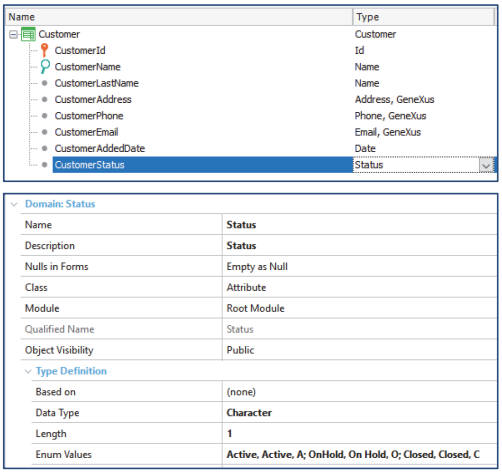
データセレクター

GeneXus™

GeneXus

データセレクター

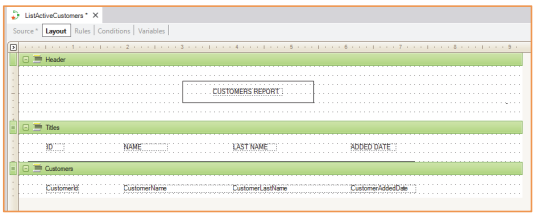


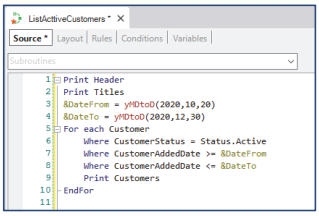
旅行代理店向けアプリケーションの Customer トランザクションに、旅行代理店のシステムにおける顧客のステータス (アクティブ、保留、クローズのいずれか) を表す CustomerStatus という項目属性があるとします。このために、新しい列挙型データタイプが定義されています。

このアプリケーションは、いくつかの場所で、特定の期間 (2 つの日付の間) に入力されたアクティブな顧客を対象に処理を行う必要があるとします。
たとえば次のような処理です。

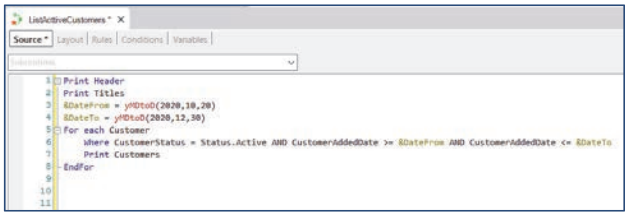
日付の範囲を受け取り、その範囲内にシステムに入力されたアクティブな顧客を示す PDF リストを生成します。

データセレクター





≈



これを実現するために、ListActiveCustomers というプロシージャーを作成します。

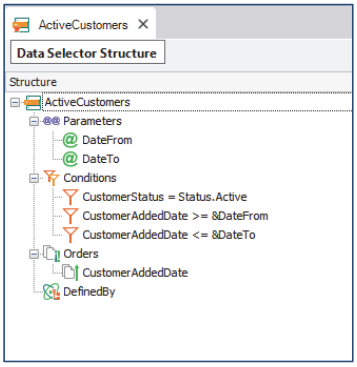
ここまでに分かっていることでクエリを実装するとしたら、3 つの条件 (ステータスがアクティブ、開始日、終了日) に従うことになります。

これは複数の場所で使用されるため、そのたびに繰り返し定義しなくても済むように、1 つの場所で定義して、それに名前を付け、その名前を参照として使用します。
その場所が Data Selector オブジェクトです。

GeneXus

データセレクター

構造:



「ActiveCustomers」というデータセレクターを作成し、必要な条件をここで定義します。
オブジェクトの構造を見てみましょう。

[Parameters] ではデータセレクターが受け取るパラメーターを定義します。これが [Conditions] で使用されます。この例には、&DateFrom と &DateTo という 2 つの変数があります。

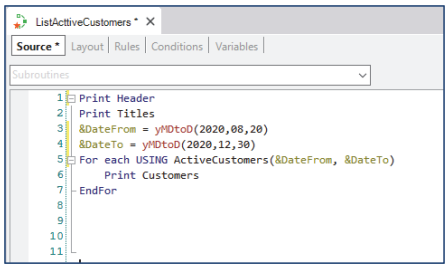
[Conditions] には、取得するデータをフィルタするために満たす必要がある条件を入力します。この場合は、顧客のステータスがアクティブであることを指定します。また、顧客の入力日が DateFrom 変数の値以上で、DateTo 変数の値以下である必要があります。

[Orders] では、取得したデータを受け取る順序を指定できます。この場合は CustomerAddedDate を入力して、データを日付で並べ替えます。

最後の [Defined By] では、最終的なベーステーブルを定義するための項目属性または項目属性のリストを入力できます。これは、ベーステーブルを判断する際の曖昧さを解消するのに役立ちます。ここでは空のままにします。

