

Suppose that the countries and cities tables already have data. Since we previously deleted the attractions and categories data, our objective will be to initialize the categories and attractions tables with data so as not to start with empty tables.

To do so, we open the "MassiveInsertRemove" web panel that was created in the previous video, add a button labeled "Initialize data" and use the Data Provider together with the Business Components that we've just studied.

MassiveInsertRemove* ×
Web Form * Rules Events Conditions Variables
✓ <no action="" group="" selected=""></no>
MainTable Initializedata
Remove Data



If we could obtain a collection variable of Business Component items corresponding to Category, loaded with the categories to be added to the database, we will only have to apply the Insert() method to this collection variable, because as we said in the previous video, this will allow us to Insert all the items, that is to say, all the Business Components in the collection.

	GeneXus
BC Category Category Category Category	
&categories	
&categories.Insert()	

RassiveInsertRemo	we• ×			•
Web Form * Rules	Events Conditions	Variables *		
Name	Туре	Is Collection	Description	
■ & Variables				
 attraction 	Attraction		attraction	
category	Category		category	
	Category		categories	

So, now we only need to obtain this collection. How do we go about it?

So far, we've known that a Data Provider allows us to return structured data, of both simple and collection type.

Page.

			GeneXus
	O Properties	* ×	
	E 21 V Filter	×	
		kingCountriesWithMttractionsQty	
	Name	RunkingCountriocMithAttraction.	
Data Browidar	Description	Ranking Countries With Attractio	
Data Provider	Expose as Web Serv	ic False	
	Module/Folder	Root Module	
	Qualified Name	RankingCountriesWithAttractions	
	Object Visibility	Public	ainanta
	~ Output		simple
	Infer Structure	No	
	Output	SDTSomething <	SDT
	Collection	False	
	Vetwork		collection
			Concetion

In this case, we want to return a category collection, but these categories are not structured data types; instead, they are **Business Components**.

			GeneXus
	Properties	* ×	
	Vane	AngCountriesWithAttractionsQty	
	Description	Ranking Countries With Attractio	
ta Provider	Expose as Web Servio		
	Module/Folder	Root Module	
	Qualified Name	RankingCountriesWithAttractions	
	Object Visibility	Public	aimala
	~ Output		simple
	Infer Structure	No	
	Output	SDTSomething	SDT / BO
	Collection	False	
	 Network 		collection

However, the structure of a Business Component is exactly the same as that of an SDT. Therefore, Data Providers will also allow us to load and return Business Components, of both simple and collection type.

Our solution will come from there. Let's create a Data Provider to load the categories, and call it Category_DP.

Page 3

New Object			×
Select a Category:	Select a Type:		
Common Workflow Reporting Documentation Web Smart Devices	Color Palette Data Provider Data Selector Comman Procedure Structured Data Type Subtype Group Transaction		
Describe data estas ation l			
Describe data extraction	ogic here and use it anywhere you need it.		
Name:	Category_DP		
Description:	Category_DP		
		Create	Cancel

We drag the Category transaction to the Data Provider Source and see that it writes the transaction structure. Note that to the left we have the Business Component's elements, which will be saved in memory, and that have the same name as the attributes even though they are not attributes.

Meanwhile, the attributes of the corresponding table are now displayed on the right side. From there, the Data Provider will obtain the data to load the BC that is saved in memory.

🛃 MassiveInsertRemove* × 🔚 Category_DP* ×
Source * Rules Variables
1 Category 2 ∃ { 3 CategoryId = CategoryId 4 CategoryName = CategoryName 5 - }

If this is what we wanted, the Data Provider should return a collection of this Business Component, because the table has many records. In the properties we can see that the Output property now has the Business Component value, but the Collection property is not set to True as we need.

Properties	Р Х
〕 ■ Al 🌾 Filter	×
 Data Provider: Cate 	gory_DP
Name	Category_DP
Description	Category_DP
Expose as Web Servio	False
Module/Folder	Root Module
Qualified Name	Category_DP
Object Visibility	Public
✓ Output	
Infer Structure	No
Output	Category
Collection	False
V Network	
Connectivity Supp	Inherit
V Miscellaneous	
Generate Object	True

So, we change it and the new Collection name property is displayed. By default, it takes the Data Provider name. We change it to CategoryCollection.

