Technical Note 04-13
階層リストのインポート

By Olivier Deschanels, 4D S.A.
Technical Note 04-13

（原題: Importing a Hierarchical List）

概要

構造化されたデータは、階層リストにして管理すると便利です。今回は、そのようなデータを、階層リストに変換するような汎用メソッドについて考えてみたいと思います。

目標は、下記のデータソースを、右のような階層リストに整理することです。

Champ1	Champ2	Champ3	Champ4
Outdoors	Sport	Bike	
Outdoors	Sport	Rollers	
Outdoors	Sport	Skateboard	
Outdoors	Camping	Tent	
Outdoors	Camping	Sleeping bag	
Clothing	Sport	Sport suit	Woman
Clothing	Sport	Sport suit	Man
Clothing	Sport	Sport suit	Kid
Clothing	Casual	Suit	
Clothing	Motocross	Jersey	
Clothing	Motocross	Boots	
Clothing	Motocross	Knee protector	

基本的な考え方

階層リストには以下のような特徴があります。

要素の数は一定ではない
階層の深さは一定ではない
親要素が異なる同名の子要素がある
空の要素は存在しない

これらの点を踏まえて、少しずつコードを作成してゆきましょう。

階層リストをテストする

階層リストでいろいろ試してみたいときには、SAVE LIST コマンドを使ってリストを保存してしまえば、デザインモードで結果を確認することができるので便利です。

```
C_LONGINT($List)
$Liste:=LH_Import
SAVE LIST($List;"LH_Import")
```

再帰メソッドを作成する

階層リストを作成する過程では、ある条件が満たされるまでの不特定回数、同じ処理を繰り返すことになります。このようなコードには、再帰メソッド、つまり自らをコールするメソッドが向いています。はじめはパラメータなしでコールし、初期化ルーチンを経た後、パラメータを渡して自らをコールすることによって、次から別ルーチンへ処理が流れるようにするのが基本です。

```
Case of
¥ (Count parameters=0)
	`initializing  method
Else
	`processing  recursive calls
End case
```

階層リストを作成する場合、最終的にリスト参照を結果として返すメソッドを作成します。

作成にあたっては、要素ごとに次の階層を解析し、値がユニークなデータの集まりごとに要素を追加してゆきます。値がユニークなデータを調べるためには、インポート用のテーブルを用意して、そこにデータを読み込んでから DISTINCT VALUES コマンドを使用します。

再帰ルーチンは、階層の数だけ繰り返しコールされますから、自らをコールするときのパラメータには、階層の数を渡すのがもっとも自然です。各要素にはユニークな要素参照をつけますが、ローカル変数では再帰しながらカウントを続けることができないので、プロセス変数を使用します。

```
C_LONGINT($0)  `hierarchical  list  obtained

Case of
¥ (Count parameters=0)   `Initializing  method
	lh_counter:=0
	$0:=LH_Import (0)
Else   `processing  recursive calls
	ALL RECORDS([IMPORT])
	DISTINCT VALUES([IMPORT]Field1;$_value)
	$list:=New list
	For ($i;1;Size of array($_value);1)
		lh_counter:=lh_counter+1
```

```
            APPEND TO LIST($list;$_value{$i};lh_counter)
        End for
        $0:=$list
End case
```

とりあえず、最上層フィールドのユニークな値に基づいて親要素が作成できました。

第二階層以降も同じ再帰ルーチンをコールすることになりますが、最上層のときとは若干、異なる箇所があります。

最上層では ALL RECORDS→DISTINCT　VALUES
次の階層では QUERY→DISTINCT　VALUES

QUERY を実行するためには、条件となる値、つまり現在処理中の階層の値が必要です。この値を、再帰するときに第２引数として渡すようにコードを書き換えましょう。

```
C_LONGINT($0) `Hierarchical list obtained
C_INTEGER($1) `Level in Recursivity
C_STRING(80;$2) `value

Case of
¥ (Count parameters=0) `Initializing method
        lh_counter:=0
        $0:=LH_Import (0;"")
Else `processing recursive calls
        If ($1=0)
                ALL RECORDS([IMPORT])
                DISTINCT　VALUES([IMPORT]Field1;$_value)
        Else
                QUERY([IMPORT];[IMPORT]Field1=$2)
                DISTINCT　VALUES([IMPORT]Field2;$_value)
        End if
        $list:=New list
        For ($i;1;Size of array($_value);1)
                If ($1=0)
                        $sublist:=LH_Import (1;$_value{$i})
                End if
                lh_counter:=lh_counter+1
                APPEND TO LIST($list;$_value{$i};lh_counter;$sublist;True)
        End for
        $0:=$list
End case
```

