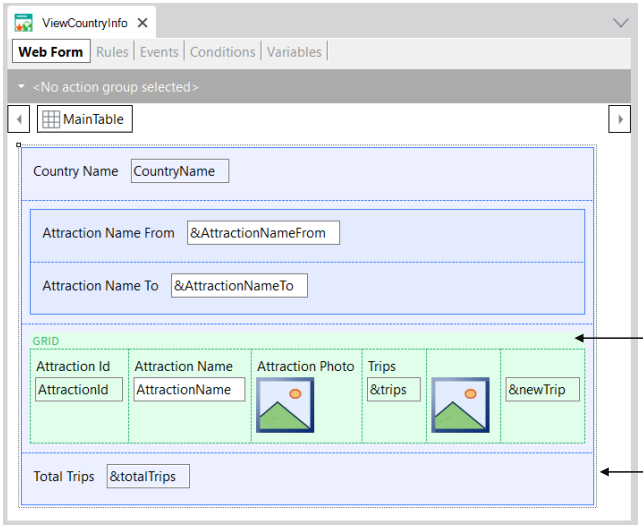
Pantallas web con foco en Back-office

Objeto Web Panel. Múltiples grids

GeneXus

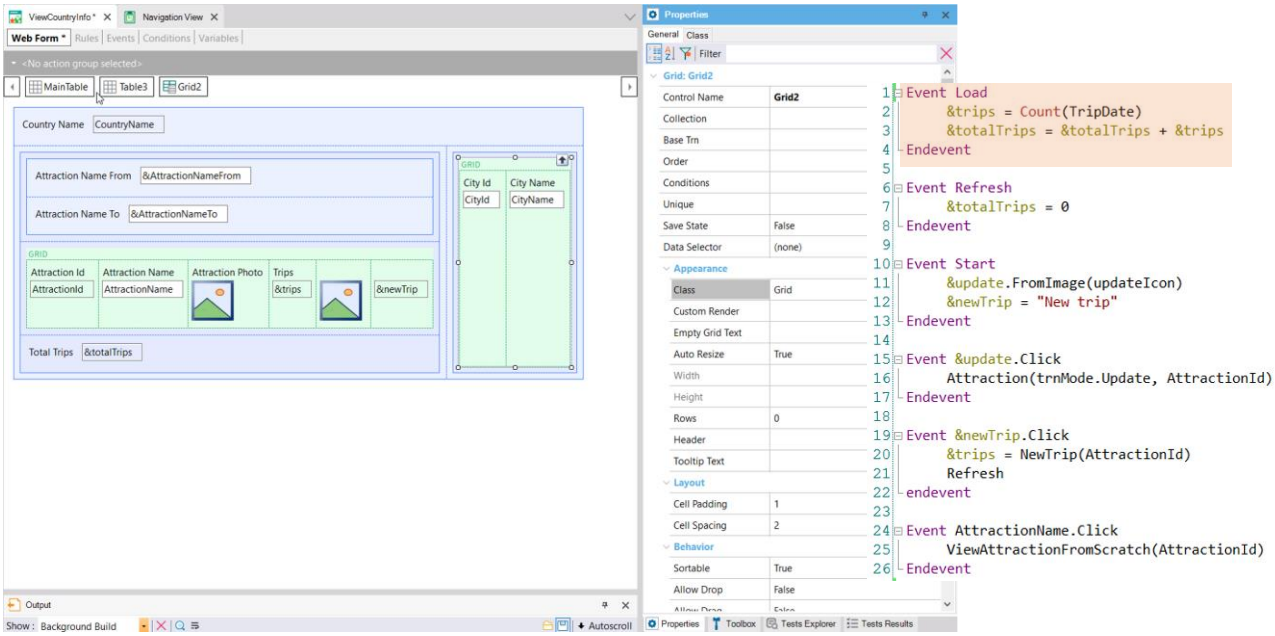
Más de un grid




```
Event Grid1.Load
     &trips = Count(TripDate)
     &totalTrips = &totalTrips + &trips
Endevent
```

Dijimos un par de veces que tal vez hubiese sido mejor utilizar el evento Load del grid y no el genérico, que solo sirve en el caso de un web panel sin grid o con un grid. Usando el Load del grid nos anticipamos al futuro, a la posibilidad de ingresar otro grid más.

