Carregando tipos de dados compostos

Objeto GeneXus: Data Provider

GeneXus™

Em vídeos anteriores vimos o conceito de tipo de dado estruturado, a possibilidade de defini-los simples ou como coleções, e alguns exemplos de carga manual, embora tenhamos mencionado que existe outra maneira de mais alto nível para realizar a carga.

Novo requisito: Ranking de países

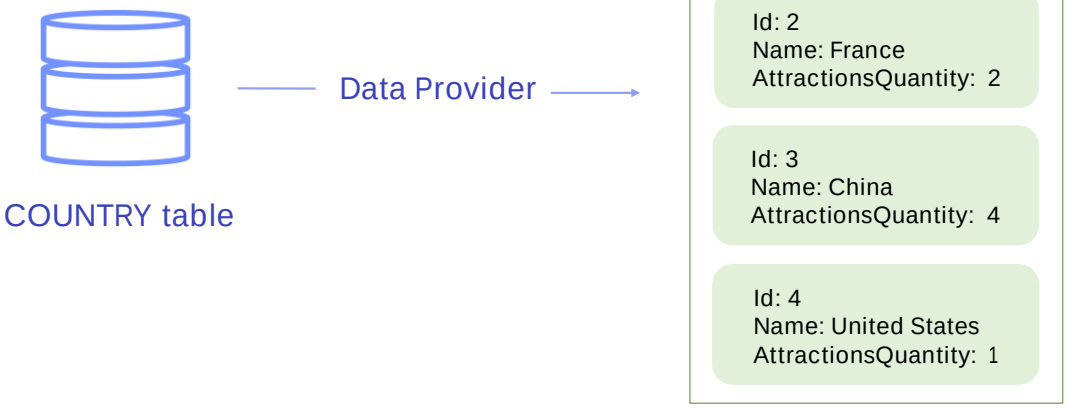
Ranking

Country name	Attractions quantity
France	3
United States	2
Egypt	1
Brazil	1
China	1
Uruguay	0

Se nos recordarmos, temos pendente para resolver um requisito solicitado pela Agência de Viagens que consiste em implementar um ranking de países segundo a quantidade de atrações turísticas que oferece.

Ou seja, devemos mostrar todos os países, ordenados da maior para a menor, por esta quantidade.