Data Provider: Category_DP			
Name Category_DP			
Description	Category_DP		
Expose as Web Service False			
Module/Folder	Root Module		
Qualified Name	Category_DP		
Object Visibility	Public		
~ Output			
Infer Structure	No		
Output	Category		
Collection	True		
Collection Name	CategoryCollection		
> Network			

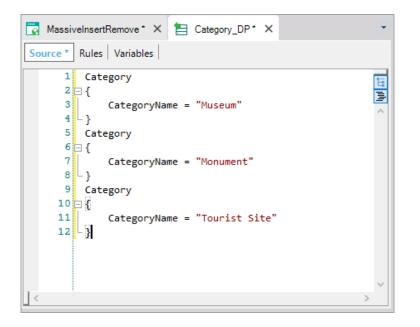
In addition, we don't want to load this collection with data from the database; instead, we want to assign it new values entered by us.

Therefore, we enter groups associated with the collection items one by one:

 $\mathsf{Page}\mathsf{S}$

RassiveInsertRe	move* × 挂 Category_DP* ×	-
Source * Rules \	/ariables	
	ntegoryId = 1 ntegoryName = "Museum"	100 Emiliaria
7 ⊟ { 8 Ca 9 Ca 10 - }	ntegoryId = 2 ntegoryName = "Monument"	
	rry itegoryId = 3 itegoryName = "Tourist Site"	
[<		>

Since CategoryId is an autonumbered attribute, we don't need to assign it a value when we want to insert a record. This is what we will do next, so we simply delete this assignment:



And since we want to return a collection called CategoryCollection —even though it's not necessary because by setting the Collection property to True, the Data Provider knows that it will return a collection— to clarify the code we can explicitly indicate what GeneXus has already inferred: to do so, we enclose all the Category groups in the CategoryCollection group corresponding to the collection.

Image: MassiveInsertRemove* × ★ ★ ★	-	Properties	7 ×
Source * Rules Variables		📜 灯 🌾 🛛 Filter	×
1 CategoryCollection	-	 Data Provider: Categoria 	jory_DP
2 ⊡ {		Name	Category_DP
3 Category 4 - {	^	Description	Category_DP
5 CategoryName = "Museum"		Expose as Web Servic	e False
6 - } 7 Category		Module/Folder	Root Module
8 3 4		Qualified Name	Category_DP
9 CategoryName = "Monument" 10 - }		Object Visibility	Public
11 Category		✓ Output	
12 { 13 CategoryName = "Tourist Site"		Infer Structure	No
14 - }		Output	Category
15 - }		Collection	True
	~	Collection Name	CategoryCollection
[] <	>	✓ Network	

Now we only need to invoke this Data Provider from the event associated with the web panel button:

```
Event 'Initialize data'
    &Categories = Category_DP()
Endevent
```

and enter this in the database:

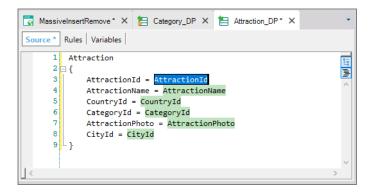
```
Event 'Initialize data'
   &Categories = Category_DP()
   &Categories.Insert()
   Commit
Endevent
```

Next, we will have to initialize the attractions table. Likewise, we will create a Data Provider called Attraction_DP.

New Object			×
Select a Category:	Select a Type:		
Common Workflow Reporting Documentation Web Smart Devices	Color Palette		
Describe data extraction	logic here and use it anywhere you need it.		
Name:	Attraction_DP		
Description:	Attraction_DP		
		Create	Cancel

We drag the Transaction (from which we had already obtained the Business Component) and see that every element of the Business Component is initialized by default with the corresponding attribute in the table.

Page



Once again, we see that only the attributes physically present in the table are taken into account, and that the attributes inferred in the transaction or formulas are not included.

Since we're not interested in loading existing attractions (because we run this Data Provider to load the initial data), we delete all these attributes and enter these values manually. In addition, since the ID is autonumbered, we don't need to assign a value to this Business Component element either. The attractions' photos will be assigned later, so we also remove this attribute.