Nuevo grid



Supongamos que en el web panel que muestra la información de un país (su nombre y sus atracciones turísticas), queremos agregar también un grid con sus ciudades. Antes de hacerlo, veamos que su listado de navegación indica la carga del -por ahora- único grid.
Antes de agregar el grid para las ciudades, insertemos dentro de una tabla todo lo que correspondía a las atracciones del país, para que quede toda esa información junta.

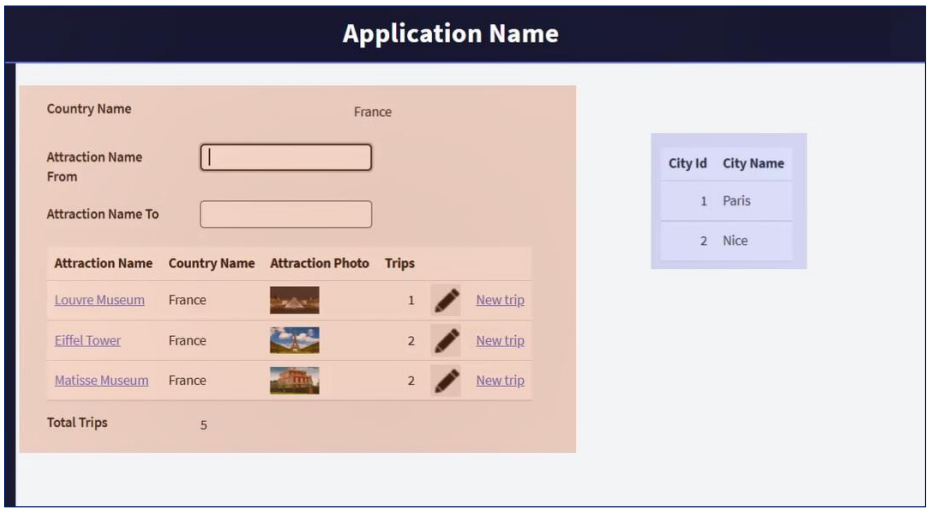
Luego insertemos otra tabla para la información de las ciudades. Allí dentro insertemos el nuevo grid, compuesto por los atributos CityId y CityName.
Si observamos sus propiedades, vemos que lo nombró por defecto como Grid2.

Cada grid podrá tener o no tabla base. En este caso ambos grids tienen atributos, así que ambos tendrán tabla base. ¿Cómo se sabe a cuál de ambos aplica el código del evento Load genérico? De hecho, si grabamos, vemos que el listado de navegación arroja un error advirtiéndolo.

Está mostrando las navegaciones que deberá realizar para cargar cada grid, y hasta entendió que la fórmula para calcular los trips debe pertenecer a la carga del Grid1, pero nos pide que precisemos esto. Y es lo que haremos.

Ahora, al grabar el objeto, el listado de navegación ya no muestra el error.

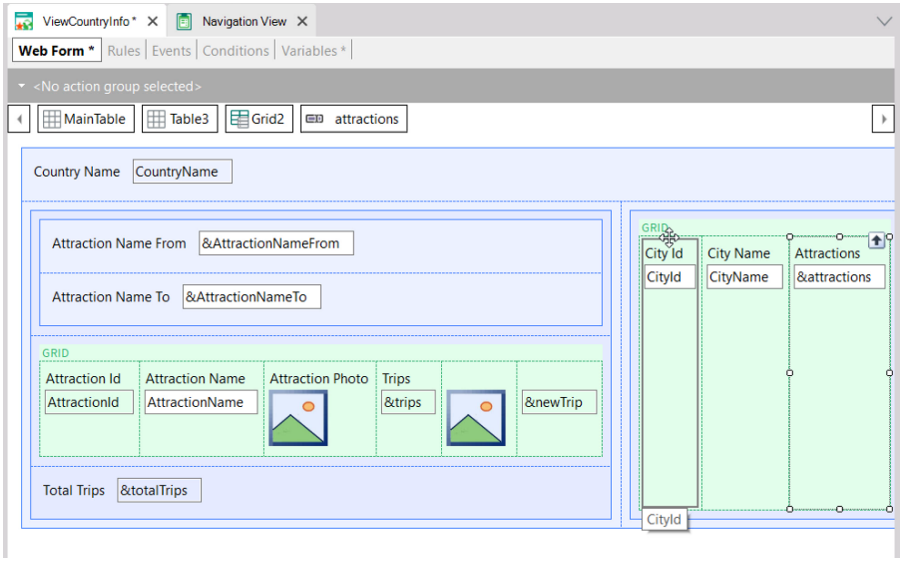
Nuevo grid



Ejecutemos.