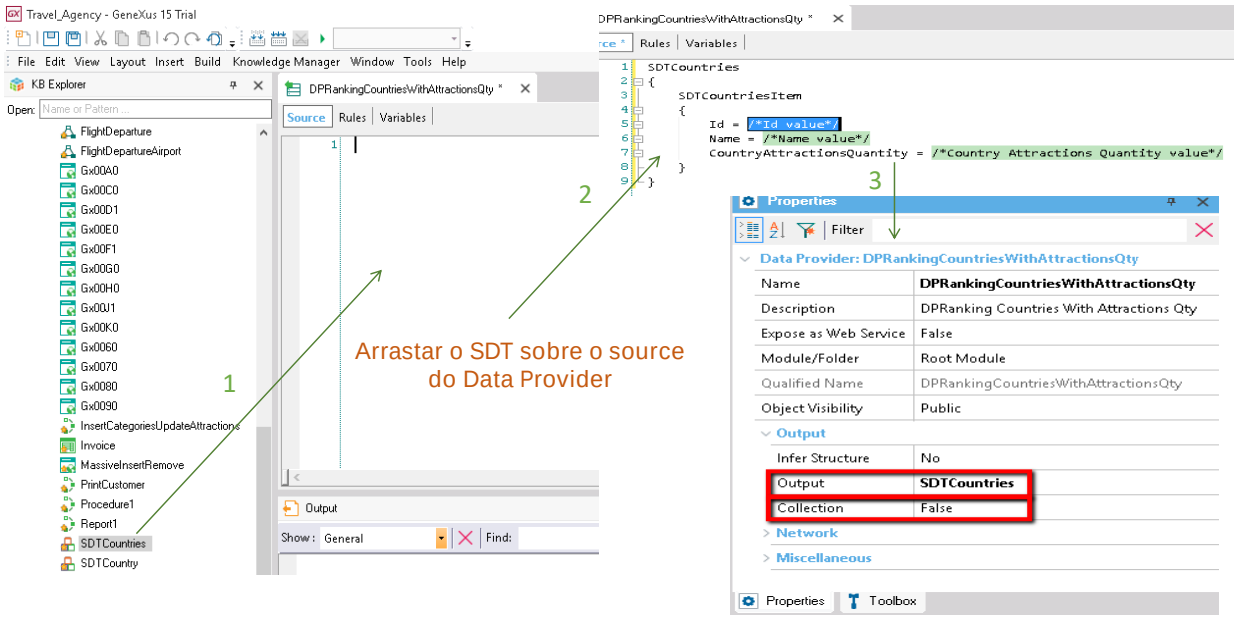
Estrutura da coleção



Salvamos esta definição, e para carregar os dados da coleção vamos utilizar um objeto GeneXus do tipo Data Provider.

Este objeto nos permite carregar uma estrutura de dados, por exemplo, a partir de informação obtida da base de dados, e nos devolve essa estrutura já carregada.

Novo requisito: Ranking de países



Criamos um objeto Data Provider e colocamos como nome RankingCountriesWithAttractionsQty

GeneXus nos posiciona na seção Source do Data Provider. Aqui é onde vamos declarar como queremos que os dados sejam carregados na coleção que queremos devolver. Observemos como é fácil declarar a carga: O que vamos fazer, simplesmente é, a partir da janela KB Explorer, arrastar o tipo de dado estruturado SDTCountries sobre o source do Data Provider.

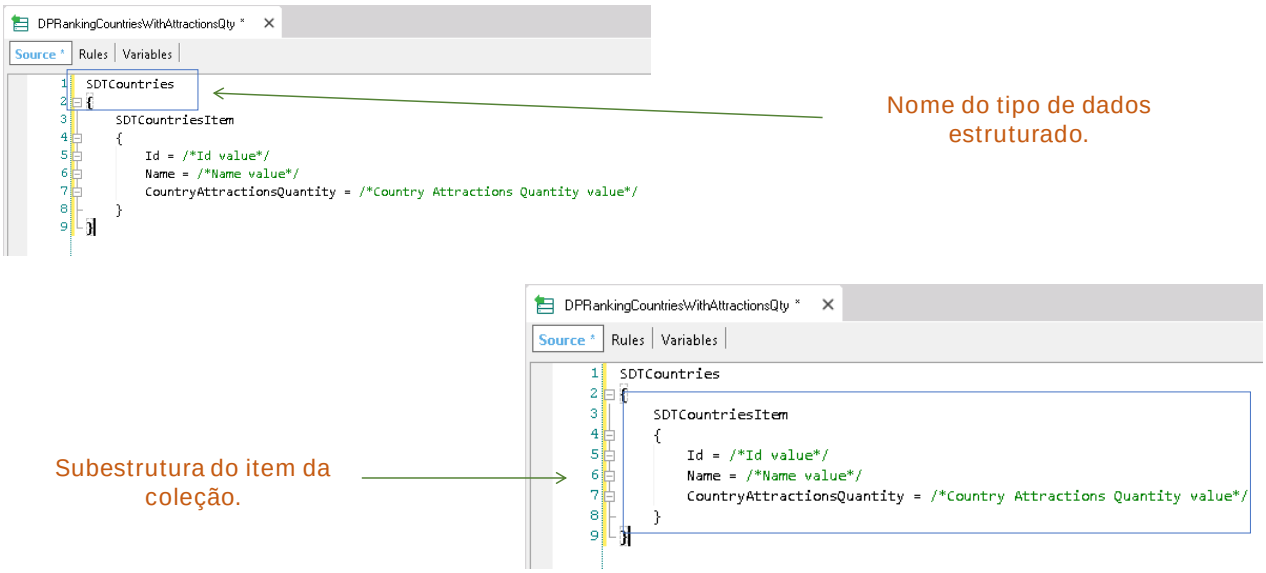
Ao fazer isto, GeneXus automaticamente escreve várias linhas de texto.

Se abrimos as propriedades do DataProvider, observemos que GeneXus atribuiu o nome da coleção SDTCountries à propriedade OutPut. Isto significa que o DataProvider devolverá uma coleção do tipo de dados estruturado SDTCountries, carregada com dados.