☑ MassiveInsertRemove* × Image: Category_DP × Image: Attraction_DP* ×	٣
Source * Rules Variables	
1 Attraction	12
2 🖃 {	13 13
3 AttractionName = "Louvre Museum"	
4 CountryId = 2	^
5 CategoryId = 1	
6 CityId = 1	
7 - }	
8 Attraction	
9 🖃 {	
10 AttractionName = "The Great Wall"	
11 CountryId = 3	
12 CategoryId = 3	
13 CityId = 1	
14 - }	
15 Attraction	
16 - {	
17 AttractionName = "Eiffel Tower"	
18 CountryId = 2	
19 CategoryId = 2	
20 CityId = 1	
21 - }	
22 Attraction	
23 🖃 {	
24 AttractionName = "Christ the Redemmer"	
25 CountryId = 1	
26 CategoryId = 2	
27 CityId = 1	
28 - }	
29 Attraction	
30 🖃 {	
31 AttractionName = "Smithsonian Institute"	
32 CountryId = 4	
33 CategoryId = 1	
34 CityId = 2	
35 - }	
36 Attraction	
37 🖃 {	
38 AttractionName = "Matisse Museum"	
39 CountryId = 2	
40 CategorvId = 1	۷
< >>	

We're assigning the Countryld, Cityld and Categoryld values by heart, which means that they may not exist in the corresponding tables. If any of the values doesn't exist, when trying to insert the records with the Business Component, the corresponding referential integrity checks will be triggered and the insertion will fail.

To avoid assigning values that may not exist, we will use the Find formula to find the correct identifiers based on the name of the country, city or category.

Massivelns	ertRemove × 🔁 Category_DP × 扫 Attraction_DP* ×	
ource * Rule	es Variables	
3	Attraction	12
4 🖨	{	
5	AttractionName = "Louvre Museum"	-
6	CountryId = find(CountryId, CountryName = "France")	
7	CategoryId = find(CategoryId, CategoryName = "Museum")	- 6
8	CityId = find(CityId, CityName = "Paris")	
9 -	}	
10	Attraction	
11 🛱	{	
12	AttractionName = "The Great Wall"	
13	CountryId = find(CountryId, CountryName = "China")	
14	CategoryId = find(CategoryId, CategoryName = "Tourist Site")	
15	CityId = find(CityId, CityName = "Beijing")	
16 -	}	
17	Attraction	
18	{	
19	AttractionName = "Eiffel Tower"	
20	CountryId = find(CountryId, CountryName = "France")	
21	CategoryId = find(CategoryId, CategoryName = "Monument")	
22	CityId = find(CityId, CityName = "Paris")	
23 -	}	
24	Attraction	
25	(Attended and a life int the Dedemonal	
26	AttractionName = "Christ the Redemmer"	
27	CountryId = find(CountryId, CountryName = "Brazil")	
28	CategoryId = find(CategoryId, CategoryName = "Monument")	
30 -	CityId = find(CityId, CityName = "Rio de Janeiro")	
31	} Attraction	
32		
33	۱ AttractionName = "Smithsonian Institute"	
34	CountryId = find(CountryId, CountryName = "United States")	
35	CategoryId = find(CategoryId, CategoryName = "Museum")	
36	CityId = find(CityId, CityName = "Washington")	
37 -	}	
38	, Attraction	
39	{	
	L	1
		>

Note that the Find formulas are accessing the database only to search for the identifiers corresponding to the names we've used, but the rest of the values assigned to the Business Component are fixed.

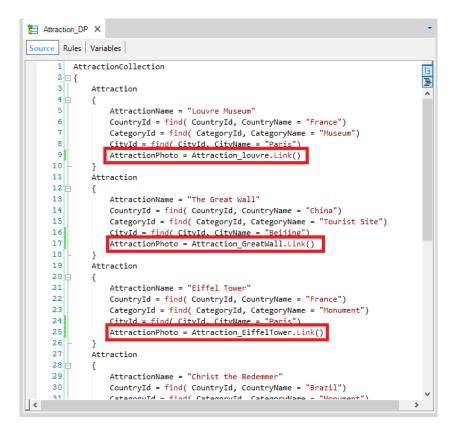
Just like we did with the categories Data Provider, we must set the Collection property to True because we will return many attractions. Also, we will adjust the notation in the source by enclosing the groups inside the AttractionCollection group to indicate that it is an attraction collection.

