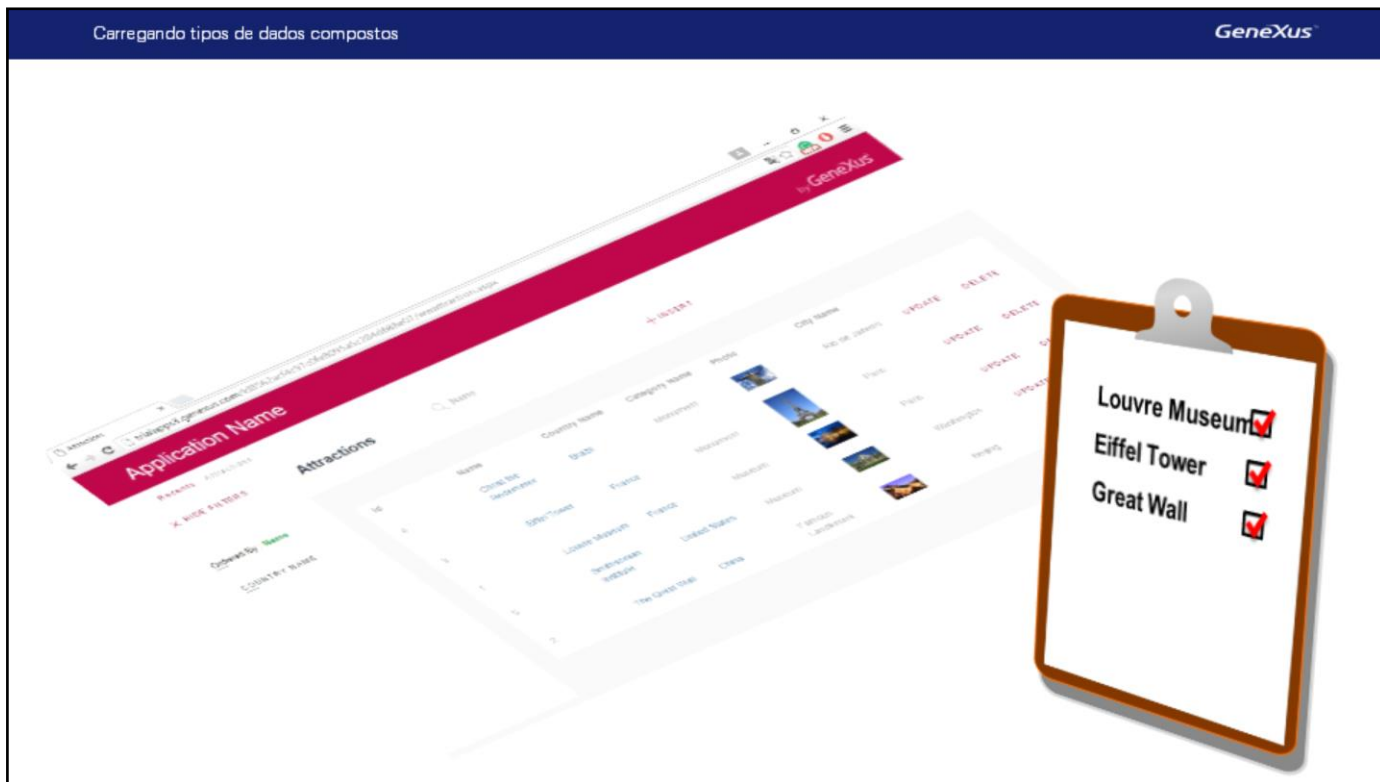


CARREGANDO TIPOS DE DADOS COMPOSTOS

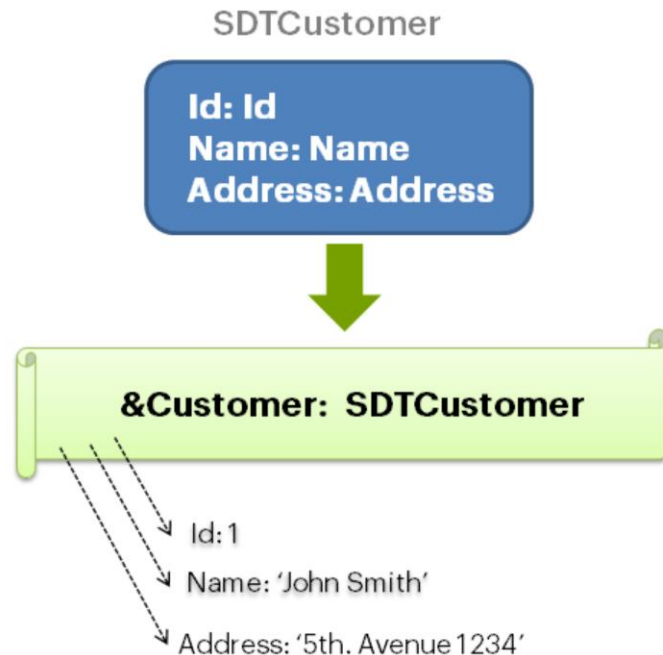
Objeto GeneXus: Data Provider

GeneXus™ 16

Em mais de uma ocasião, precisamos armazenar na memória **uma lista de elementos** que possuem o mesmo tipo de informação com diferentes valores armazenados.



Por exemplo, a agência de viagens pode precisar realizar operações com um grupo de clientes que compartilham uma característica comum, ou podem ser solicitados a processar informações sobre dados específicos em um grupo de atrações turísticas e isso pode significar que devemos carregar as listas em memória temporariamente.



Para resolver este tipo de exigência, temos de criar uma estrutura em memória **capaz de armazenar uma coleção de elementos**.

Temos visto que **tipos de dados estruturados**, como **SDTCustomer** neste exemplo, nos permitem definir estruturas que armazenam dados diversos, correspondendo a **um elemento**.

Nesta representação gráfica estamos então armazenando na memória o identificador, o nome e o endereço de um cliente.

SDTCustomer

Id: 1
Name: John Smith
Address: 5th. Avenue
1234



One customer

SDTCustomers

Id: 1
Name: John Smith
Address: 5th. Ave.

Id: 2
Name: Susan
Brown
Address: 7th.Ave.

Id: 3
Name: Robert Hill
Address: 81th. St..

Id: 4