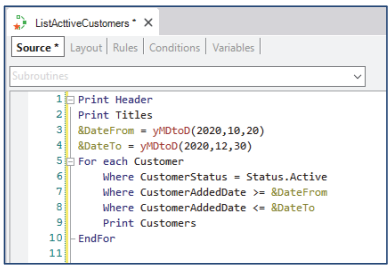
USING 節

データセレクターあり

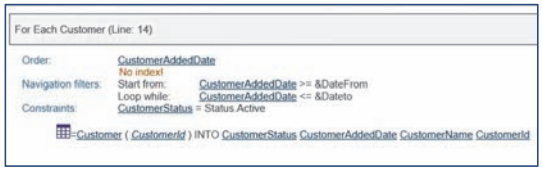


≈

データセレクターなし



ナビゲーション表示



先ほど説明したとおり 1 か所で定義することにより、このクエリを必要とするすべての場所で再利用でき、メンテナンスも容易になります (定義を変更する必要が生じた場合、1 か所で変更するだけで、このナレッジベースが使用されているすべての場所に自動的に変更が適用されます)。

データセレクターは、受け取ったパラメーターに基づいて、データに対する一連の条件や順序を一元的に指定します。これにより、Order、Where、Defined by の各節を必要とする場所でこれらを繰り返し記述する必要がなくなります。

データセレクターを作成した後、顧客をリストするプロシージャーからそれをどのように使用するかを見てみましょう。

これには USING 節を使用します。図に、この例での使用方法が示されています。
For Each コマンド内で USING 節を使って、ActiveCustomers データセレクターにパラメーターを送り、データのフィルタに使用する日付に対応する 2 つの変数を渡します。
この動作は、前のケースと同じようになります。

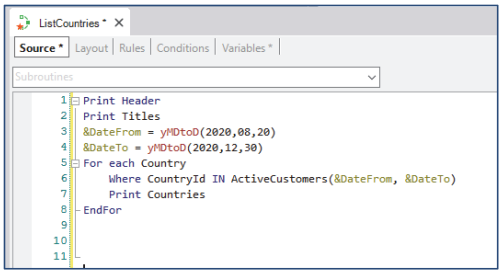
ベーステーブルの判断や、ナビゲーションの解決についても、データセレクターを使用せずにその節を直接記述した場合と同じようになります。
データセレクターからのものに加えて、Order 節や Where 節を For each に追加することは可能でしょうか。可能です。節が追加されます。

ナビゲーション表示を見ると、Customer テーブルが参照され、Customer AddedDate を基準に並べ替えられます。レコードは、Conditions 内で適用したフィルタに従って参照され、ステータスがアクティブの顧客のみがフィルタされます。

IN 演算子

PDF レポート:

ListActiveCustomers (ソース)



ナビゲーション表示

たとえば、顧客の国を追加し、指定された 2 つの日付の間にシステムに入力されたアクティブな顧客の国をリストするとします。データセレクターを使ってこれを行うには、次のように処理します。

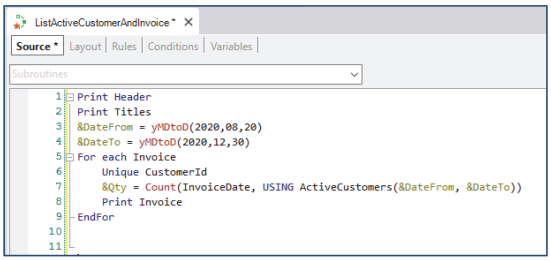
For each Country 内で、
Where CountryId IN ActiveCustomers と記述し、パラメーターで &DateFrom および &DateTo 変数を渡します。
この場合、USING 節ではなく IN 演算子を使用します。
こうすることで、データセレクターを独立したデータベースクエリであるかのように使用します。

ここには 2 つのクエリがあります。1 つはデータセレクターのクエリで、指定された 2 つの日付の間に入力されたアクティブな顧客のセットと、それぞれに対応する国が返されます。もう 1 つは For Each コマンドに対応するもので、そのセットに含まれる国をフィルタします。

ナビゲーション表示を見ると、Country テーブル全体が参照され、CountryId によって並べ替えられます。また、制約事項 (Constraints) として、For each の Where 節内でデータセレクターの顧客リスト内にある国のみをフィルタすると宣言しています。

GeneXus

式でのデータセレクターの使用



これは、データセレクターの 3 つ目の使用例です。
すべての顧客を、請求書とその金額とともに PDF に表示する必要があり、日付の範囲を入力して、その範囲内にシステムに入力されたアクティブな顧客の請求書のみをカウントするとします。アクティブではない顧客や、指定した日付範囲以外に入力された顧客は、リストには含まれますが、請求書の数は 0 と表示されます。

