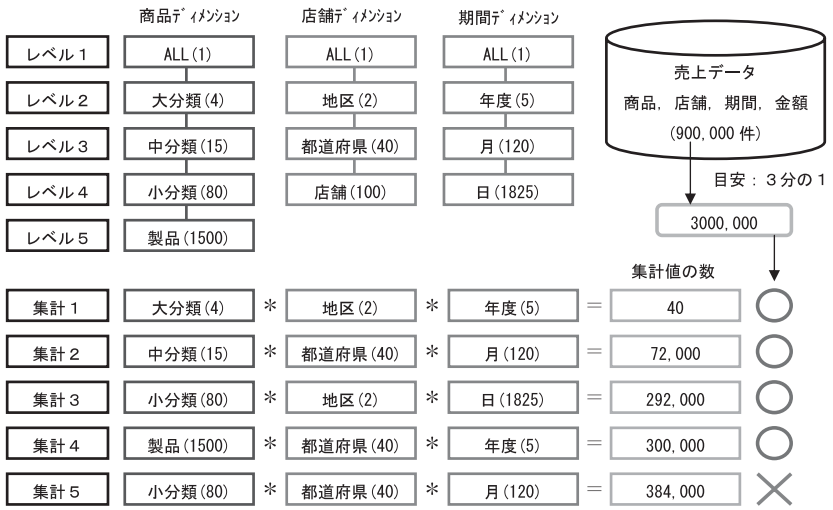

図５　集計の３分の１ルール

理と同様である．またレスポンスの観点からは，最初の検索は時間を要するが，以後はキャッシュに集計値が残ることで対応可能と考えたのではないかと思われる．

ただし，大規模なデータではキャッシュの効果は期待できない部分もあり，レスポンスを確保するために，処理に時間がかかっても集計値を作成しなければならないケースにおいては問題となる．

また，どうしても集計オプションで３分の１ルールに抵触する集計パターンを作成したい場合はキューブの Fact Table Size プロパティに意図的に大きな値を設定することで対応は可能である．しかし，目的以外の集計パターンも作成されるのとキューブの構成を変更した際に AS が Fact Table Size プロパティの値を書き換えるので注意が必要である．

3.2　Analysis Services が作成する不要な集計パターン

AS は任意にディメンションのレベルを組み合わせて，パフォーマンスを平均化する方向で集計パターンを作成するが，各ディメンションは全て同等として扱うためにユーザの検索ニーズに存在しない組み合わせの集計パターンも作成してしまう．

よくある例では図６のような「月ディメンション」と「週ディメンション」が存在するケースである．検索のニーズでは月か週いずれかで分析するため，二つのディメンションを同時に使用することは無い．しかし，AS の集計オプションでは両方にレベルを設定した集計パターンを作成してしまう．

このように必要に応じて使い分けを行うディメンションが多数存在する場合には，使われない不要な集計パターンが多く作成されてしまう．

4.　有効な集計パターンの作成の考え方

これまで述べてきたように，大規模なデータからキューブを構築する場合には AS が作成する集計パターンでは過不足が生じ，レスポンスと処理時間を両立させることが困難である．このため，有効な集計パターンを限定して作成する必要がある．まずは作成可能な集計パターンの数を特定することから始める．AS の集計オプションの設定で集計パターンを作成してキュ

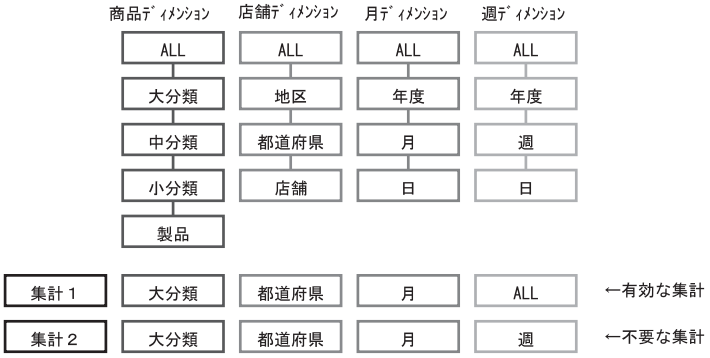

図 6　不要な集計パターン例

ーブを処理し，処理時間に余裕があれば，集計パターンの数を増やしていく．これで処理時間から設定可能な集計パターン数を割り出す．その後，不要な集計パターンの削除および有効な集計パターンの追加を行い，処理時間の要件から最適な集計パターンを導き出す．