Como o SDTCountries já é uma coleção, não é necessário configurar a propriedade Collection com valor True do Data Provider.

É importante mencionar que, no caso de indicar esta propriedade Collection com valor True, o DataProvider devolverá uma coleção do SDT que tenha sido indicado na propriedade OutPut, sem importar o quão complexa possa ser sua estrutura.

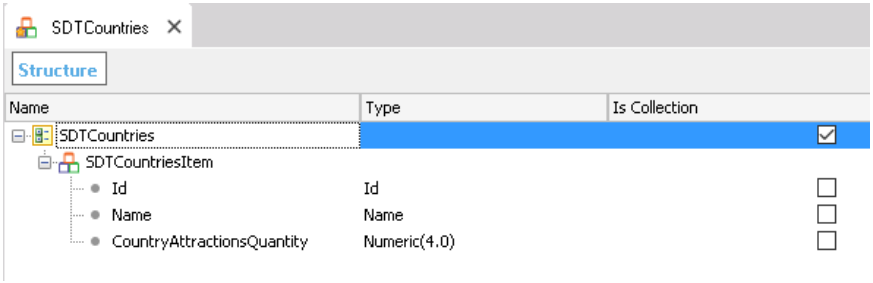
Novo requisito: Ranking de países

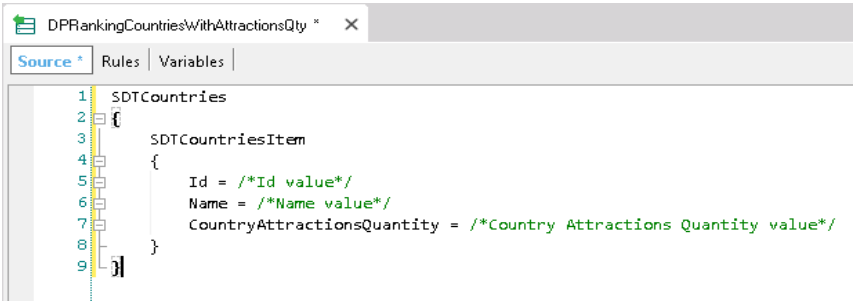


Estudaremos agora o que GeneXus escreveu no source.

Reconhecemos o nome do tipo de dados estruturado SDTCountries que é uma coleção. E depois entre as chaves está a subestrutura do item da coleção.

Novo requisito: Ranking de países

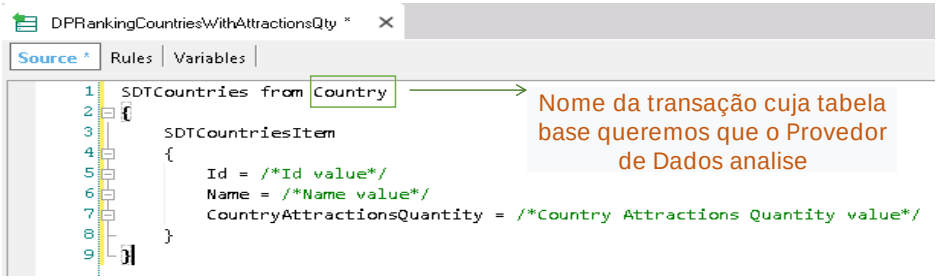




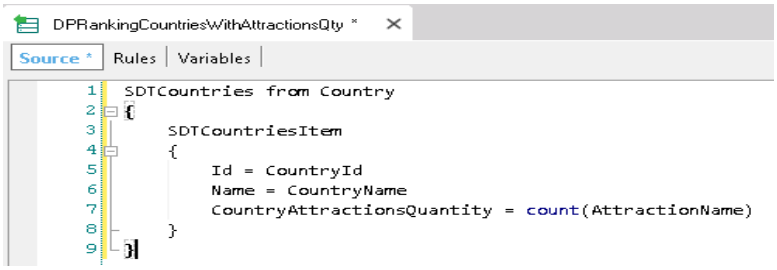
Comparemos isto com a estrutura do SDT:

Vemos que GeneXus representou em forma de texto a estrutura do SDTCountries. E nos deixou prontos os membros Id, Name e AttractionsQuantity da subestrutura do SDTCountries para lhes carregar seu valor.