Name: Peter Jensen
Address: St,Paul Rd.

.....



A collection of customers

Para armazenar vários elementos com dados do cliente, vimos que precisamos definir um tipo de dados estruturados e indicar que é uma coleção.

Novo requisito: Ranking de países

Ranking de países de acordo com a quantidade de atrações turísticas que possuem:



Vamos supor que temos que implementar um novo requisito para a agência de viagens.

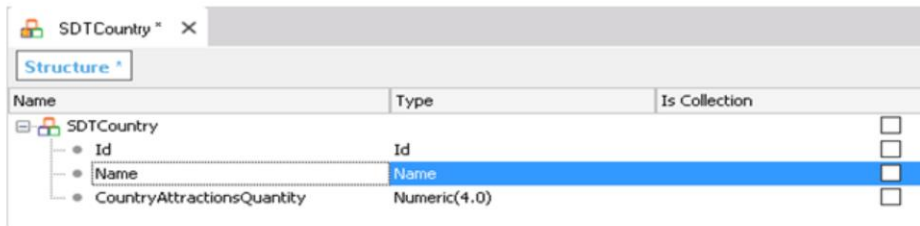
A agência de viagens deseja fornecer a seus clientes um ranking de países com base no número de atrações turísticas oferecidas.

Isto significa que temos de mostrar todos os países, ordenados do maior para menor no que se refere ao número de atrações oferecidas.

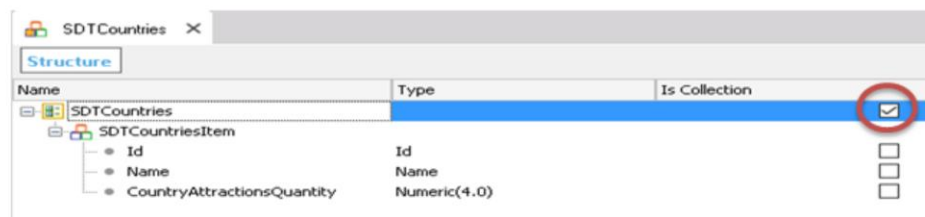
Para resolver este requisito, devemos primeiro carregar todos os países armazenados no banco de dados em uma estrutura, cada um com seu correspondente número de atrações turísticas. E então teremos que classificá-los de acordo com esse número do maior para o menor valor, para então, mostrá-los em um web panel ou uma lista.

Novo requisito: Ranking de países

Para carregar um país em memória:



Para carregar uma coleção de países em memória:



Vamos começar criando um novo objeto do tipo de dados estruturados, que chamamos de SDTCountries.