Se trata de dos navegaciones independientes, pero como las dos tienen relación con el país recibido por parámetro, en ambas de está filtrando por país.

Nuevo grid



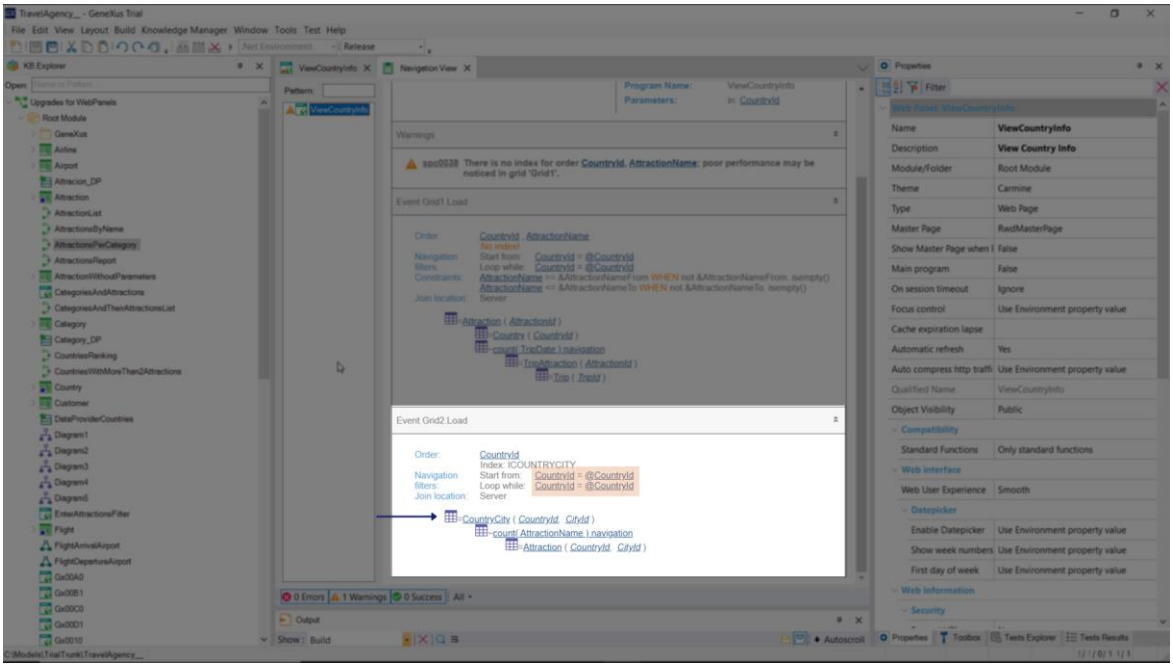

```
1  Event Grid1.Load
2       &trips = Count(TripDate)
3       &totalTrips = &totalTrips + &trips
4  Endevent
5
6  Event Grid2.Load
7       &attractions = Count(AttractionName)
8  endevent
9
10
11 Event Refresh
12      &totalTrips = 0
13 Endevent
```

Si, así como para las atracciones calculamos la cantidad de trips, para las ciudades quisiéramos calcular la cantidad de atracciones que cada una tiene... entonces agregamos una variable &attractions al grid, y la calculamos cada vez que se va a cargar una línea, esto es, en el evento Load del grid de nombre Grid2.

¿Por qué no es necesario condicionar esta fórmula para que cuente solo las atracciones del país y ciudad?