Novo requisito: Ranking de países



São indicados os atributos ou cálculos com os quais serão carregados os elementos da coleção **:**



!ueremos carregar a coleção a partir do conteúdo da tabela COUNTRY.

Então, devemos indicar ao Data Provider que tem que percorrer essa tabela. Para isto usamos a cláusula From, e ao lado dela indicaremos o nome da transação cuja tabela base queremos percorrer, tal como fizemos para indicar a transação base do For each.

Então, neste caso, escrevemos: From Country

Se a transação tivesse mais de um nível, então para poder especificar um determinado nível, associado a determinada tabela base que queremos navegar, teríamos que escrever o nome da transação, ponto, o nome do nível.

Vamos indicar que ao elemento Id queremos carregá-lo com o valor do atributo CountryId, ao item Name o carregaremos com o valor do atributo CountryName, e ao item AttractionsQuantity queremos carregá-lo com a quantidade de atrações turísticas que tem cada país, então atribuímos a este membro: o resultado da fórmula inline Count(AttractionName).

Revisemos um conceito já estudado: e é que esta fórmula inline definida, navegará a tabela ATTRACTION, pelo atributo que temos indicado dentro dos parênteses. Além disso, entre a tabela navegada pelo Data Provider, ou seja, COUNTRY, e a tabela navegada pela fórmula, ou seja, ATTRACTION, há um atributo comum que é CountryId, esta fórmula contará as atrações do país navegado pelo Data Provider cada vez.

Novo requisito: Ranking de países

DPRankingCountriesWithAttractionsQty *

Source * | Rules | Variables

```
1  SDTCountries from Country
2  {
3      SDTCountriesItem
4      {
5          Id = CountryId
6          Name = CountryName
7          CountryAttractionsQuantity = count(AttractionName)
8      }
9  }
```

A tabela base do Data Provider é: COUNTRY

Assim, o que fizemos foi simplesmente: declarar uma tabela a ser navegada pelo Data Provider, e para cada registro acessado, indicamos os valores que queremos atribuir a um item novo na coleção de países.

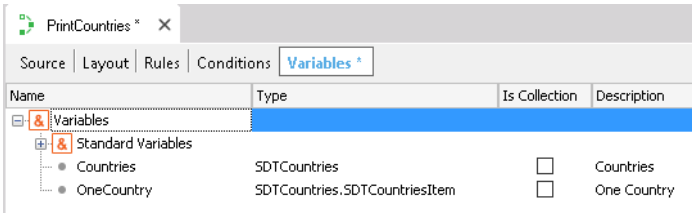
Dado que o Data Provider percorre a tabela COUNTRY, costumamos dizer que a tabela base do Data Provider, é COUNTRY.

O resultado final será que ficarão armazenados na coleção em memória, os dados de todos os países da base de dados, cada um com sua quantidade de atrações.

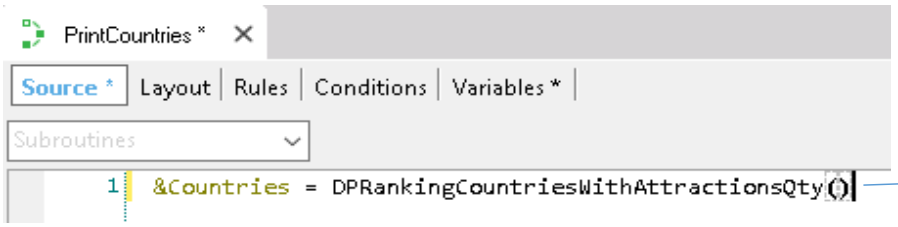
Novo requisito: Ranking de países

1) Criamos um procedimento

2) Definimos as variáveis:



3) No source do procedimento, invoca-se o Data Provider:



Recebe-se na variável &Countries o que retorna o Data Provider

Criamos um procedimento: "PrintRanking".