Para cada país, precisamos de seu identificador, nome e número de atrações turísticas registradas.

Com a primeira definição no slide, indicamos que estaremos guardando na memória as informações em relação a um único país.

Mas precisamos projetar um ranking de países, então devemos armazenar vários países. Portanto marcamos a caixa IS COLLECTION para ter uma estrutura que nos permitirá armazenar dados de vários países (como mostrado na segunda definição no slide).

Essa estrutura inclui os mesmos membros que definimos no início, embora agrupados em uma subestrutura chamada SDTCountriesItem. Esta subestrutura foi criada automaticamente quando marcamos o checkbox de coleção.

Cada item irá armazenar os dados de um país, e a coleção irá armazenar o grupo de dados dos países.

Collection structure

Id: 1
Name: Brazil
CountryAttractionsQuantity: 1

Id: 2
Name: France
CountryAttractionsQuantity: 2

Id: 3
Name: China
CountryAttractionsQuantity: 1

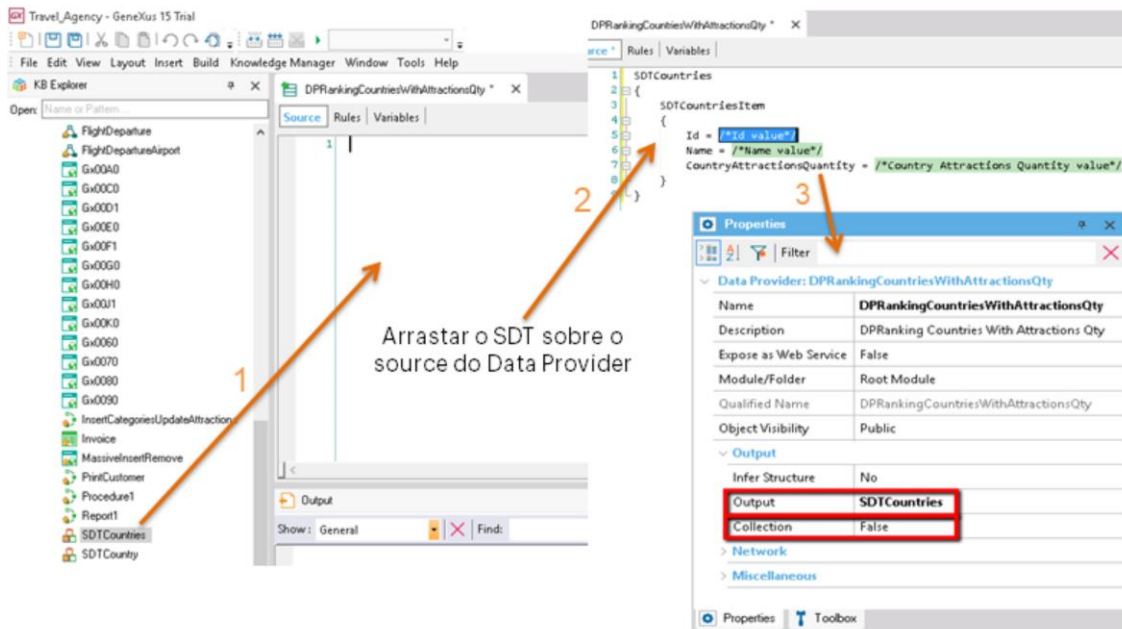
Id: 4
Name: United States
CountryAttractionsQuantity: 1



Para carregar os dados da coleção nós usaremos um objeto GeneXus chamado **Data Provider**.

O objeto Data Provider nos permite carregar uma estrutura de dados, por exemplo, com base em dados do banco de dados, e ele retorna a estrutura carregada.

Novo requisito: Ranking de países



Criamos um objeto Data Provider em GeneXus e o nomeamos: DPRankingCountriesWithAttractionsQty.

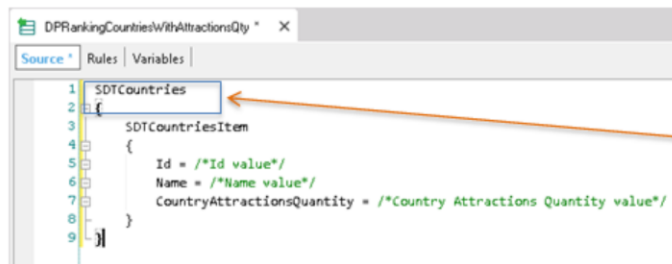
Podemos ver que GeneXus nos posiciona na seção Source do Data Provider. Aqui é onde declararemos como queremos que os dados sejam carregados na coleção que queremos retornados. Veja como é fácil declarar a carga: Vamos para a janela do KB Explorer e encontramos o structured data type SDTCountries e, em seguida, o arrastamos para o source do Data Provider.

