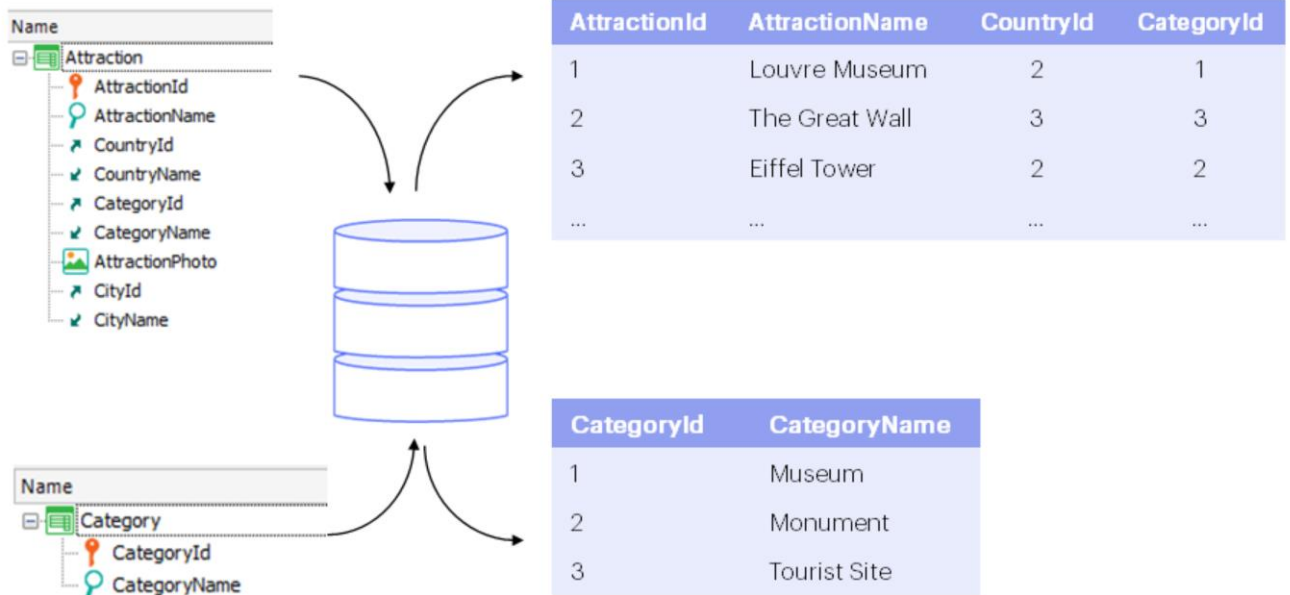


Poblar de datos las tablas desde la propia transacción

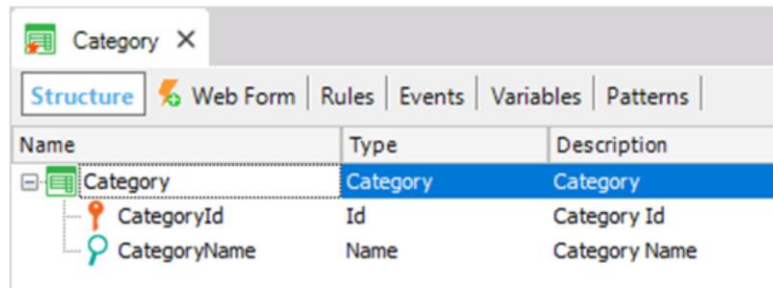
GeneXus™ 16

Transacciones y tablas



Cuando creamos una transacción, por defecto GeneXus creará tablas asociadas para almacenar la información que ingresamos a través de su pantalla.

GeneXus ofrece una solución para inicializar los datos de una transacción



Name	Type	Description
Category	Category	Category
CategoryId	Id	Category Id
CategoryName	Name	Category Name