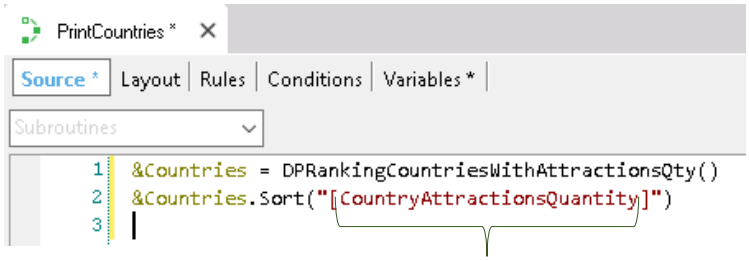
Na seção de variáveis vamos definir uma variável &Countries baseada no tipo de dado SDTCountries.

Vamos depois ao Source do procedimento, e daqui, a esta variável Countries, atribuímos-lhe o resultado que devolve o Data Provider que acabamos de criar.

Desta forma estamos invocando o Data Provider e este retornará uma coleção de países, que ficará carregada na variável &Countries.

Novo requisito: Ranking de países

Para implementar um ranking, a coleção deve ser ordenada do maior para o menor pela quantidade de atrações....
Usamos o método ***Sort***:



Campo do SDT pelo qual se deseja ordenar a coleção

Os colchetes dentro das aspas indicam a ordem inversa, isto é, do maior para o menor.

Recordemos que o requisito solicitado é visualizar um ranking de todos os países ordenados da maior para a menor segundo a quantidade de atrações que têm registradas.

Portanto, está faltando ordenar a coleção que obtivemos carregada. Ou seja, ordenar os itens da coleção de países, antes de ser mostrada, por ordem da maior para a menor segundo a quantidade de atrações que têm registradas.

Para resolver isto contamos com o método Sort. Recordemos que as variáveis ou atributos têm propriedades e métodos disponíveis, de acordo com seu tipo de dados. No caso de um SDT coleção, o método Sort permite ordenar a coleção por um determinado item.

A sintaxe é a seguinte:

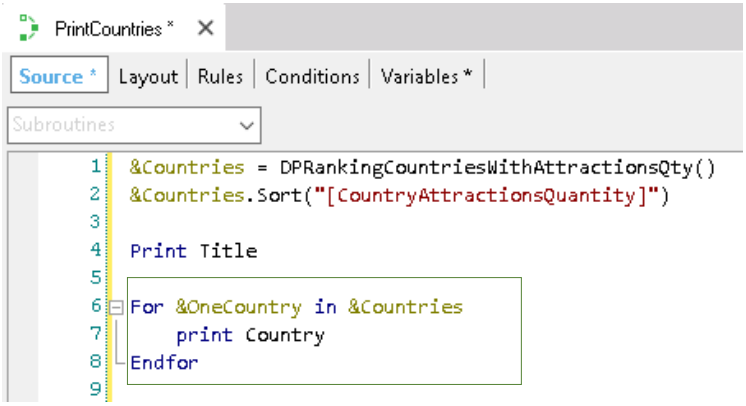
&Countries.Sort(″CountryAttractionsQuantity″)

Mas desta forma a coleção de países ficará ordenada da menor para a maior pela quantidade de atrações e nós precisamos que se ordene da maior para a menor, já que queremos implementar um ranking.

Assim, para indicar a ordem inversa, dentro das aspas adicionaremos colchetes.

Novo requisito: Ranking de países

Para percorrer a coleção de países e imprimir cada elemento no printblock, utilizamos a estrutura *For... in*