この集計パターンでもパフォーマンスの要件を満たせない場合は CPU，メモリなどのリソースの追加を検討することになる．

4.1　集計パターンの作成ステップと使用するツール

最適な集計パターンは図 7 にあるように五つのステップを踏んで作成する．ステップ①は作成可能な集計パターン数を標準機能の集計オプションで見積もる．ステップ②ではパーティションマネージャというツールを使用してキューブにある集計パターンを CSV に Export する．そして，トップダウンの観点で最低限必要となる集計パターンを選択する．ステップ③ではユーザが検索で使用した集計パターンを AS のクエリログからプログラムを用いて CSV ファイルに抽出し，ステップ②の結果とマージさせる．ステップ④とステップ⑤では重複している集計パターンまたは連続したレベルの集計パターンを削除して，最終的に有効な集計パターンを作成する．

集計パターンを作成するためには下記の二つのツールを使用する．一つは SQL 2 K のリソースキットに含まれている，「PartitionManager.exe」と日本ユニシスで作成した「QlogToCsv.exe」である．パーティション・マネージャは既存の集計パターンの CSV 出力や作成した集計パターン（CSV）をインポートすることができる．クエリログ抽出はクエリログから検索で使用したディメンションとレベルの情報を抽出して集計パターンとして CSV に出力するプログラムである．

■PartitionManager.exe（パーティション・マネージャ）

機能は豊富にあるが，集計に関する機能としては集計パターンのエクスポート，インポート（CSV 形式）および編集，コピーなどがある．

■QlogToCsv.exe（クエリログ抽出）

このツールは AS が採取しているクエリログから参照しているディメンションとレベルの情報を抽出するプログラムである．飲料メーカ A 社をサポートした際にマイクロソフトのコンサルタントの方から情報を入手して日本ユニシスにて作成した．

