ＤＢ流通における更新差分データ抽出方式に関する検討

１Ｐ－７

奥村 昌和　岸本 義一　池田 哲夫

NTT 情報通信研究所

1　はじめに

NTTなど規模の大きな会社には、数多くのデータベース（ＤＢ）が存在する。これらのＤＢはそれぞれ個別の業務対応に別々に設計、運用されていることが多い。そのためにそれぞれのＤＢ上で重複するデータが存在し、データ投入稼働の無駄、投入ミス、投入遅れによるデータの不整合などの問題を生じている。この問題の解決のため、我々はＤＢ間でデータを自動流通させるＤＢ流通システムDB-STREAMを開発している[1]。

現状のDB-STREAMは、データ項目値やデータ構造の変換、統合、分配、集配信処理についてはその処理を簡易に記述することが可能であるが、提供側ＤＢからのデータの抽出処理、利用側ＤＢへのデータの格納処理についてはユーザによるＡＰの作成が必要である。全データを流通する場合は、ユーザがDBMSの提供するロード／アンロード機能を利用することで対処可能である。しかし、更新された差分データだけを流通する更新差分流通を行なうためには提供側ＤＢからの更新差分データの抽出、利用側ＤＢへの更新差分データの格納を行なうAPを業務処理に密接に関連して作成する必要があり、このことがＤＢ流通を適用する上での課題となっている。

本稿では特に上記の提供側ＤＢからの更新差分データの抽出処理の作成を簡易化する方式を検討する。

2　更新差分データ抽出処理の技術課題

2.1　更新差分データ抽出方式

更新差分データ抽出方式は、業務AP中に更新差分データ抽出処理を記述するＤＢ更新要求方式、ＤＢトリガ機能を用いるＤＢトリガ方式、更新ログファイルから更新差分データを抽出する更新ログ方式、ダンプファイルを比較しその変更点から更新差分データを抽出するダンプファイル方式の４つに分類できる。

この４方式の得失を表1に整理する。表1より、ＤＢ更新要求方式のみがＤＢを停止しない更新差分データの抽出、実システムへの適用性を満たし、解となり得ることがわかる。

そこで以下ではこのＤＢ更新要求方式の欠点である、更新差分データ抽出処理の実現時のユーザ負担を軽減する方式について検討する。

表1: 更新差分データ抽出方式の得失

更新差分データ抽出処理方式	抽出処理時のＤＢ停止	実システムへの適用性	処理実現時のユーザ負担	総合評価
ＤＢ更新要求方式	ＤＢ停止の必要なし	更新APは埋込SQLで記述	業務AP毎に処理記述	△
ＤＢトリガ方式	ＤＢ停止の必要なし	トリガは通常具備されていない	抽出対象テーブルの選択	×
更新ログ方式	ＤＢ停止の必要あり	ファイル構造は一般に非開示	DBMS毎に処理記述	×
ダンプファイル方式	ＤＢ停止の必要あり	ファイル構造は一般に非開示	DBMS毎に処理記述	×

2.2　ＤＢ更新要求方式

ＤＢ更新要求方式は図1に示す次の２つの処理で構成される。

（1）**更新差分データ分離処理** データの更新時に更新差分データを後から抽出可能な形式で分離する処理である。この処理は更新発生時に行なう必要があるため、個々の業務AP中の処理と連係して記述する。

（2）**流通データ作成処理** 更新差分データ分離処理で分離されたデータおよび関連する未更新のデータから、他システムへ流通させるデータを作成する処理である。

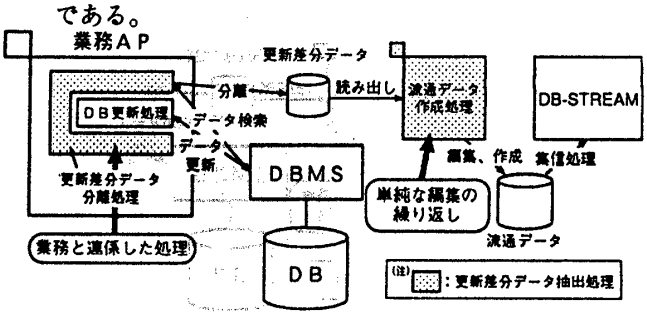

図1: 更新差分データ抽出処理

更新差分データ分離処理は、更新発生時に更新差分データを作成し、分離する必要があり、多くの記述が必要となる。またデータの更新業務と密接に関係しているために汎用的な処理方式が確立していない。一方、流通データ作成処理は、単純な編集処理を行なうだけである。そこで本稿では次の技術的課題について検討する。

