Web Panels

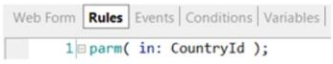
Cómo implementar cortes de control en Grids anidados

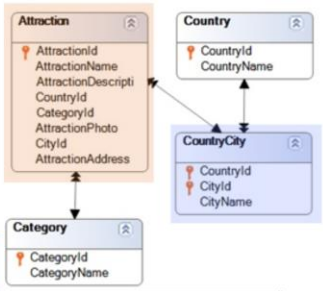
GeneXus™

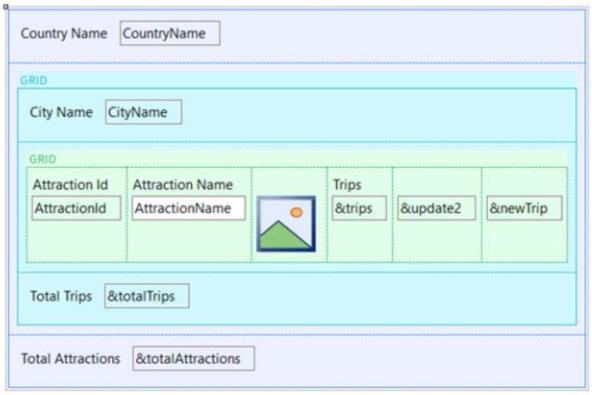
Web Panel with SEVERAL Grids

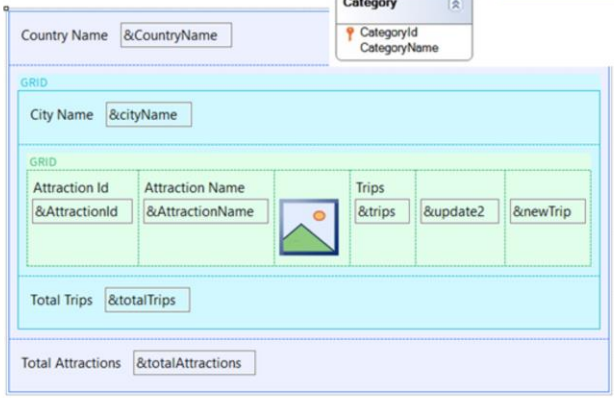
En otro video estudiamos cómo se determinaban las tablas base y la navegación de un web panel con varios grids.

With several Grids: nested









Y en particular vimos un ejemplo de grids anidados que realizaban un **join**.

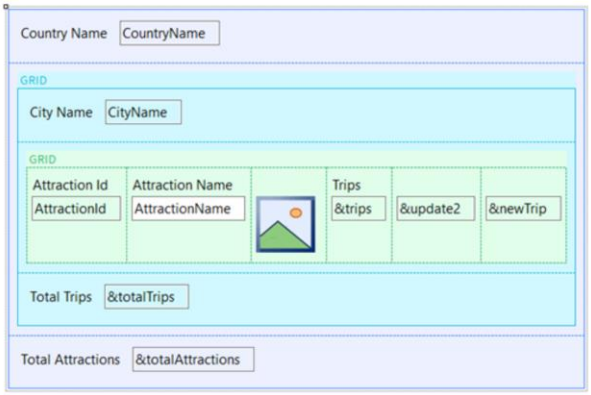
Es decir, el grid externo recorría una tabla con una relación 1 a N con la tabla que recorría el grid interno, ya fuera que esos grids se implementaran con o sin tabla base.

En ambas soluciones el web panel recibía por parámetro un identificador de país, y en el grid externo se mostraban las ciudades de ese país, es decir, iba a recorrer CountryCity; y en el interno se mostraban las atracciones turísticas de esa ciudad. Es decir, iba a recorrer Attraction.