Page9

🔜 MassiveInsertRemove × ╞ Category_DP × 🍃 Attraction_DP* × 🔹	0	Properties	P	×
Source * Rules Variables	>	🚦 🛃 🌾 Filter		×
1 AttractionCollection	~	Data Provider: Att	traction_DP	
1 AttractionCollection		Name	Attraction_DP	
3 Attraction		Description	Attraction DP	
			-	
5 AttractionName = "Louvre Museum"		Expose as Web Serv	False	
6 CountryId = find(CountryId, CountryName = "France") 7 CategoryId = find(CategoryId, CategoryName = "Museum")		Module/Folder	Root Module	
8 CityId = find(CityId, CityName = "Paris")		Qualified Name	Attraction_DP	
9 - }		011 000 000	D.L.C	
10 Attraction		Object Visibility	Public	
11 🗄 {		✓ Output		
12 AttractionName = "The Great Wall"		Infer Structure	No	
13 CountryId = find(CountryId, CountryName = "China")	Ν.	inter structure	IND	
<pre>14 CategoryId = find(CategoryId, CategoryName = "Tourist Site")</pre>	Ν.	Output	Attraction	
15 CityId = find(CityId, CityName = "Beijing")		Collection	True	
16 - }			-	
17 Attraction		Collection Name	AttractionCollection	
18 0 {		✓ Network		
19 AttractionName = "Eiffel Tower"				
<pre>20 CountryId = find(CountryId, CountryName = "France")</pre>		Connectivity Sup	Inherit	
21 CategoryId = find(CategoryId, CategoryName = "Monument")		✓ Miscellaneous		
<pre>22 CityId = find(CityId, CityName = "Paris")</pre>		Generate Object	True	
23 - }		Generate Object	nue	
24 Attraction				

To also load the attractions' photos, we may insert them first as image objects in the KB...

Next, for each Data Provider group we may simply assign the name of the image to AttractionPhoto.



Now we only have to invoke the Data Provider so that it returns the loaded collection...

Page **L**(

MassiveInsertRemove *	× 扫 Category_DP ×	E Attraction_DP	× -
Web Form * Rules Event	s* Conditions Variables	s *	
Name	Туре	Is Collection	Description
🖃 🚷 Variables			
🗄 🚷 Standard Variables			
• • attraction	Attraction		attraction
 category 	Category		category
categories	Category	\checkmark	categories
attractions	Attraction		attractions

Veb Form R	ules Events Conditions Variables	
Teo Form N		
ents	~	
1 🖂 Ev	vent 'RemoveData'	ſ
2 🖨	For each Attraction	-
3	<pre>&attraction.Load(AttractionId)</pre>	Ŀ
4	<pre>&attraction.Delete()</pre>	
5		
6 -	endfor	
7 📥	For each Category	
8	<pre>&category.Load(CategoryId)</pre>	
9	<pre>&category.Delete()</pre>	
10 -	endfor	
11	Commit	
12 L EI	ndevent	
13		
14		
15 🖃 Ev	vent 'Initialize data'	
16	<pre>&categories = Category_DP()</pre>	
17	<pre>&categories.Insert()</pre>	
18	Commit	
19		
20	<pre>&attractions = Attraction_DP()</pre>	
21	<pre>&attractions.Insert()</pre>	
22	Commit	
23 - EI	ndevent	

Note that to be able to insert attractions, the categories must have been created first; for this reason, the order is the one we used in the event code.