これに DISTINCT　VALUES が 0 を返す場合の処理を加えます。

```
C_LONGINT($0) `Hierarchical list obtained
C_INTEGER($1) `Level in Recursivity

Case of
¥ (Count parameters=0) `Initializing method
        lh_counter:=0
        $0:=LH_Import (0;"")
```

```
Else
	`processing recursive calls
	If ($1=0)
		ALL RECORDS([IMPORT])
		DISTINCT   VALUES([IMPORT]Field1;$_value)
	Else
		QUERY([IMPORT];[IMPORT]Field1=$2)
		DISTINCT   VALUES([IMPORT]Field2;$_value)
	End if
	If (Size of array($_value)>0)
		$list:=New list
		For ($i;1;Size of array($_value);1)
			If ($1=0)
				$sub_list:=LH_Import (1;$_value{$i})
			End if
			lh_counter:=lh_counter+1
			APPEND TO LIST($list;$_value{$i};lh_counter;$sub_list;True)
		End for
	Else
		$list:=0
	End if
	$0:=$list
End case
```

ここまでの処理をハードコーディングしてきましたが、ここからは汎用的なコードを目指して完成させてゆきましょう。

汎用的なコードを作成する

DISTINCT VALUES は階層つまりフィールドごとに実行するので、何度も繰り返すことができるようにフィールドをポインタで渡すようにコードを書き換えます。また、ひとつ上の階層の要素を指すのにもポインタが利用できます。

これにより、不特定回数の再帰に耐えられるルーチンになります。あとは、繰り返し過ぎて存在しないフィールドを指定してしまわないように、ストップポイントを設け、空の要素を作らないようにコードを修正しましょう。

```
C_LONGINT($0)  `Hierarchical  list  obtained
C_INTEGER($1)  `Level  in recursive calls

Case of
¥ (Count parameters=0) `Initializing method
	lh_counter:=0
	$0:=LH_Import (0;"")
Else
	`processing recursive calls
	$Processing_Level:=$1+1
	If ($Processing_Level<=Count fields(Table(->[IMPORT])))
	   If ($Processing_Level=1)
	   ALL RECORDS([IMPORT])
	Else
	   QUERY([IMPORT];Field(Table(->[IMPORT]);$Processing_Level-1)->=$2)
```

```
        End if
        DISTINCT  VALUES(Field(Table(->[IMPORT]);$Processing_Level)->;$_value)
        If (Size of array($_value)>0)
           $list:=New list
           For ($i;1;Size of array($_value);1)
                If ($_value{$i}#"")
                     $sub_list:=LH_Import ($Processing_Level;$_value{$i})
                     lh_counter:=lh_counter+1
                     APPEND TO LIST($list;$_value{$i};lh_counter;$sub_list;True)
                End if
           End for
        Else
           $list:=0
        End if
        Else
           $list:=0
        End if
        $0:=$list
End case
```

スタックを利用する

以上のコードを実行すると、階層リストの一部に重複する要素が生まれてしまいます。これは要素を絞り込むコードが上の階層をみていないため、つまり Outdoors/Sport と Clothing/Sport を区別していないために起きています。

階層が浅いときは問題になりませんでしたが、実際には QUERY→DISTINCT VALUES という絞り込みではなく、QUERY→QUERY→DISTINCT VALUES というように、階層の深さだけ QUERY を繰り返して絞り込みをしなくてはいけませんでした。

不特定回数の QUERY を実行するためには、階層の深さ分のサイズを持った配列に必要な値を納めたもの（=スタック）を使用します。

```
C_LONGINT($0)  `Hierarchical  list  obtained
C_INTEGER($1)  `Level  in recursive calls

Case of
¥ (Count parameters=0) `Initializing method
      lh_counter:=0
      ARRAY  STRING(80;_champ;Count fields(Table(->[IMPORT])))
      $0:=LH_Import (0;"")
Else
      `processing recursive calls
      $Processing_level:=$1+1
If ($Processing_level<=Count fields(Table(->[IMPORT])))
      If ($Processing_level=1)
           ALL RECORDS([IMPORT])
      Else
      For ($j;1;$1-1)
           QUERY([IMPORT];Field(Table(->[IMPORT]);$j)->=_champ{$j};*)
      End for
      QUERY([IMPORT];Field(Table(->[IMPORT]);$Processing_level-1)->=$2)
```

```
End if
DISTINCT   VALUES(Field(Table(->[IMPORT]);$Processing_level)->;$_value)
If (Size of array($_value)>0)
	$list:=New list
	For ($i;1;Size of array($_value);1)
		If ($_value{$i}#"") _champ{$Processing_level}:=$_value{$i}
			$sub_list:=LH_Import ($Processing_level;$_value{$i})
			lh_counter:=lh_counter+1
			APPEND TO LIST($list;$_value{$i};lh_counter;$sub_list;True)
		End if
	End for
	Else
		$list:=0
	End if
	Else
		$list:=0
	End if
	$0:=$list
End case
```