With several Grids: nested

Web Form | **Rules** | Events | Conditions | Variables

```
1 parm( in: CountryId );
```



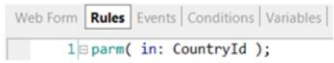
Grid1 and Grid2 with Base Tables

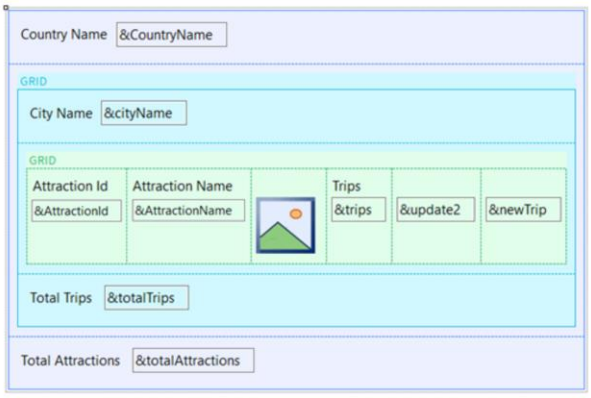
Pseudo-code

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

For each Country.City
where CountryId = &CountryId

     Event Grid1.Load
          &attractions = Count(AttractionName)
          &totalAttractions = &totalAttractions + &attractions
     endevent

     Load

     Event Grid2.Refresh
          &totalTrips = 0
     Endevent

     For each Attraction order AttractionName
     where CountryId = &CountryId
     where CityId = &CityId

          Event Grid2.Load
               &trips = Count(TripDate)
               &totalTrips = &totalTrips + &trips
          Endevent