Podemos ver que GeneXus automaticamente escreveu várias linhas de texto.

Se agora abrimos as propriedades do Data Provider, veremos que GeneXus atribuiu o nome da coleção SDTCountries para a propriedade Output. **Isto significa que o DataProvider irá retornar uma coleção do structured data type SDTCountries, carregado com dados.**

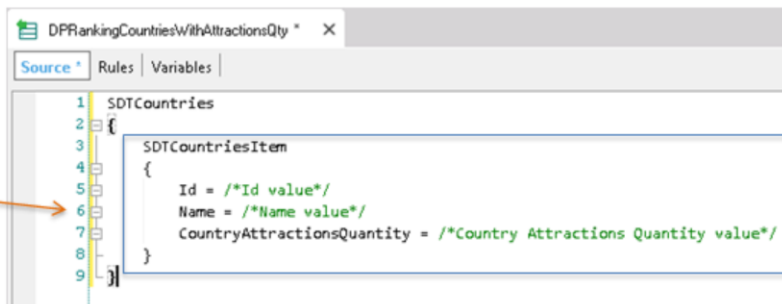
Como o SDTCountries já é uma coleção, não precisamos configurar a propriedade Collection do the Data Provider com o valor True. Teríamos que fazer isso se quiséssemos que o Data Provider retornasse uma coleção baseada em um structured data type simples.

Novo requisito: Ranking de países



Nome do tipo de dados estruturado.

Subestrutura do item da coleção.



Agora vamos analisar o que GeneXus escreveu no source.

Reconhecemos o nome do structured data type SDTCountries que é uma coleção. E lá dentro, a subestrutura do item da coleção está entre chaves.

Novo requisito: Ranking de países

The screenshot displays two windows from the GeneXus IDE. The top window, titled 'SDTCountries', shows the 'Structure' tab. It contains a table with the following data:

Name	Type	Is Collection
SDTCountries		<input checked="" type="checkbox"/>
SDTCountriesItem		
Id	Id	<input type="checkbox"/>
Name	Name	<input type="checkbox"/>
CountryAttractionsQuantity	Numeric(4,0)	<input type="checkbox"/>

The bottom window, titled 'DPRankingCountriesWithAttractionsQty', shows the 'Source' tab. It contains the following code:

```
1 SDTCountries
2 {
3   SDTCountriesItem
4   {
5     Id = /*Id value*/
6     Name = /*Name value*/
7     CountryAttractionsQuantity = /*Country Attractions Quantity value*/
8   }
9 }
```

Vamos comparar isso com a estrutura do SDT:

Podemos ver que GeneXus representou, como texto, a estrutura do SDTCountries, e forneceu membros Id, Nome e CountryAttractionsQuantity da subestrutura do SDTCountriesItem para carregar seus valores.

Novo requisito: Ranking de países



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



Como carregaremos essa coleção com base no conteúdo da tabela COUNTRY, devemos indicar para o Data Provider que ele deve ir àquela tabela. Para fazer isto usamos a cláusula From e ao lado dela, incluímos o nome da transação cuja tabela base queremos ir.

No nosso caso, Country.

Quando a transação tiver mais de um nível, a fim de especificar um nível associado a uma tabela base particular que queremos navegar, devemos escrever o **nome da transação, ponto, e o nome do nível**.

Em seguida indicamos que queremos carregar o elemento de ID com o valor do atributo CountryId, enquanto o membro Name será carregado com o valor de CountryName e o membro CountryAttractionsQuantity é para ser carregado com o número de atrações turísticas que cada país tem, então atribuímos a este membro o resultado da fórmula inline Count(AttractionName).

Vamos rever um conceito que já vimos: esta fórmula inline definida navegará a tabela ATTRACTION pelo atributo indicado dentro dos parênteses. Também, como há um atributo em comum entre as tabelas navegadas pelo Data Provider e a fórmula, que é CountryId, então a fórmula irá contar as atrações **do país** navegado pelo Data Provider cada vez.