Data	
Data Provider	False
Update Policy	Updatable

Data	
Data Provider	True
Used to	Populate data
Update Policy	Updatable

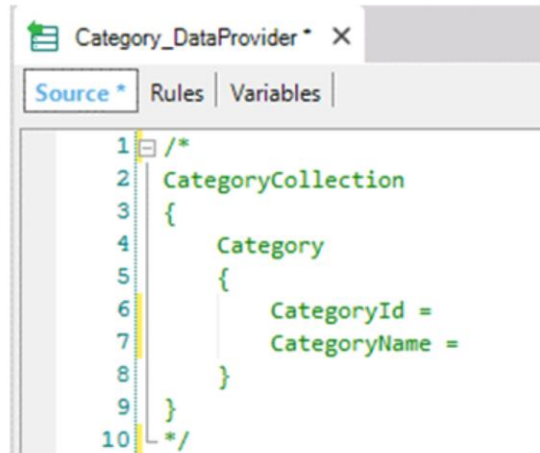
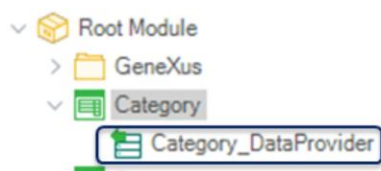
En el video anterior vimos que estas tablas podían ser inicializadas utilizando el Business Component asociado a la transacción, a través de una variable colección cargada mediante un Data Provider.

Pero GeneXus ya nos ofrece una solución para inicializar los datos correspondientes a una transacción, sin que tengamos que hacer todos los pasos anteriores manualmente (obtener el business component, crear el data provider, la variable colección, invocar el data provider, hacer el Insert).

Para ello la transacción cuenta con una propiedad, bajo el grupo Data, llamada **Data Provider**. Veámoslo con la transacción Category. Por defecto está en False... pero la pasaremos a True.

Con esto le estamos diciendo que existirá un Data Provider asociado. Y en esta nueva propiedad ... le informamos que lo usaremos para Inicializar los datos de la tabla.

Crea un Data Provider que usará el BC asociado a la transacción.



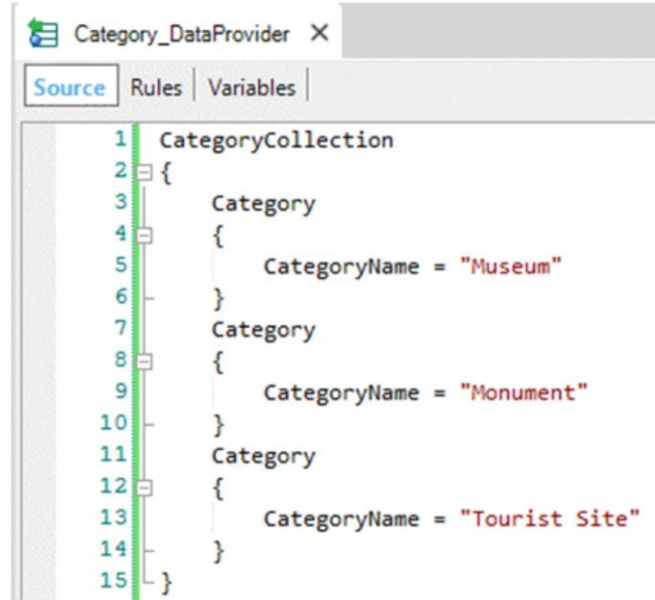
El código sugerido es casi el mismo que el código que usamos en el video previo...

Así, vemos al grabar que se creó un objeto del tipo Data Provider, al que llamó Category_DataProvider.

Además si no hubiéramos tenido en True la propiedad Business Component de la transacción, la hubiera puesto en True, haciendo que se cree entonces el business component asociado a la transacción.

Si abrimos el Data Provider podemos ver que ya nos ofrece el código para que simplemente completemos los datos de las categorías.

Reutilizamos el código que usamos para completar el DP



```
1 CategoryCollection
2 {
3     Category
4     {
5         CategoryName = "Museum"
6     }
7     Category
8     {
9         CategoryName = "Monument"
10    }
11    Category
12    {
13        CategoryName = "Tourist Site"
14    }
15 }
```

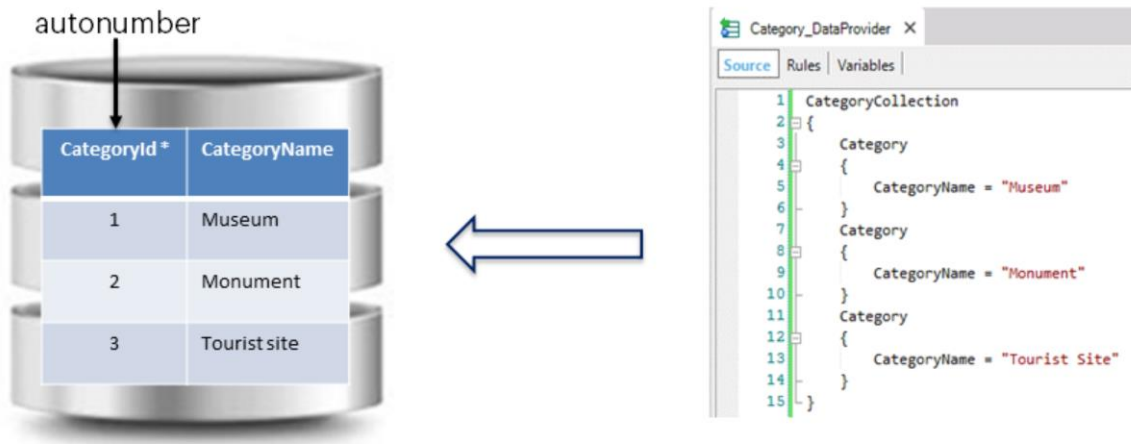
Este código es prácticamente un calco del que hicimos antes manualmente