          Load

     endfor

endfor
```

Para el caso de grids con tabla base, ese join se establecía automáticamente, sin tener que hacer nada. GeneXus lo detectaba y agregaba el filtro en su programa fuente.

With several Grids: nested



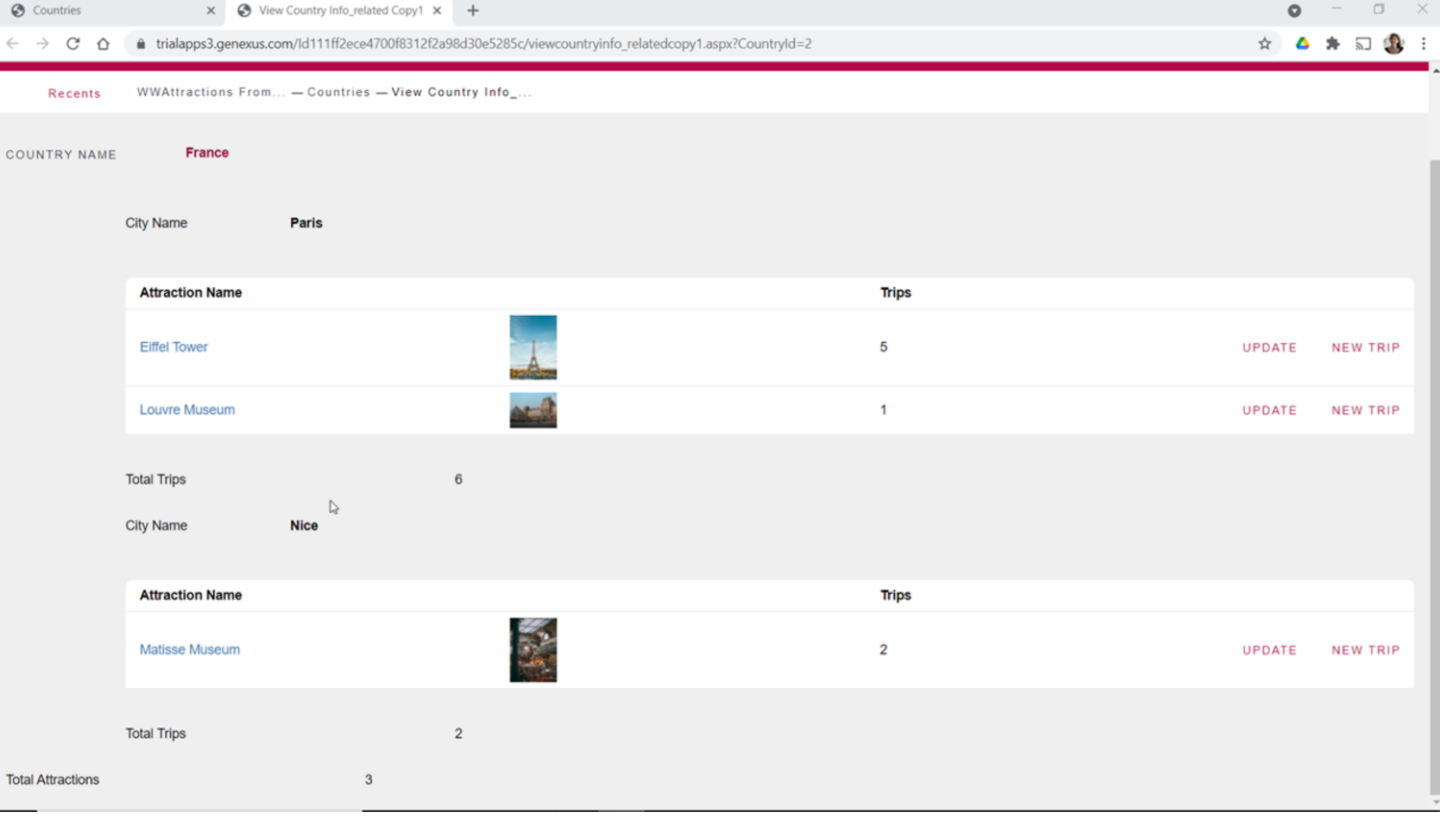


Grid1 and Grid2 **without** Base Tables

Pseudo-code

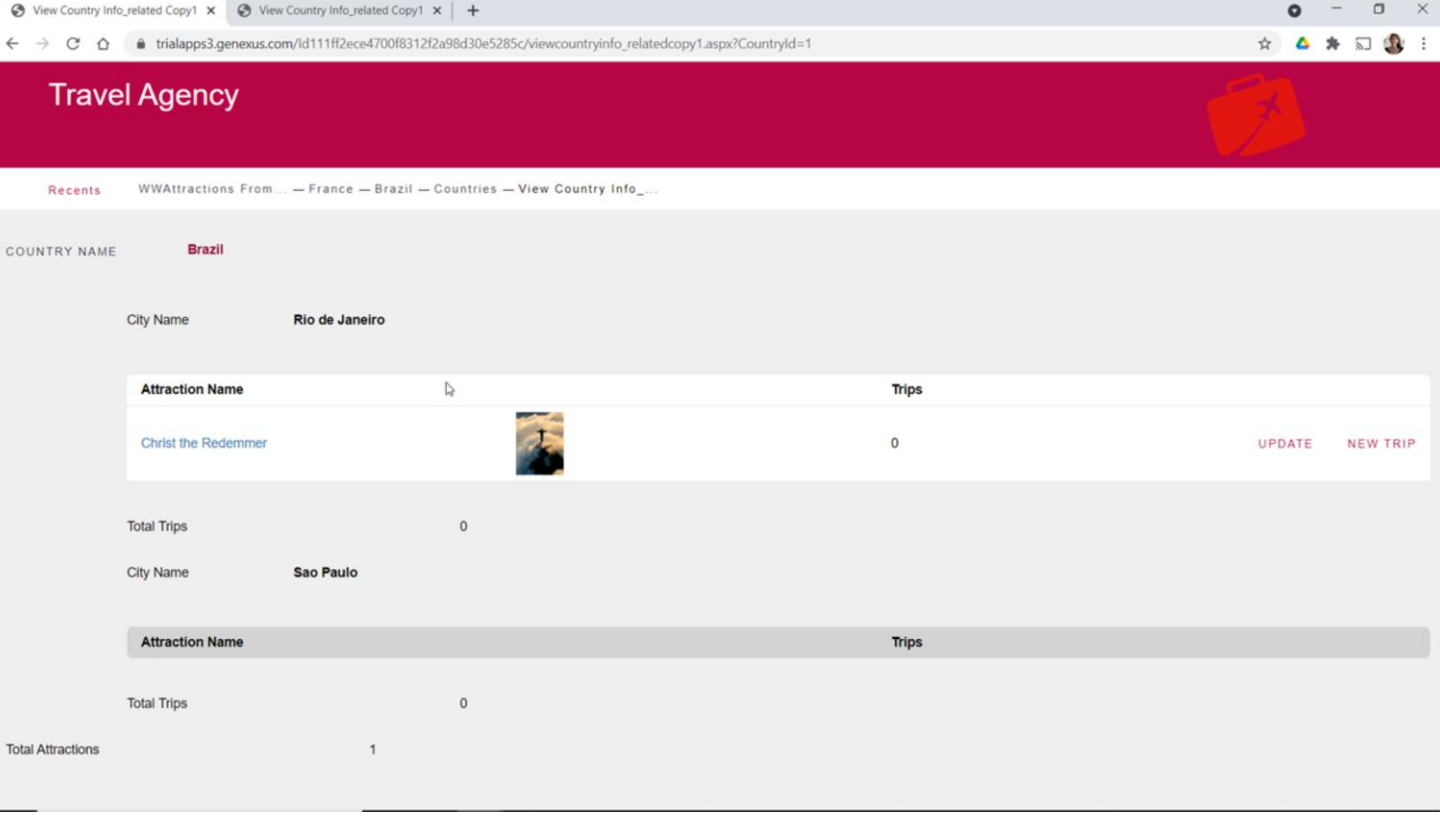
```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid1.Load
     For each Country.City
          &CountryName = CountryName
          &cityName = CityName
          &attractions = Count(AttractionName)
          &totalAttractions = &totalAttractions + &attractions
          Load
     endfor
endevent