以降に集計パターンの作成の各ステップに関する詳細な説明を行う．

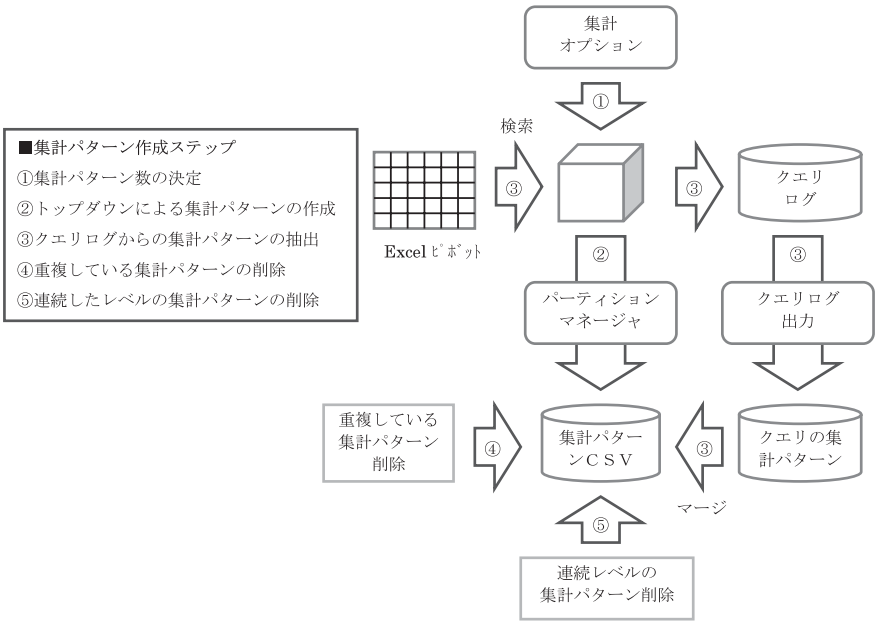

図 7　集計パターンの作成ステップ

4.2　集計パターン数の決定

作成可能な集計パターンの数はキューブの処理にかけられる時間と並行処理数で決まってしまう．4 時間で 1 個のキューブ（3 年分のデータを年月で 36 個のパーティションに分割して，各パーティションに同一の集計パターンが設定されている）を 6 並行で処理する場合，一つの処理系列で扱うパーティションの数は 6 個となる．それを 4 時間で処理するには 1 パーティション当たり処理時間は 40 分となる．つまり，集計オプションを使用して処理時間が 40 分に納まる集計パターン数を求める．

4.3　トップダウンによる集計パターンの作成

クライアントツールに Excel を使用する場合，目的の形にピボットテーブルを作成していく段階（ディメンションをピボットに配置して上位のレベルを表示させるケースなど）でレスポンスが悪くなることがある．これは，ユーザが最終的に見たいディメンションの配置だけを意識して，ディメンションの下位レベルを中心に集計パターンを作成した場合に発生する．

ディメンションは必ず上位レベルを経て下位のレベルに到達することになるため，ディメンションの上位レベルの集計パターンを作成しておかないと下位レベルに到達する前段階でレスポンスが悪くなることがある．新規にピボットで検索を行うケースを考慮して，単純な集計パターンを用意しておく必要がある．

作成の考え方としては図 8 のように，まず，全ディメンションの最上位レベルを指定した集計パターンを一つ作成する（①）．次に一つのディメンション（店舗）のみレベルを指定し，他のディメンションは全て最上位レベルを指定した集計パターンを作成する（②）．これを主要なディメンションに対して全て行う．最後にピボットテーブルの縦軸，横軸を想定して二つのディメンションでレベルを指定した（他のディメンションは最上位レベル）集計パターンも作成する（③）．

集計１	(All)	(All)	(All)	①
集計２	(All)	地区	(All)	②
集計３	(All)	都道府県	(All)	②
集計４	(All)	店舗	(All)	②
集計５	(All)	店舗	年度	③

図 8　トップダウンで作成する集計パターン例

4.4　クエリログからの集計パターンの抽出

クエリログから集計パターンを作成する目的は，ユーザの検索で必要となる集計パターンを限定して作成することである．ここで注意が必要なのは，AS は導入したままの状態ではクエリログを採取する設定になっていない点である．ログを採取するには分析マネージャからサーバのプロパティ画面を表示させ，「ログ記録」のタブを選択してログの設定を行う．このとき，ログの書き込み単位は必ず 1 クエリ単位とすること（図 9）．標準は 10 クエリであるが，この場合同一クエリが 10 回実行されなければログへの書き込みは行われず，ログの採取に時間がかかるためである．

クエリログを採取可能にした後でユーザが行う検索パターンを実行すると，ログには検索で使用したディメンションとレベルの情報が格納される．これをプログラム（QlogToCsv.exe）を使用して集計パターンとして CSV に抽出する．

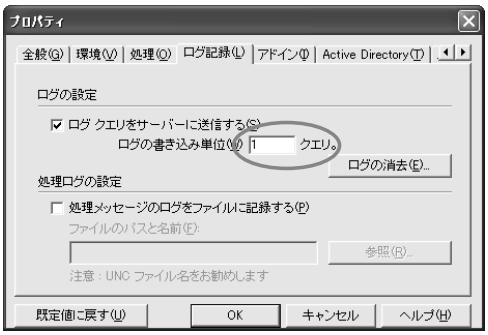

図 9　クエリログの採取設定画面

4.5　重複している集計パターンの削除

4.3 節と 4.4 節のステップを行うことにより，新規作成などにおけるパフォーマンスを確保するための集計パターンとユーザからの検索におけるパフォーマンスを確保するための集計パターンが作成できる．この二つをマージさせて基本的な集計パターンが作成できる．しかし，必ず重複した集計パターンが存在するため，重複を洗い出して一方を削除する必要がある．