... por lo que copiamos el código que habíamos escrito hacia el nuevo data provider.

Con esto ya hemos definido cómo el Data Provider asignará valores a las nuevas categorías que se crearán. Lo que no hemos razonado aún es el momento en el que se invocará a este Data Provider para que realice la tarea. Si lo pensamos, el momento adecuado será cuando se crea la tabla en la base de datos.

Sin embargo, para no sobrecargar el proceso de creación de las tablas, GeneXus diferirá la ejecución del Data Provider hasta el momento en que se ejecute la aplicación, que será el momento en el que verdaderamente necesitamos que estén los datos en la tabla.

GeneXus ejecuta el Data Provider cuando se crea la tabla en la BD

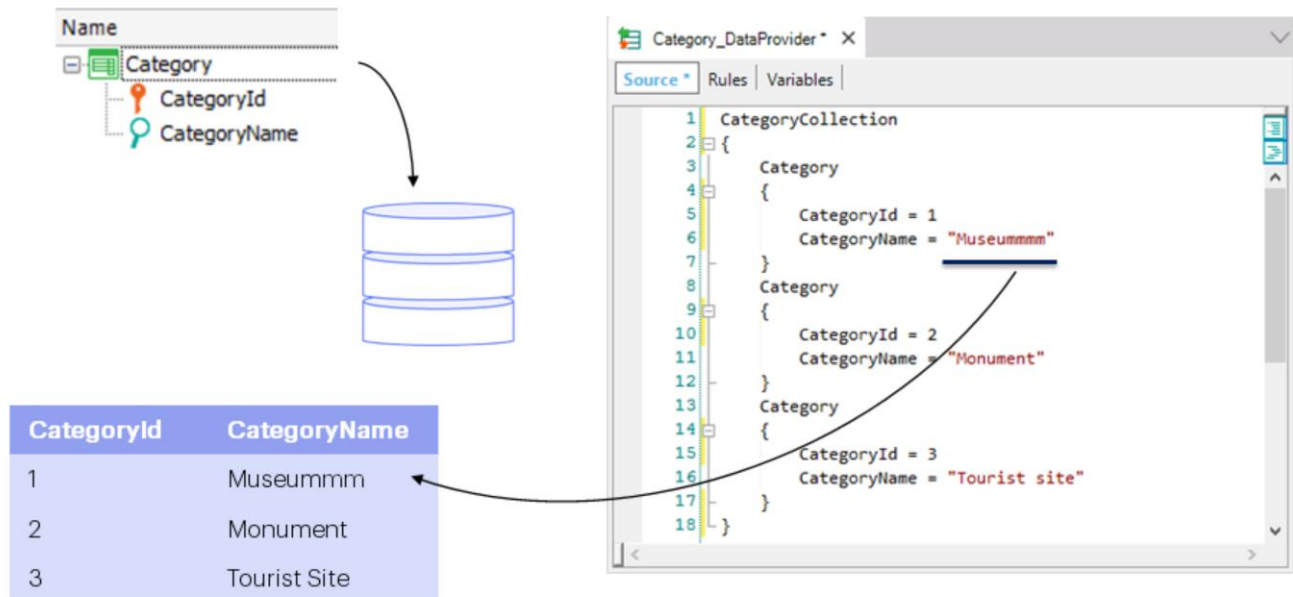


Pero en nuestro caso la tabla ya está creada y no tendrá datos porque los vamos a eliminar. Presionamos Remove Data y con eso se eliminan los datos de Category y de Attraction.

Sin embargo, como acabamos de habilitar la inicialización de los datos de la transacción, entonces en la próxima ejecución GeneXus va a proceder a ejecutar el Data Provider.