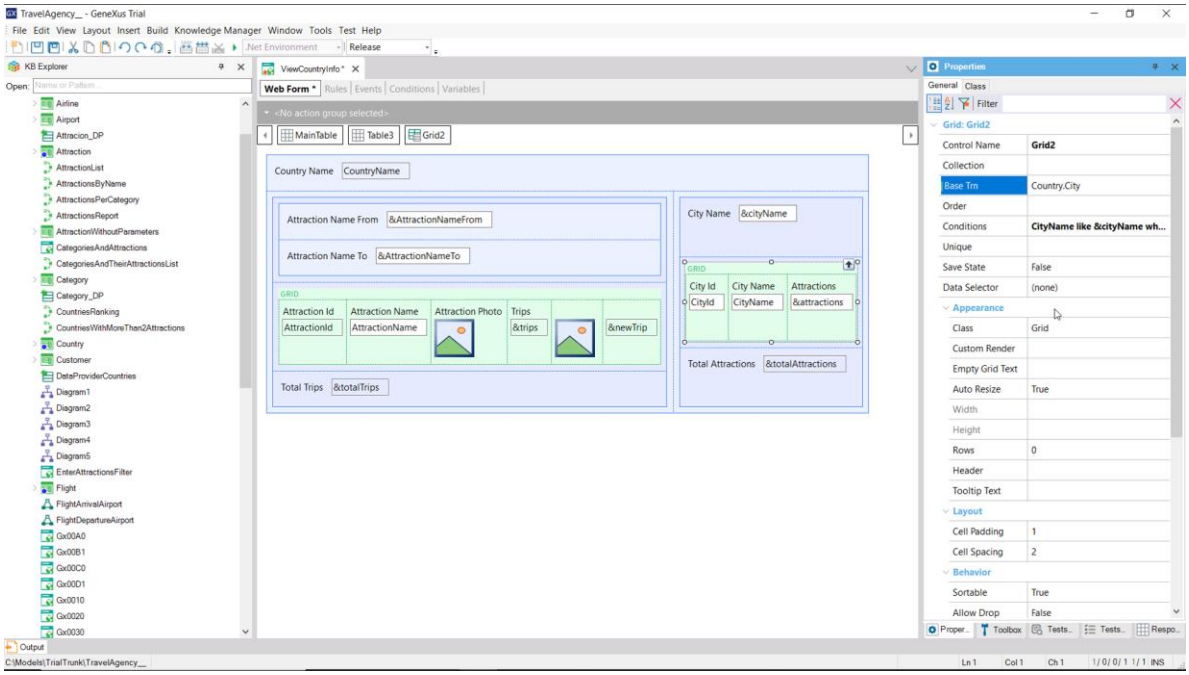
Ejecutemos.

Nuevo grid



Mientras genera, observemos el listado de navegación. Vemos que dentro del evento Load que se ejecutará cada vez que se encuentre un registro de la tabla de las ciudades que corresponda al país recibido por parámetro, se dispara el cálculo de la fórmula count sobre Attraction, filtrando por país y ciudad.

Nuevo grid



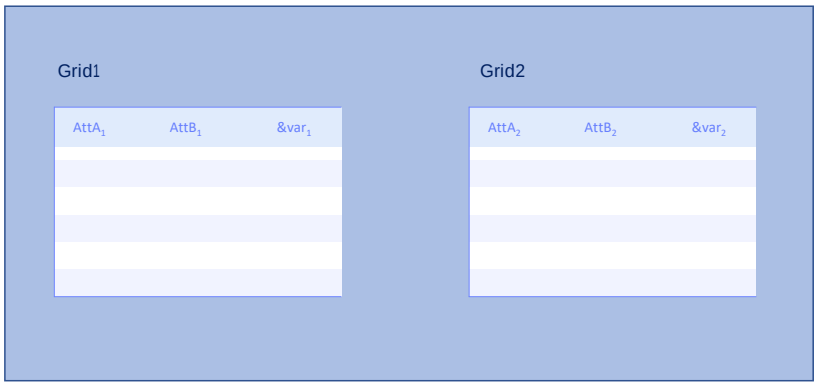
Para hacerlo funcionalmente igual al otro, podemos agregar al grid una variable para filtrar las ciudades mostradas y otra para mostrar el total de atracciones de todas las ciudades. Aquí ya lo hicimos.