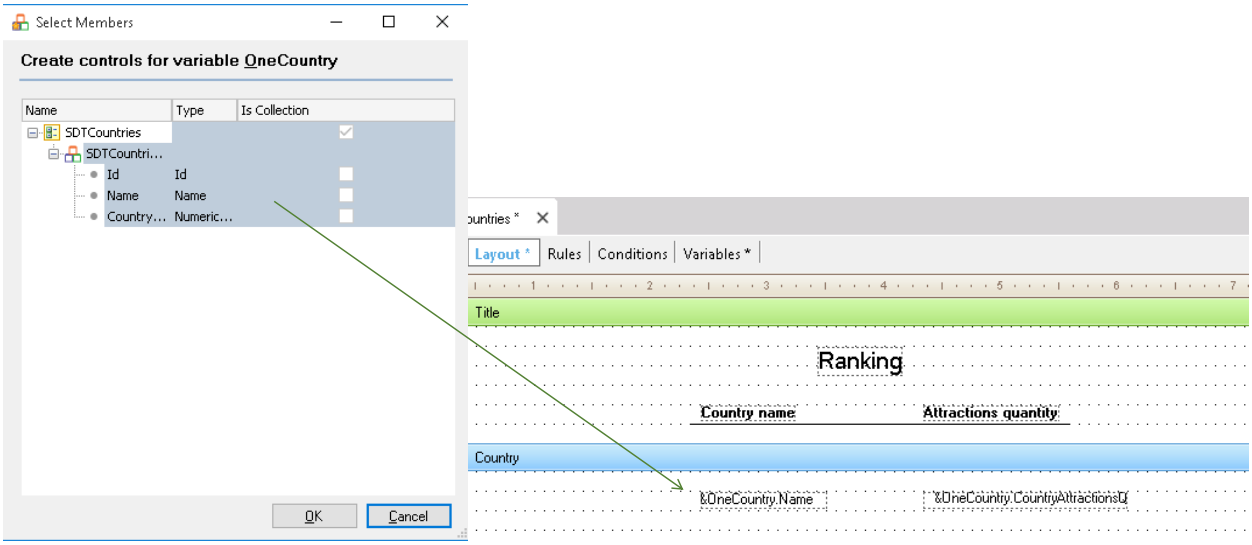
Uma vez ordenada a coleção devemos percorrê-la para poder mostrá-la no printblock. Para percorrer a coleção de países e imprimir cada elemento no printblock, utilizamos a estrutura For... in

For &oneCountry in &Countries .... Print do printblock Countries

Definimos a variável &oneCountry baseada no tipo de dado que corresponde ao elemento da coleção.

Novo requisito: Ranking de países

Inserimos as variáveis no printblock Country....



Vamos agora ao layout e inserimos a variável &oneCountry.

Novo requisito: Ranking de países

Ranking

Country name	Attractions quantity
France	3
United States	2
Egypt	1
Brazil	1
China	1
Uruguay	0

Já configuramos as propriedades necessárias, e a regra para definir o formato PDF, então nosso requisito solicitado está completo.

Para ver o nosso ranking em execução selecionamos a opção Run. E vemos a lista PDF com todos os países que estavam registrados na base de dados, cada um com sua correspondente quantidade de atrações e na ordem solicitada.

Clausula Where

```
SDTCountries from Country
Where CountryName <> "Frande"
{
      SDTCountriesItem
      {
            Id = CountryId
            Name = CountryName
            AttractionsQuantity = count(AttractionName)
      }
}
```

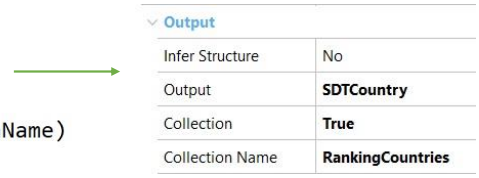
Os Data Providers aceitam opcionalmente a cláusula Where para filtrar, igual ao comando For each... por exemplo, se não queremos que a lista inclua a França, como faríamos?

Podemos adicionar a cláusula Where CountryName... diferente... de França.

Outra opção para implementar o requisito

Name	Type	Description	Is Collection
SDTCountry		SDTCountry	☐
Id	Numeric(4.0)	Id	☐
Name	Character(20)	Name	☐
AttractionsQuantity	Numeric(4.0)	Attractions Quantity	☐

```
SDTCountry from Country
{
     Id = CountryId
     Name = CountryName
     AttractionsQuantity = Count(AttractionName)
}
```



Name	Type	Is Collection	Description
Variables			
Standard Variables			
CountriesCollecition	SDTCountry	☑	Countries Collecition