ここでは式、つまり Count 式の中でデータセレクターを使用します。
Aggregate 式の 2 番目のパラメーターは、レコードが「集計される」ために満たす必要がある条件を記述するためのものです。

詳しく見ると、For each 内で Invoice テーブルを参照しています。各顧客に N 個の請求書がある可能性があるため、Unique 節を記述して、該当する顧客に請求書のうちの 1 つを保持するようにします。そのため、同じテーブルに対して Count 式を適用すると、データセレクターの節にも適合する、その顧客の請求書をカウントします。つまり、顧客がアクティブではない場合、Count 式の結果は 0 になります。これは、その請求書のすべてに対して、該当する顧客がアクティブな顧客にならないためです。顧客がアクティブな場合は、対象となる範囲内の請求書のみがカウントされます。

GeneXus

For each でデータセレクターを使用する方法

USING 節　　≈　　For each 内に Where 節を追加

For each のベーステーブルを判断する際に項目属性が考慮される

IN 演算子

データベースに対する異なるクエリ

For each のベーステーブルを判断する際に項目属性が考慮**されない**

これまでの説明をまとめると、データセレクターは 2 つの異なる方法で使用することができます。USING 節を使用する場合、For each 内に直接データセレクターの Order 節や Where 節を記述することと同様になります。また、Aggregate 式内で使用する場合は、データセレクターの "Conditions" で入力した条件を入力するのと同様になります。

前に説明したように、USING 節を宣言することで、データセレクターの順序とフィルタを For each の順序とフィルタに追加するように指示します。そのため、データセレクターに含まれる項目属性が、For each コマンドのベーステーブルの拡張テーブルに属している必要があります。

その一方で、Where 節内で IN 演算子を使用する場合は、独立したクエリを作成することになります。そのため、データセレクターは、そのベーステーブルを独立した For each であるかのようにナビゲートして解決される必要があります。
サブクエリを実行する際には IN 演算子を使用します。その結果が、後でフィルタとして For each で使用されます。

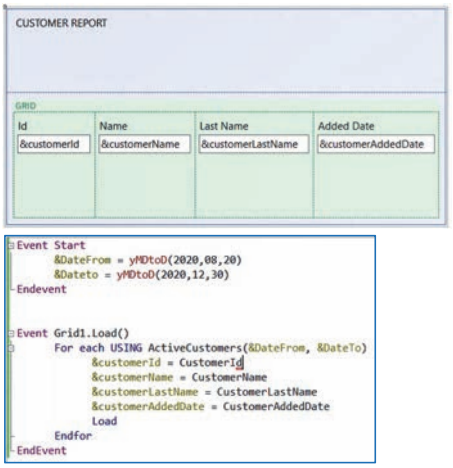
データセレクター内で定義した項目属性は、呼び出す方法によって、For each のベーステーブルを判断する際に考慮される場合と考慮されない場合があります。

- USING 節を使用してデータセレクターを呼び出す場合、そのオブジェクト内で宣言された項目属性は、For each のベーステーブルを判断する際に考慮されます。
- IN 節を使用してデータセレクターを呼び出す場合、項目属性は For each のベーステーブルを判断する際に考慮されません。

Web パネルでのデータセレクター

ベーステーブルを持つ Web パネル

ベーステーブルを持たない Web パネル




```
Event Start
      &DateFrom = yMDtoD(2020,08,20)
      &Dateto = yMDtoD(2020,12,30)
Endevent

Event Grid1.Load()
      For each USING ActiveCustomers(&DateFrom, &DateTo)
            &customerId = CustomerId
            &customerName = CustomerName
            &customerLastName = CustomerLastName
            &customerAddedDate = CustomerAddedDate
            Load
      Endfor
EndEvent
```

これまでに説明してきた 3 つの例は、手続き型オブジェクトに対するものでした。これは、情報を PDF ファイルで表示する必要があったためです。情報を Web パネルを使用して画面上に表示する場合は、どのようにしたらよいでしょうか。

ここで、指定した日付範囲でアクティブな顧客を、作成したデータセレクターを使用してリストにする最初の例に戻って考えてみましょう。ただし、今回は Web パネルを使用して、レコードを画面に表示します。

Web パネルの Web レイアウトでグリッドを宣言し、リストにしたい項目属性を追加します。
[Grid] プロパティに [Data Selector] があります。ここに、使用するデータセレクターの名前を入力します。これは、プロシージャーオブジェクトから、For each で USING 節を使用して呼び出すときと同じになります。