Novo requisito: Ranking de países



**A tabela base do Data Provider é:
COUNTRY**

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

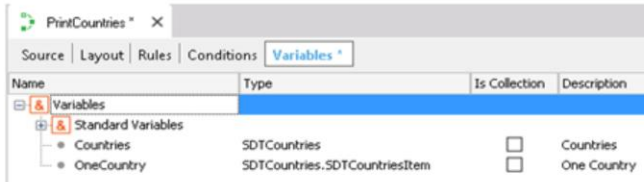
Como o Data Provider vai à tabela COUNTRY, costumamos dizer que a **tabela base do Data Provider é COUNTRY** :

O resultado final será que a coleção em memória irá armazenar os dados de todos os países no banco de dados, cada um com seu próprio número de atrações.

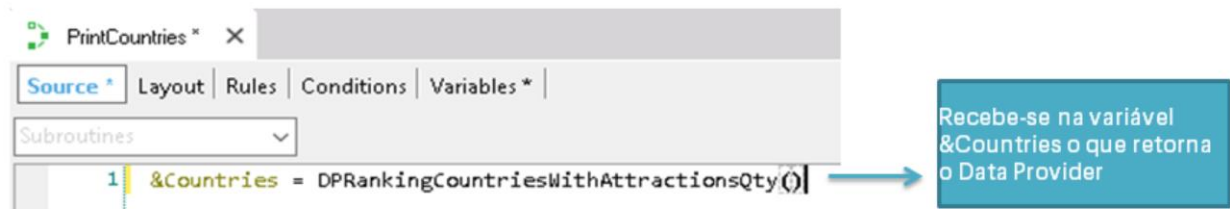
Novo requisito: Ranking de países

1) Criamos um procedimento

2) Definimos as variáveis:



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



Agora criamos um objeto Procedure para exibir o conteúdo da coleção de países. Chamamos este procedimento de "PrintRanking"

Vamos para a seção Variables deste procedimento e definimos a variável &Countries do tipo de dados SDTCountries.

Então vamos para o source, e carregamos esta variável do tipo coleção, atribuindo a ela o resultado retornado pelo Data Provider que acabamos de criar.

Com a instrução que escrevemos estamos **invocando** o Data Provider, que irá retornar uma coleção de países que será carregado 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.

Agora vamos recordar a exigência exata da agência de viagens: visualizar um ranking de todos os países **classificados do mais alto ao mais baixo número de atrações gravado**.

Portanto o que resta a fazer é classificar a coleção que carregamos. Ou seja: classificar os itens da coleção de países antes de mostrá-lo, na ordem do mais alto ao mais baixo número de atrações gravado.

Para resolver isto, temos o método Sort, e a sintaxe é a seguinte:

```
&Countries.Sort("CountryAttractionsQuantity")
```

Mas isso vai classificar a coleção dos países de menor ao maior número de atrações e precisamos o oposto porque precisamos implementar um ranking.

Então, para indicar a ordem inversa, dentro das aspas adicionamos colchetes.

Novo requisito: Ranking de países

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



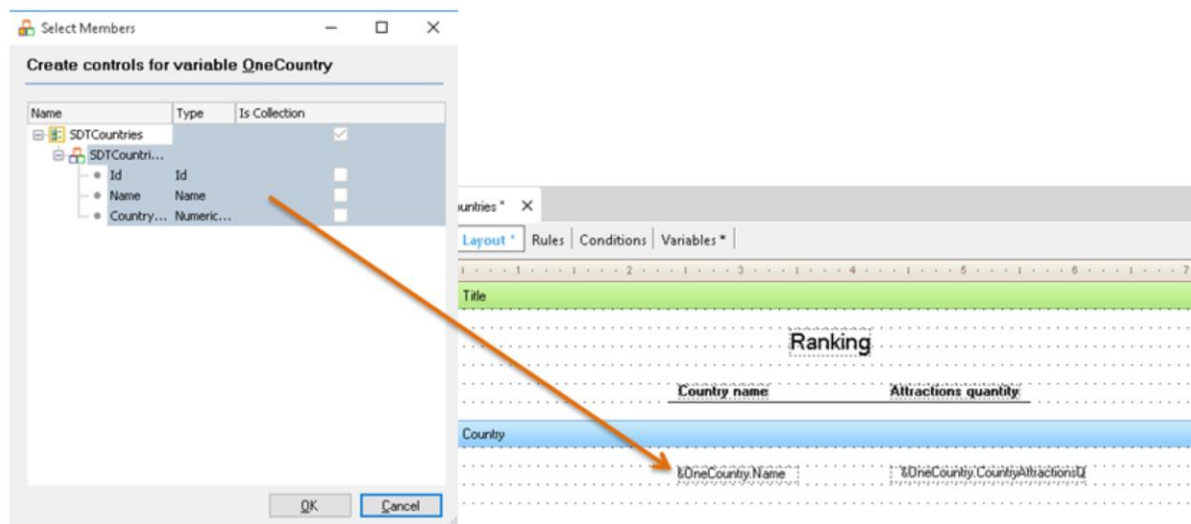
Agora a única coisa restante é ir à coleção novamente com o Data Provider e imprimir cada item, com os dados armazenados.