Observemos que colocamos el filtro en las conditions del grid, utilizando el operador like. No indicamos transacción base y vimos que GeneXus la descubrió solo, pero nos conviene hacerlo.

Ahora bien, teníamos la inicialización de la variable &totalTrips en el evento Refresh genérico, y ahora debemos inicializar también la variable &totalAttractions.

Pero tenemos, en verdad, tres eventos Refresh: el genérico, que es el que tenemos por ahora programado, y tenemos un Refresh de cada grid.

Orden de ejecución de eventos



Start

Refresh

Grid1.Refresh

Grid1.Load

Grid2.Refresh

Grid2.Load

El orden de ejecución de los eventos al ejecutarse el web panel por primera vez será:

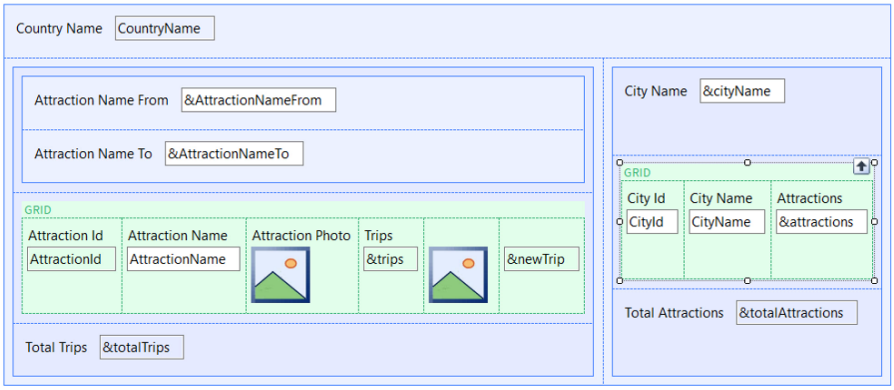
El evento Start

El evento Refresh genérico primero.

El Refresh del primer grid luego y a continuación, si tiene tabla base, se va a recorrer esa tabla filtrando los registros que correspondan, y ejecutando el Load de ese grid por cada uno. Si no tiene tabla base, entonces se ejecuta el evento Load del grid por única vez.

Y luego lo mismo con los eventos Refresh y Load del segundo grid.

Orden de ejecución de eventos




```
Event Grid1.Refresh
    &totalTrips = 0
Endevent

Event Grid2.Refresh
    &totalAttractions = 0
Endevent
```

En nuestro ejemplo, entonces, las variables &totalTrips y &totalAttractions deberían inicializarse en el Refresh de cada grid, y no en el genérico, porque luego la idea es que si cambiamos las variables de filtro de un grid, solamente se refresque lo que hace a ese grid, y no al resto de la pantalla.

Entonces cambiaríamos nuestros eventos de este modo.

¿Qué se desea refrescar?

Start

Refresh

Grid1.Refresh
Grid1.Load

Grid2.Refresh
Grid2.Load

Event ´User-event´
…
Form.Refresh
endevent

Event ´User-event´
…
Refresh
endevent

Event ´User-event´
…
Grid1.Refresh()
endevent

Por supuesto, la aparición de más grids hace que el comando Refresh que habíamos visto en otra clase programado en un evento de usuario, pueda especializarse.

Por ejemplo:

Tenemos el comando **Form.Refresh** que provocará se refresque toda la página, ejecutándose Start, Refresh genérico, Refresh y Load de cada grid.

El comando **Refresh** genérico (el que habíamos visto) provoca que se ejecuten Refresh genérico, y Refresh y Load de cada grid (es decir, todo menos el Start).

Y ahora tenemos también el método Refresh de un grid, que hará que se refresque solo el grid, es decir que se ejecuten el Refresh del grid y el Load del grid (una vez o n, dependiendo de si no tiene o sí tiene tabla base).

¿Comando Load?

Start

Refresh

Grid1.Refresh
Grid1.Load

Grid2.Refresh
Grid2.Load

Event Grid1.Load
…
Load
endevent

Event 'User-event'
…
Grid1.Load()
endevent

Para el caso del comando Load la cosa cambia un poco.