技術的課題：業務の性質に依存せずに自動的に更新差分データ分離処理を作成する方式の確立

A Study of a Data Extract Method for a Data Delivery System

Masakazu OKUMURA, Giichi KISHIMOTO, Tetsuo IKEDA

NTT Information and Communication Systems Laboratories

3 更新差分データ分離処理生成方式

上記の技術的課題を解決するために、業務 AP 中に記述されている更新要求（SQL 文）を書き換え、更新差分データ分離処理を自動的に生成する方式を提案する。

3.1 更新要求 (SQL 文) の置換規則

（1） INSERT の置換規則（図 2）

（a）値を指定した INSERT の場合

INSERT する値（VALUES 句）から更新差分データを作成し、出力する処理を付与する。

（b）副問い合わせを用いた INSERT の場合

副問い合わせ部分をカーソルを用いて FETCH し、1 レコードづつ INSERT、および更新差分データの出力を行なように処理を置換する。

（2） UPDATE/DELETE の置換規則（図 3）

（a）探索型 UPDATE/DELETE の場合

更新発生前のキー値を取得するために、WHERE 句により選択されるレコードをカーソルを用いて FETCH し、1 レコードづつ更新する処理に置換する。この FETCH したデータと更新情報 (SET 句) より更新差分データを作成し出力する処理を付与する。

（b）位置付け UPDATE/DELETE の場合

この場合には、もともとカーソルを用いてレコードを FETCH し、1 レコードづつ更新する形になっている。しかし、FETCH する項目にキー値が含まれていない場合があり得るため、FETCH する項目にキー値が含まれるように処理を置換する。

3.2 ソースコンバータ方式

3.1節で述べた更新要求の置換規則に基づき、業務 AP のソースファイル中に記述されている更新要求およびその前後の処理を書き換えるソースコンバータを作成する。

このソースコンバータに、本来の業務処理のみを記述した業務 AP ソースファイルと、抽出対象テーブルのスキーマ定義を入力することで、更新差分データ分離処理を埋め込まれたソースファイルを自動的に生成する。処理の流れを図 4に示す。

4 まとめ

本稿では、更新差分データ抽出方式を 4 つの方式に分類整理し、ＤＢ更新要求方式が唯一解となり得ることを示した。また、そのＤＢ更新要求方式を実現する場合に、業務 AP 中に記述される、更新差分データ分離処理を自動的に生成するソースコンバータ方式を提案した。その結果、ユーザが更新差分データ分離処理を個別に業務 AP 中に記述する必要はなくなる。

今後は、今回提案したソースコンバータ方式の実システムへの適用性の検討および格納処理の実現方式の検討を行ない、DB-STREAM への抽出／格納機能の取り込みを目指す予定である。

値を指定したINSERT

```
INSERT INTO table VALUES( :v1, :v2 )
        ↓
INSERT INTO table VALUES( :v1, :v2 )
output(); /* 更新差分データの出力 */
```

副問い合わせを用いたINSERT

```
INSERT INTO table1 ( item1, item2 )
  SELECT data1, data2 FROM table2 WHERE data1>1
        ↓
DECLARE c CURSOR FOR SELECT data1, data2
                FROM table2 WHERE data1>1
OPEN c
LOOP:
 FETCH c INTO :v1, :v2
 INSERT INTO table1 VALUES( :v1, :v2 )
 output(); /* 更新差分データの出力 */
CLOSE c
```

図 2: INSERT の置換規則例

探索型UPDATE/DELETE

```
UPDATE table SET item1=5 WHERE item2>1
        ↓
DECLARE c CURSOR FOR SELECT * FROM table
                WHERE item2>1 FOR UPDATE OF item1
OPEN c
LOOP:
 FETCH c INTO :v1, :v2
 UPDATE table SET item1=5 WHERE CURRENT OF c
 output(); /* 更新差分データの出力 */
CLOSE c
```

位置付けUPDATE/DELETE

```
DECLARE c CURSOR FOR SELECT item1 FROM table
                WHERE item2>1 FOR UPDATE OF item1
OPEN c
LOOP:
 FETCH c INTO :v1
 UPDATE table SET item1=5 WHERE CURRENT OF c
        ↓
DECLARE c CURSOR FOR SELECT * FROM table
                WHERE item2>1 FOR UPDATE OF item1
OPEN c
LOOP:
 FETCH c INTO :v1, :v2
 UPDATE table SET item1=5 WHERE CURRENT OF c
 output(); /* 更新差分データの出力 */
CLOSE c
```

図 3: UPDATE/DELETE の置換規則例

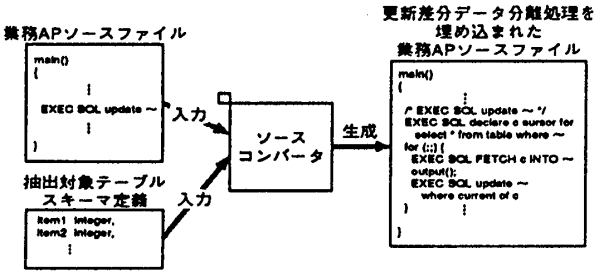

図 4: ソースコンバータ方式

参考文献

［1］ 池田他: "ＤＢ流通の基本方式について", 情報処理学会第 46 回全国大会,1993

［2］ 奥村他: "ＤＢ流通におけるデータ抽出方式に関する考察", 情報処理学会第 51 回全国大会,1995

［3］ C.J. デイト: "標準 SQL", 株式会社 トッパン, 1988