Technical Note 04-14
配列とセレクション

By Roland Lannuzel, 4D S.A.
Technical Note 04-14

（原題: Operations on Arrays and Selections）

概要

クエリの結果をセットに保存することは、初期の4Dから継承されている機能で、UNION、INTERSECTION、DIFFERENCE によってクエリを段階的に実行したり、結果を統合したりできる点が便利です。バージョン 6 から導入された命名セレクションは、セットによく似ていますが、両者には幾つかの相違点があります。

セット vs セレクション

セットは固定的なセレクションです。セットは、テーブルに存在するすべてのレコードにつき 1 ビットのメモリを占有します。たとえば、10 万件のレコードがあるテーブルの 50 レコードからなるセットのサイズは 12500 バイト（100000/8）です。

命名セレクションは、セレクションのレコードにつき 4 バイトを使用します。上記の例では、200 バイト（50*4）です。メモリのサイズだけを考えれば、セレクションがテーブル全レコードの 32 分の 1 を超えない限り、命名セレクションのほうが経済的だといえます。

反面、命名セレクションはセットのように UNION、INTERSECTION、DIFFERENCE といった操作ができません。今回はこの制限を克服する方法を考慮します。

前述のように、命名セレクションは、レコードごとに 4 バイトを消費します。この 4 バイトとは、要するにレコードの番号です。この番号は、Record number 関数で返される番号と同じで、GOTO RECORD コマンドに渡される番号でもあります。

それでは、以下のコマンドについて少し考慮してみましょう。

LONGINT ARRAY FROM SELECTION([Table];Array;’’Selection)
CREATE SELECTION FROM ARRAY([Table];Array;’’Selection)

LONGINT ARRAY FROM SELECTION は、カレントセレクション、あるいは COPY NAMED SELECTION で作成された命名セレクションから倍長整数の配列を作ります。

CREATE SELECTION FROM ARRAY は、ちょうど逆の動作をし、倍長整数の配列でカレントセレクションを変更するか、命名セレクションを作成/変更します。

LONGINT ARRAY FROM SELECTION で配列を作り、その配列を操作してから CREATE SELECTION FROM ARRAY で命名セレクションを作ると、元のセレクションとは異なるものができることになります。

これらのコマンドで扱う倍長整数は実際のレコード番号です。特定のレコードを直接指定するのでアクセスは高速ですが、無効な番号を使用するとエラーが発生します。

以上の原則を理解していれば、配列を使用してセレクションの操作をする方法が幾つか存在することが分かります。例を挙げて考えてみましょう。

クエリを実行した結果、3 レコードのセレクションができました。LONGINT ARRAY FROM SELECTION を実行するとレコード番号（たとえば 1000、2000, 3000）の配列が作られます。

別のクエリを実行した結果、今度は 2 レコードのセレクションができました。再び。LONGINT ARRAY FROM SELECTION を実行すると、レコード番号（たとえば 1500、1510）の配列が作られます。

両方の配列を合体させれば、その内容は 1000、1500、1510、2000, 3000 となり、CREATE SELECTION FROM ARRAY を実行すれば、セッレクションの UNION と同じ結果になります。今回の目標は、このように配列と命名セレクションを組み合わせて、さまざまな操作を実現することです。

配列の操作

配列の操作をするためのコマンドを用意しました。ポインタを使用しているので、ピクチャタイプ以外であれば、どのタイプの配列に対しても使用することができます。第 1 第 2 引数は元の配列、第 3 引数は結果を受け取る配列です。第 1 第 2 引数と同じ配列を指定することもでき、その場合、元の配列の内容が変更されることになります。任意の引数*は、0 番目の要素を処理に含めるかどうかを指定するものです。

```
4DARR_Union(->Array1; ->Array2; ->Array3;{“*”})
4DARR_Intersection(->Array1; ->Array2; ->Array3;{“*”})
4DARR_Difference(->Array1; ->Array2; ->Array3;{“*”})
```

コードの説明

```
4DARR_CheckParam($1;$2;$3)
$j:=4DARR_BuildArraysCheck($1;$2;$Start)
```