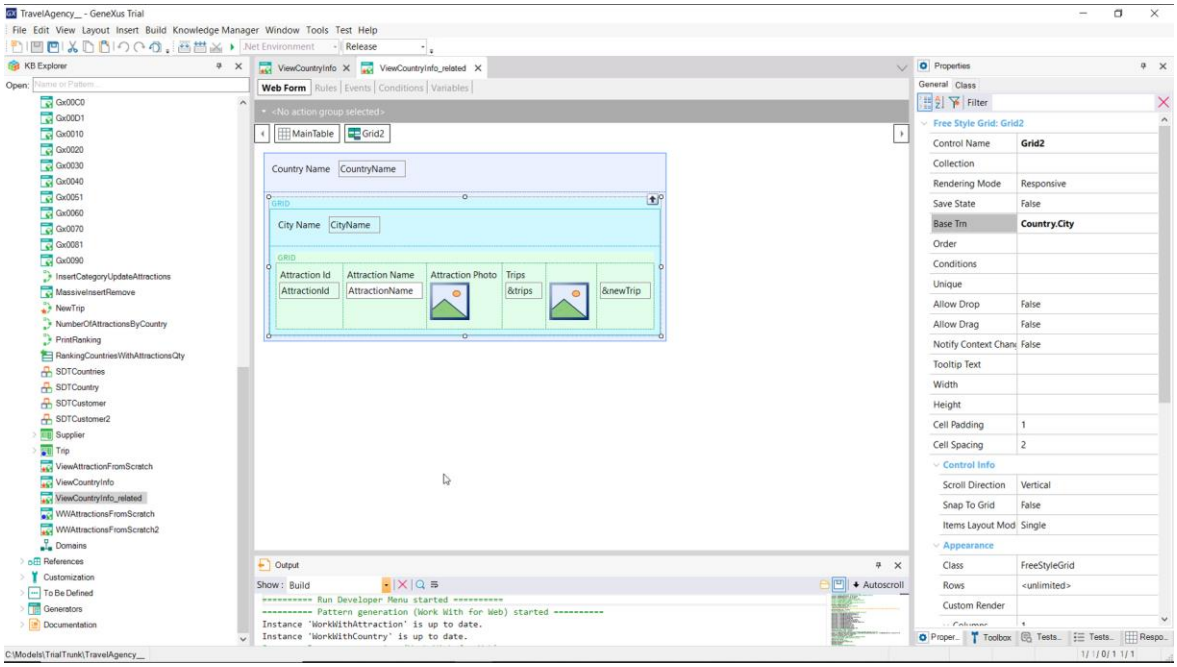
Cuando hay más de un grid, el **comando Load** a secas solo podrá escribirse dentro del **evento Load** del grid del que se trate.

Y si se quiere cargar una línea en uno de los grids desde un evento de usuario, para eso habrá que utilizar, necesariamente, el método Load del grid del que se trate.

¿Grids paralelos o anidados?

Aquí solo vimos un ejemplo de grids paralelos, pero los grids también pueden anidarse, como los for eachs.