Observemos que si la tabla ya tuviera datos en ese momento, debido a que tenemos el identificador como autonumber ...lo que hará será agregar nuevos registros. Es decir, no importa si ya existía una categoría Museum, insertará otra. No es nuestro caso, ya que tuvimos la precaución de vaciar la tabla antes. En cambio, si no fuera autonumber el identificador tendríamos que indicar su valor para cada nuevo grupo del data provider ... y si ya existieran en la tabla registros con los valores que estamos agregando, se los actualizará.

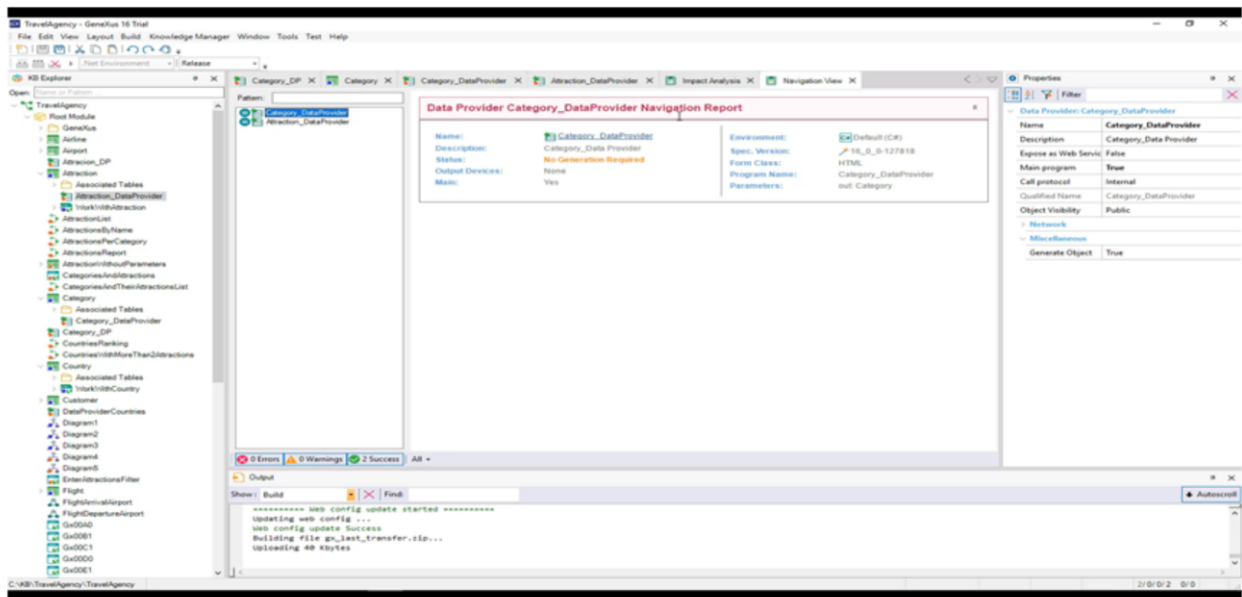
Poblando tablas



Es que al intentar hacer la inserción vía Business Component, encontrará clave duplicada y allí lo que hace es un update. Por eso, en este ejemplo, cambiaría el valor del nombre de la categoría "Museum" por esta otra con muchas emes.

Bien, para poblar con datos la transacción Attraction haremos lo mismo que hicimos antes con Category.

DEMO



[DEMO: <https://youtu.be/MhzFWrA7UIw>]

Ejecutemos lo hecho hasta ahora. F5.

Veamos que el listado de navegación informa que deberá correrse el programa de inicialización de la tabla Category. Y también de la tabla Attraction.

Y vemos en ejecución que efectivamente corrió esos programas y volvemos a tener datos en las tablas (¡observemos los identificadores! Recordemos que son autonumerados)

Ahora bien, si ahora o más adelante necesitamos modificar el Data Provider de inicialización, agregando, por ejemplo, una categoría que no habíamos contemplado inicialmente ... al hacer F5 GeneXus se dará cuenta de que cambió el data provider y volverá a ejecutarlo.