Para ir a uma coleção armazenada na memória, temos o comando **For element in Collection**.

Definimos uma variável &oneCountry para carregar nela cada elemento iterado da coleção.

Novo requisito: Ranking de países

Inserimos as variáveis no printblock Country....



Na seção Layout", chamamos este printblock "Country", e selecionamos: Insert / Variable e escolhemos &oneCountry

Novo requisito: Ranking de países

Ranking

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

Agora tudo o que temos a fazer é definir as propriedades necessárias para imprimir a lista em formato PDF.

Nós vamos para propriedades do procedimento e em “Main program”, escolhemos o valor True.

Em seguida, em “Main object properties”, selecionamos “Call protocol” e escolhemos "HTTP".

Finalmente temos que inserir a regra OutputFile na seção rule: selecionamos Insert / Rule e escolhemos a regra OutputFile.

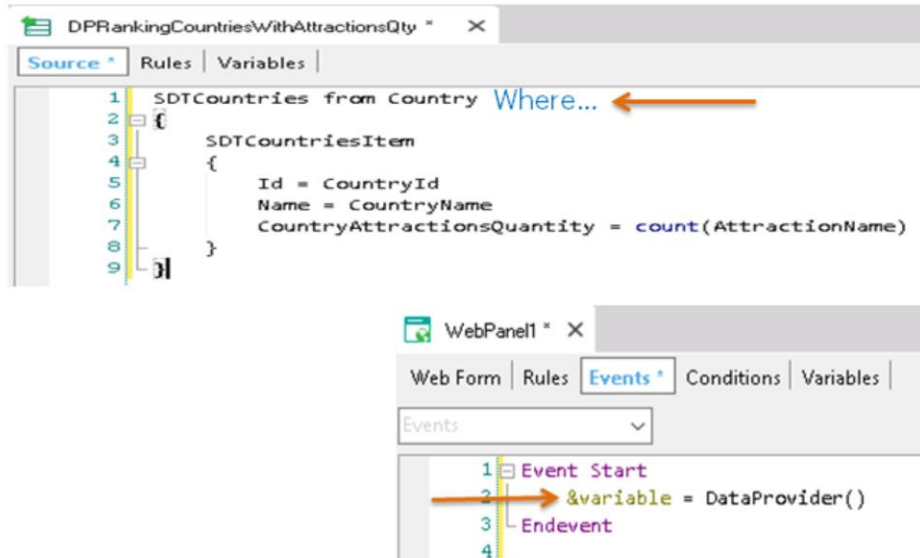
Terminamos, nomeando o arquivo como "Ranking.PDF" e o formato que usaremos como "PDF".

O desenvolvimento do requisito está agora completo. Vamos selecionar run sobre o procedimento para visualizar o ranking em tempo de execução!

E podemos ver a listagem pdf com todos os países armazenados no banco de dados, cada um com seu próprio número de atrações e a ordem dos países conforme solicitado!

Então, vimos o poder dos Data Providers para carregar dados em uma estrutura de dados na memória, especificamente no caso do tipo coleção. E vimos como é simples declarar o que queríamos carregar, e então GeneXus resolveu tudo o necessário para realizá-lo.

Novo requisito: Ranking de países



Data Providers, opcionalmente, admitem a cláusula Where para filtrar, como o comando For each... Mais adiante veremos outros exemplos onde os Data Providers são usados. Eles podem também ser chamados, em outras seções do objeto, como em eventos de web panel.



Videos

training.genexus.com

Documentation

wiki.genexus.com

Certifications

training.genexus.com/certifications