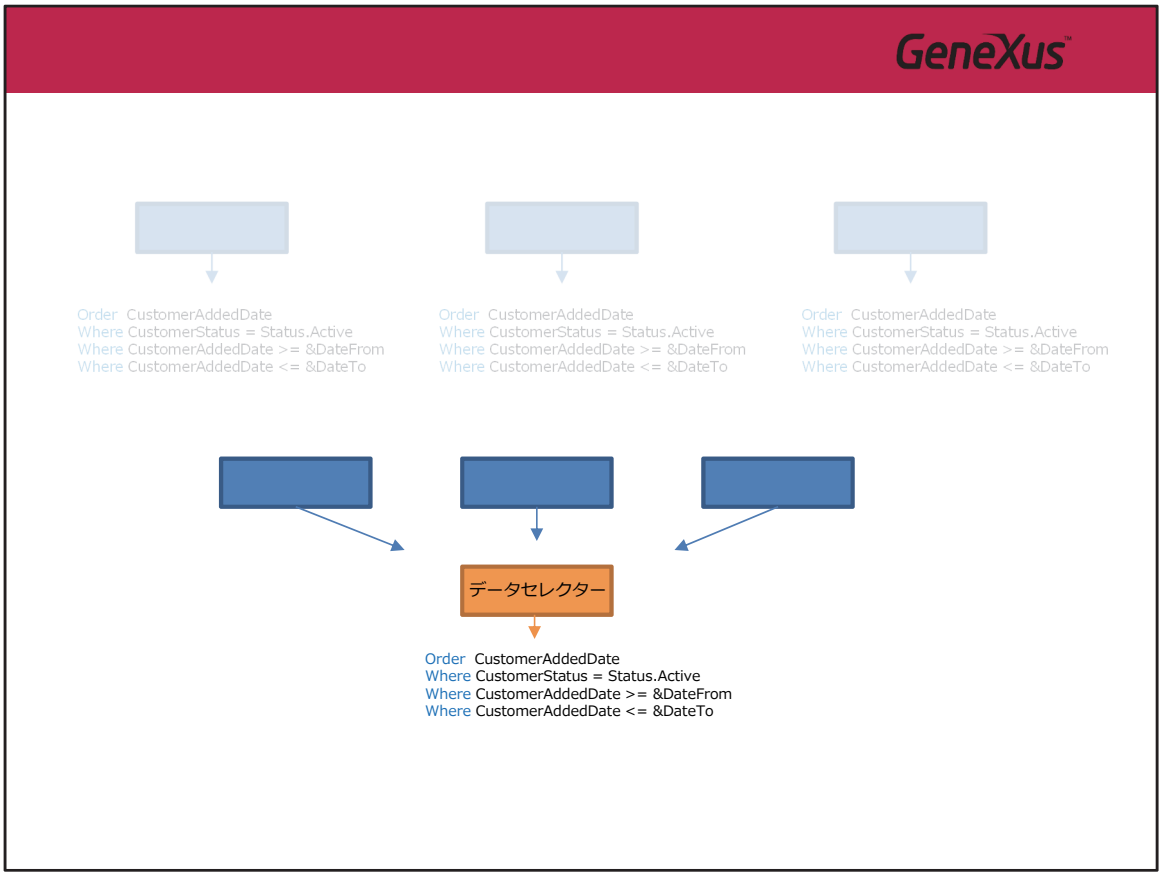
試してみましょう。エラーがスローされます。これは、ActiveCustomers データセレクターがパラメーターで日付を格納する 2 つの変数を受け取ったためです。この情報を渡す必要があります。
そのために、Date 型の DateFrom 変数と DateTo 変数を作成します。Start イベントでこれらを宣言し、テストするために日付の値で初期化します。
データセレクターにパラメーターを渡すには、グリッドの [Parameters] プロパティから行います。

もう一度実行します。
アクティブで、指定された日付範囲内にある顧客が画面に表示されます。データセレクターで定義したとおり、入力された日付で並べ替えられます。

同じ例で、Web パネルで項目属性ではなく変数が使用される場合を考えてみましょう。前の場合と異なり、Web パネルにはベーステーブルがありません。そのため、ここではグリッドの [Data Selector] プロパティは役に立ちません。すべての顧客を参照するには For each を使用する必要があり、そこでデータセレクターを呼び出す必要があるためです。この場合、ステートメントは Procedure オブジェクトの場合と同じようになります。

Event セクションで、グリッドの Load イベントを宣言する必要があります。For each 内に USING 節とデータセレクターの名前を記述し、日付をフィルタするために必要な変数をパラメーターで渡します。
また、For each 内で、表示したい項目属性の値をグリッド変数に割り当てます。
最後に Load コマンドを定義し、For each と Load イベントをクローズします。

これを実行すると、期待する結果になります。



要約すると、データセレクターは、さまざまなクエリや計算で使用/呼び出すパラメーター、条件、順序のセットを格納し、同じナビゲーションを何度も再利用できるオブジェクトです。

では、データセレクターはどこで使用できるでしょうか。データベースを照会するあらゆるケースで使用できます。たとえば、パネルや Web パネルのグリッド、データプロバイダーのグループで使用できます。

データセレクターの詳細については、GeneXus の Wiki を参照してください。