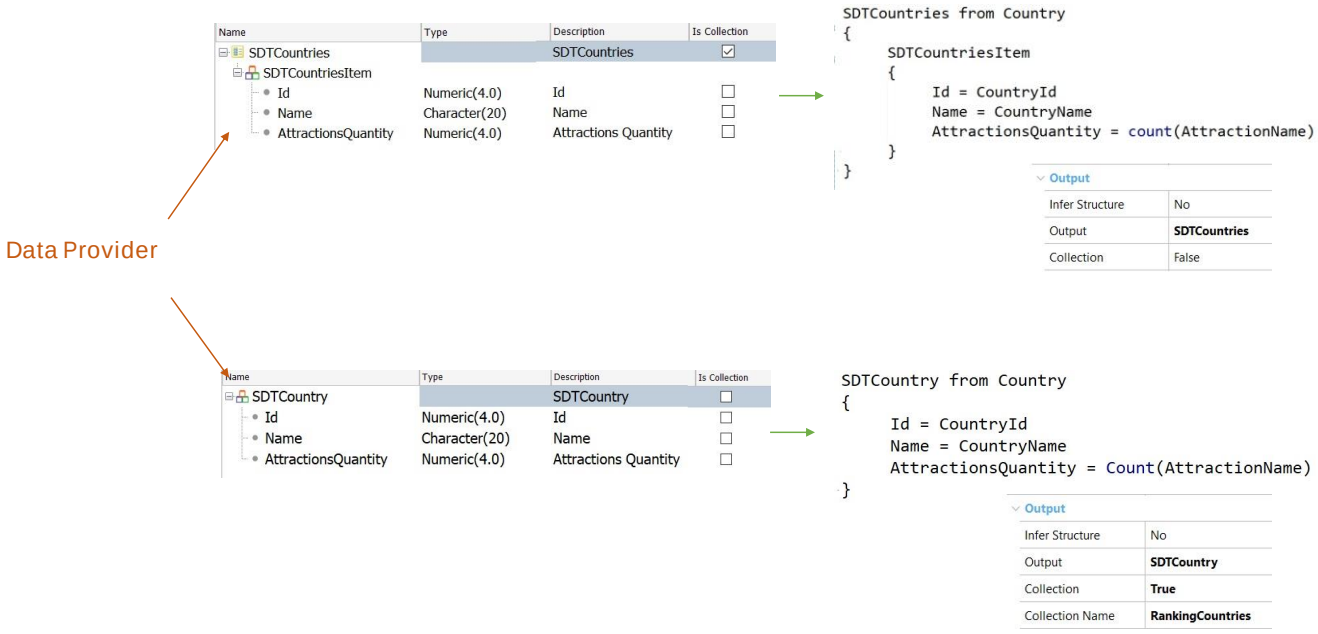
Outra forma de implementar este mesmo requisito, é a partir do SDT declarado simples e não como coleção.
Neste caso, a coleção deverá ser construída pelo Data Provider, e para isso deverá ser indicada sua propriedade Collection com o valor True.

Ao fazer isto, estamos indicando ao Data Provider que queremos que nos devolva uma coleção de elementos do tipo do SDTCountry. Também observemos que apareceu a propriedade Collection Name e que foi atribuído automaticamente um nome para a coleção.

Ao invocar o Data Provider a partir do source do procedimento, devemos definir a variável &CountriesCollection como uma coleção do tipo de dado estruturado SDTCountry.

Observemos que a sintaxe com a qual se invoca o Data Provider não muda, apenas que a variável que recebe o resultado está baseada em um SDT simples. Portanto, para que possa receber a coleção devolvida pelo Data Provider devemos declará-la como coleção marcando o check box IsCollection.

Resumo...



Resumindo, vemos que temos duas formas para que um Data Provider nos devolva uma coleção de elementos:

Uma delas, é definindo um tipo de dados estruturado do tipo coleção e ao arrastá-lo ao source do Data Provider automaticamente se configura o mesmo para retornar uma coleção desse tipo,

A outra opção é definindo um tipo de dados estruturado que não seja uma coleção e, em seguida, usando as propriedades do Data Provider podemos configurar que o mesmo objeto nos monte a coleção.

Desta forma, vimos o poder dos Data Providers para carregar informações em uma estrutura de dados em memória. Vimos como foi simples declarar os dados que queríamos carregar, GeneXus resolvendo tudo o que é necessário para realizá-lo.

GeneXus™

training.genexus.com
wiki.genexus.com