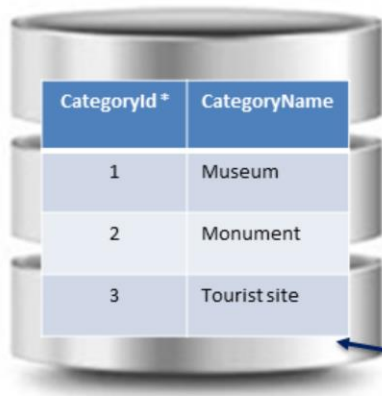
Pero al hacerlo, como ya existían las categorías Museum, Monument y Tourist site en la tabla, con ids autonumerados, entonces volverá a insertarlas con nuevos ids, además de insertar la nueva. Tenemos una forma de evitarlo, creando un índice Unique para el nombre de la categoría, para que se chequee antes de insertar el registro que no esté repetido ese valor. No lo haremos aquí.

Ejecutemos y borremos a mano los registros nuevos duplicados.

Observemos que la transacción sigue funcionando, en todo lo demás, de manera estándar. Es decir, seguiremos insertando, actualizando y eliminando sus datos a través de la pantalla, como siempre ... se ejecutarán sus reglas


...y se podrá utilizar el business component asociado, como lo hemos hecho antes. Lo que hemos visto solamente afecta a la inicialización de sus datos.

Si el DP cambia, GeneXus se da cuenta de eso y lo ejecuta nuevamente



CategoryId *	CategoryName
1	Museum
2	Monument
3	Tourist site

```
CategoryCollection
{
  Category
  {
    CategoryName = "Museum"
  }
  Category
  {
    CategoryName = "Monument"
  }
  Category
  {
    CategoryName = "Tourist site"
  }
  Category
  {
    CategoryName = "Nature reserve"
  }
}
```




























Es posible hacer que los datos de inicialización permanezcan sin cambios

Para hacer eso, configuramos la propiedad Update Policy con valor Read Only

▼ Data

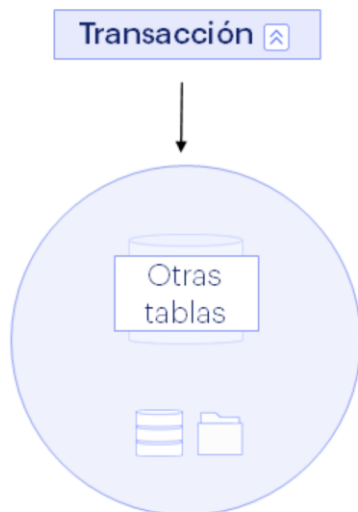
Data Provider	True
Used to	Populate data
Update Policy	Read Only

 United States
 Brazil
 Mexico
 Colombia
 Argentina
 Canada
 Peru
 Venezuela
 Chile
 Ecuador
 Guatemala
 Cuba
 Haiti
 Bolivia
 Dominican Republic
 Honduras
 Paraguay
 Nicaragua
 El Salvador
 Costa Rica
 Panama
 Puerto Rico
 Uruguay
 Jamaica
 Trinidad and Tobago

Pero si la transacción corresponde a información que no cambia con el tiempo, como por ejemplo, los países, estados o departamentos de un país, parámetros de un sistema, etcétera, resulta innecesario que la transacción o el business component nos permita actualizar sus datos. Para asegurarnos que los datos no se modifiquen, configuramos la propiedad "Update Policy" en Read Only.

Transacción Dinámica

Si los datos de la transacción se obtienen de otras fuentes, no habrá una tabla asociada.



Usos de la Transacción:

1. Insert, Update, Delete data
2. Navigate (retrieve) data

▼ Data

Data Provider	True
Used to	Retrieve data
Update Policy	Read Only

Hasta aquí el uso de la transacción corresponde más o menos al conocido, donde la transacción tiene su "tabla" asociada.

Pero tenemos otros casos en los cuales los datos de la transacción se obtienen de otras fuentes, que pueden ser consultas complejas a la base de datos que incluya ir a buscar a la información a varias tablas, o consultas a otras bases de datos, etcétera.

En este caso la transacción no tendrá esa tabla asociada. El Data Provider es quien se encarga de obtener esos datos. Para especificar este uso, a la propiedad "Used to" le configuramos el valor "Retrieve data".

A estas transacciones se les llama "transacciones dinámicas", sobre las que no ahondaremos en este curso.

En conclusión, para poblar de datos una tabla no lo haríamos en forma manual como hicimos en el video anterior, sino utilizando la propiedad Data Provider de la transacción.

Para finalizar, hagamos un Commit de nuestros cambios en GeneXus Server.



Videos

training.genexus.com

Documentation

wiki.genexus.com

Certifications

training.genexus.com/certifications