Massive Ins	ert Remove X	1	-		×
← → C	trialapps3.genexus.com/ld8562acf4c97c0fe8095a5c284d688e07/massive	₽ ☆	2	<u>a</u> 0	≡
Ар	plication Name	by	Ge	neXu	Ŝ
Rece	nts Customer — Category — Attractions — Massive Insert Rem				
Remove Da	ata Initialize data				
	Initialize data				

Category × ← → C ☐ trialapps3.gene		095a5c284d688e07/category.aspx	i - □ ×
Application	Name		by GeneXuš
Recents Customer —	Attractions — Massive Insert F	tem — Calegory	
	Category	« < > » select	
	ld	4	
	Name	Museum	
		CONFIRM CANCEL DELETE	

+ -> C 🗋 trialapps	×)	4c97c0fe8095a5c284	d688e07/wwattrac	tion.aspx			ے ایک	2 a 🤮 🗘
Applicati	on Name							GeneXus
Recents Custom	ner — Massive Insert Rem.	— Category — Attr	actions					
× HIDE FILTERS	Attractions	Q Name	2		+ INSERT			
Ordered By : Name	Id	Name	Country Name	Category Name	Photo	City Name		
COUNTRY NAME	11	Christ the Redemmer	Brazil	Monument		Rio de Janeiro	UPDATE	DELETE
	10	Eiffel Tower	France	Monument	A	Paris	UPDATE	DELETE
	14	Forbidden city	China	Tourist Site		Beijing	UPDATE	DELETE
	8	Louvre Museum	France	Museum	-	Paris	UPDATE	DELETE
	13	Matisse Museum	France	Museum		Nice	UPDATE	DELETE
	12	Smithsonian Institute	United States	Museum	Mailtin	Washington	UPDATE	DELETE
	9	The Great Wall	China	Tourist Site		Beijing	UPDATE	DELETE

```
Event 'Delete data'
&attractions = AttractionsFromDB_DP()
&attractions.Delete()
Commit
&categories = CategoriesFromDB_DP()
&categories.Delete()
Commit
endevent
```

Let's select Commit in GeneXus Server.

om	ment									
		aliz tio	e categories and at ns.	tractions through	Data Provid	ders returning	Business C	omponent	Rec	cent Comments
Pe	nding	g Cor	nmits (12/12) Ignored C	Dijects						Y (
¥			Name 4	Туре	Descriptio	Modified On	Module	Action	Last Synchro	User
•			Attraction_Christ_the	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
~		1	Attraction_DP	Data Provider	Attractio	9/7/2016 10:20	Root Module	Inserted	7/5/2016 1:0	ARTECH\CFe
~			Attraction_Egypt_Gre	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
¥			Attraction_EiffelTower	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
¥	-		Attraction_forbidden	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
~			Attraction_GreatWall	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
¥			Attraction_louvre	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
¥	.		Attraction_MatisseMu	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
¥	, D		Attraction_The_Smith	Image	Attractio	9/6/2016 11:15	None	Inserted	7/5/2016 1:0	ARTECH\CFe
¥	.	ħ	Category_DP	Data Provider	Categor	9/6/2016 9:43 P	Root Module	Inserted	7/5/2016 1:0	ARTECH\CFe
¥	4		EiffelTower	Image	Eiffel To	9/6/2016 11:11	None	Inserted	7/5/2016 1:0	ARTECH\CFe
~	8		MassiveInsertRemove	Web Panel	Massive	9/7/2016 11:21	Root Module	Modified	9/6/2016 8:4	ARTECH\CFe

In this video we saw how a Data Provider not only allows loading a structure with data from the database,

		GeneXus
Data Provider	SDT / BC	
000000		



but also from fixed data.

		GeneXus
Data Provider	SDT / BC	
0000880		

In addition, it can also do it from other external sources, as you will see in more advanced courses.

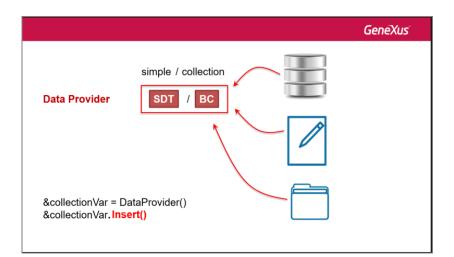
		GeneXus
Data Provider	SDT / BC	

We've also seen that a Data Provider allows loading the structure of a Business Component (and not only of an SDT) that can be of simple or collection type.

Page14

		GeneXus
Data Provider	simple / collection	

Lastly, we saw that if the structure is of collection type, we can apply methods that affect all the collection items in a single operation, such as insert() and delete().





 $\mathsf{Page}15$