Grids anidados: Free style grid

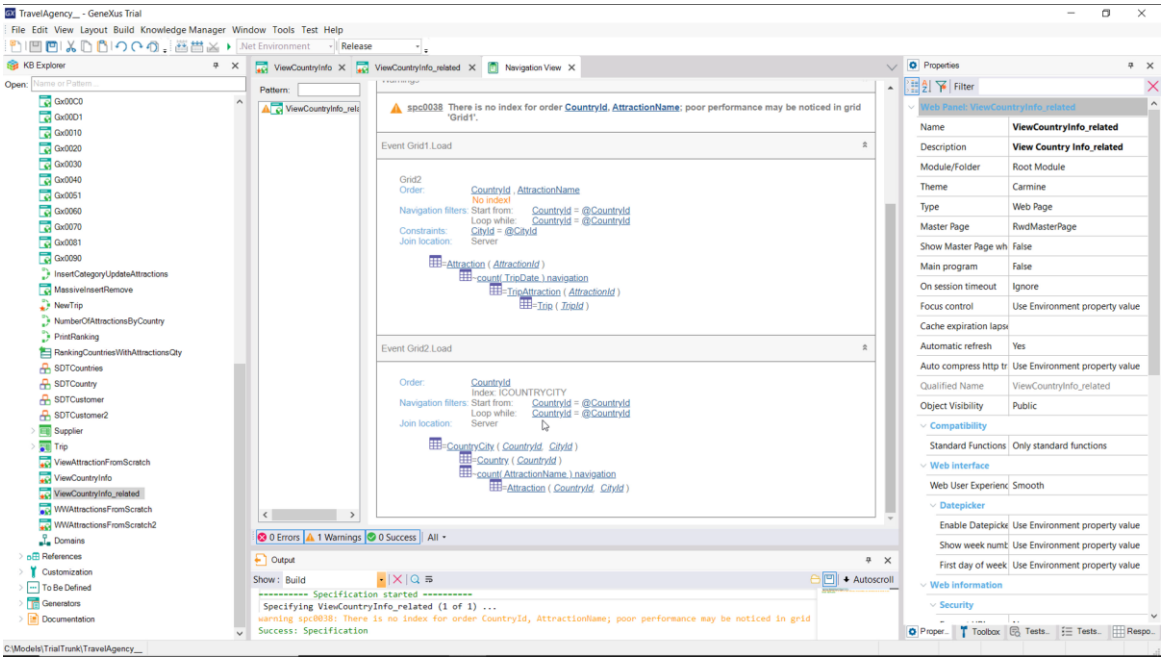


Por ejemplo, si quisiéramos mostrar al país elegido con la información que vimos antes, pero de manera relacionada.

Aquí lo implementamos. Para que un grid pueda contenter a otro, tiene que ser un tipo especial de grid, de estilo libre y no tabular. Se llama Grid Freestyle.

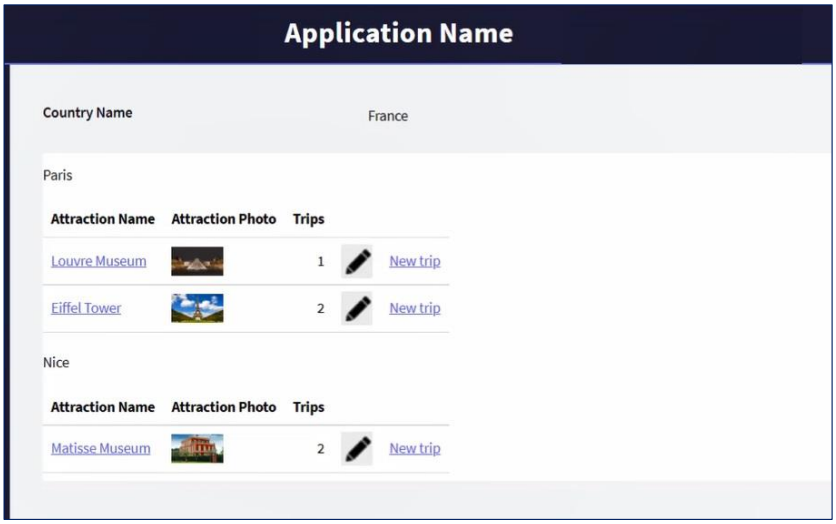
Este grid va a recorrer la tabla CountryCity y por cada ciudad encontrada, va a ejecutar el Refresh y Load del segundo grid –el grid anidado- que va a ir a buscar las atracciones de ese país-ciudad. Ahí podemos apreciar cómo la información está relacionada.

Grids anidados: Free style grid



Si observamos el listado de navegación, vemos que el Grid2 que era el de ciudades –está recorriendo esa tabla, CountryCity, filtrando por país recibido por parámetro- y luego en el Load del Grid1 que es el que corresponde a las atracciones, se está recorriendo la tabla de atracciones filtrando por el país y la ciudad en la que se está posicionado en cada registro de CountryCity y por eso está apareciendo este arroba de aquí por CityId.

Grids anidados: Free style grid



Si ejecutamos. Aquí lo vemos. París y sus dos atracciones. Niza y su atracción.

¿Grids paralelos o anidados?

Hay mucho más para investigar sobre este tema. Aquí nos conformamos con esta introducción.

GeneXus™

training.genexus.com
wiki.genexus.com