4.6　連続したレベルの集計パターンを削除

4.5 節のステップまで実施して作成した集計パターンが，見積もった上限を下回っている場合には，集計をパーティション・マネージャでインポートして処理時間とレスポンスの確認を行う．しかし，上限を上回っている場合には，さらに集計パターンを削除する必要がある．

削除する対象はレベルが連続している集計パターンとなる．商品ディメンションを例にとる

と（図 10），「小分類」の利用頻度が高い場合は「小分類」の集計は残して，「製品」と「中分類」の集計を削除する．これは，「小分類」の集計があれば，「中分類」の値は「小分類」から作成できるため，「製品」から作成するよりも負荷が軽くなるからである．

当然，「小分類」から「製品」にドリルダウンを行う場合は遅くなるが，特定の「小分類」に絞ってドリルダウンさせる運用で回避することになる．

また，「大分類」はピボットに配置した際に最初に表示されるレベルであるため，集計パターンは残すこと．ピボット上で最初に表示されるレベルに集計パターンが存在しない場合，ピボットにディメンションを配置したときにレスポンスが悪化するためである．

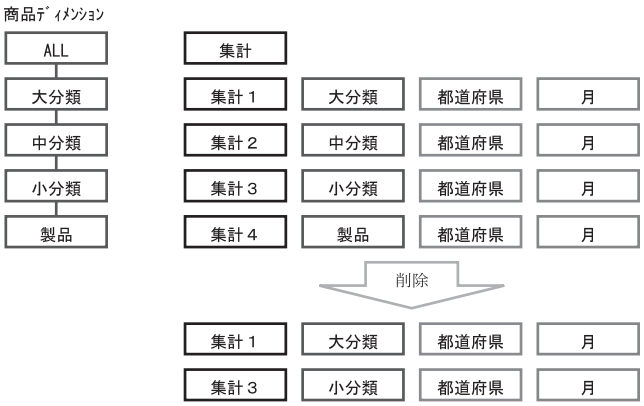

図 10　連続したレベルの集計パターンの削除

5. お わ り に

今回，飲料メーカ A 社の開発を通して大規模データでキューブを構築する場合の集計パターンの重要性を再認識した．小・中規模のデータを扱う限りでは標準機能の範囲で問題になることはあまりないが，大規模なデータを扱う場合には，レスポンスを確保するためのノウハウが必要となる．

2005 年中に SQL　Server 2005 がリリースされる予定であり，それに実装される新しい Analysis　Services で今回の題材である集計パターンに関する機能を再度確認することを検討している．

最後に，有効な集計パターン作成ステップを確立するために協力頂いた，飲料メーカ A 社の OLAP システム開発に携わったプロジェクトのメンバに感謝の意を表したい．

参考文献　［1］ Sanjay Soni,Wayne Kurtz WHITE PAPER Optimizing Cube Performance Using Microsoft Analysis Services 2000
［2］ Erik Thomsen,George Spofford,Dick Chase 著　トランスエディット/村井進　訳　ハラパン・メディアテック/宇野俊夫　監修「Microsoft OLAP ソリューション」

執筆者紹介　高　橋　恭　之（Mitsuyuki Takahashi）

1965 年生．1988 年中央大学商学部経営学科卒業．同年日本ユニシス(株)入社．　客先担当として汎用機の DBMS，運用管理ソフトウェアの適用/保守に従事．2000 年 4 月以降 SQL　Server を中心とした DWH の提案/構築支援を担当．現在は日本ユニシス・ソリューション(株)テクノロジコンサルティングサービス．NET ビジネス．NET テクノロジサービスに所属．