1 行目では、パラメータがすべてポインタであり、同じタイプのデータを指していることを確認しています。2 行目は、共通の要素を調べて、ふたつのブール配列 D4ARRaCheck1 および D4ARRaCheck2 を作成しています。それぞれ$1、$2 で示された配列と同じサイズであり、共通の要素に対応する要素が True になっています。

4DARR_BuildArrayCheck

注目したい箇所は次の 1 行です。

$Pos:=Find in array($FindInArray->;$LoopInArray->{$i};$Pos)

Find in array は、任意の第 3 引数を渡すことにより、検索の始点を設定することができます。Repeat/While ループの中でコマンドをコールしているので、この引数を使用して前回の検索が終了した箇所から検索を再開するようにしています。

4DARRaddLines ($DestinationArray;$NeededLines;$Start)

必要に応じて配列に要素を追加するためのメソッドです。

注記:
バージョン 2004 では APPEND TO ARRAY コマンドが追加されました。

4DARR_Union

はじめに$3 が$2、$1 のいずれかと同じであるかを調べます。調べた結果に基づき、コマンドはより小さいほうの配列をループの対象に選択します。次に$3 の配列のサイズを大きくします。（余った要素は最後に削除されます。）ここからが実際の UNION 操作です。_4DARRaCheck1 あるいは_4DARRaCheck2 のうち、ループの対象に選択されたほうの配列を使用し、False に対応する要素を元の配列から結果の配列に追加します。

配列の 0 番目の要素を削除するためにちょっとした技を使用しています。ポインタを使用しているため、pArray->{0}:=0 あるいは pArray->{0}:=””のようなタイプに特有のシンタックスは使用できません。そこで、位置 0 に要素を追加し、次いで要素 1（元の要素 0）を削除しています。

UNION 以外のメソッドも、基本的に似た処理によって集合の操作を実現しています。

セレクションの操作

配列の操作メソッドが完成したら、今度はセレクション用のメソッドを作ってみましょう。原理が理解できれば、セットコマンドと同等のものばかりか、セットコマンドにはないもの、たとえば exclusion のようなものを作ることが可能です。

シンタックス

基本的にセットコマンドのものを踏襲します。つまり UNION (“SET1”; “SET2”; “SET3”) に対応して 4DSEL_UNION (ptrTable; “Selection1”; “Selection2”; “Selection3”)となるようにします。セットと異なり、セレクションの属するテーブルへのポインタも引数として渡します。4DSEL_UNION メソッドは、引数のタイプを確認したのち、操作の種類を引数にして 4DSEL__Build メソッドをコールしています。

4DSEL__Build ($1;$2;$3;$4;"Union")

4DSEL__Build は、セレクションから倍長整数の配列を作り、引数で渡された操作をし、結果セレクションを作っています。メソッドには、注目したい次のような 1 行が含まれています。

4DARR_DistinctValues (->_4DaRecNum1;->_4DaRecNum1)

4DARR_DistinctValues メソッドは、元の配列と結果の配列が同じで、重複する値を除外して更新するものです、UNION、INTERSECTION、DIFFERENCE は重複する値を返すはずがないので、一見、不要なコードですが、念のために追加しました。

このようにした理由は、GOTO RECORD コマンドと同じく CREATE SELECTION FROM ARRAY も細心の注意を要するローレベルコマンドであるためです。速度を優先するため、両コマンドともレコード番号の妥当性をチェックせずに動作しています。

まとめ

命名セレクションを使用する最大の理由は、スピード向上よりもメモリ空間の節約にあります。数件の結果しか返さないクエリをセットで操作した場合、膨大なレコードが存在すればメモリが足りなくなる恐れがあります。その点、この Tech Note の手法は非常に経済的です。

サンプルデータベース

はじめに画面上部のボタンを 4 つともクリックして命名セレクション ABCD を策します。それぞれのボタンをクリックして A∩B、B∪C、(A∪D)-(B∪C)の結果が確認できます。