     Event Grid2.Refresh
          &totalTrips = 0
     Endevent

     Event Grid2.Load
          For each Attraction order AttractionName
          where CityName = &cityName
               &AttractionId = AttractionId
               &AttractionName = AttractionName
               &AttractionPhoto = AttractionPhoto
               &trips = Count(TripDate)
               &totalTrips = &totalTrips + &trips
               Load
          endfor
     Endevent

     endfor
```

En cambio, para el caso de grids sin tabla base, teníamos que especificarlo nosotros en el for each que implementábamos para cargar el grid anidado (y solo filtrábamos por ciudad, porque el filtro por país ya se encontraba implícito por recibirlo por parámetro en el atributo CountryId).



Aquí tenemos el web panel con ambos grids **con tabla base**. Hemos agregado una acción en el pattern Work With de países, para invocar a este web panel.

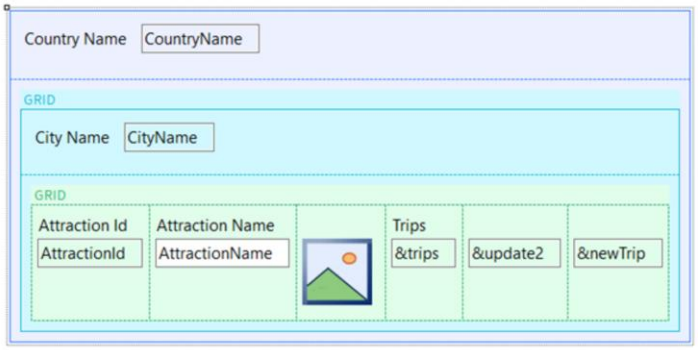
Si elegimos Francia: aquí vemos las atracciones de París y las de Niza, que son las dos ciudades que tenemos ingresadas en el sistema.

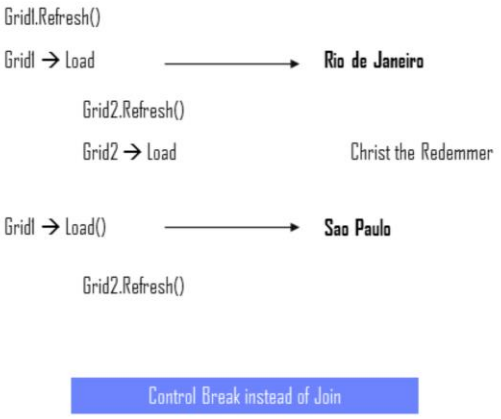


Ahora, veamos qué pasa si en lugar de elegir Francia elegimos Brasil, por ejemplo, que tiene también dos ciudades ingresadas.

Vemos que para la primera, Rio de Janeiro, se muestra una atracción turística, pero para la segunda, Sao Paulo, no se muestra ninguna. Esto es así, justamente, por tratarse de un **join**.

With several Grids: nested



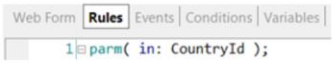


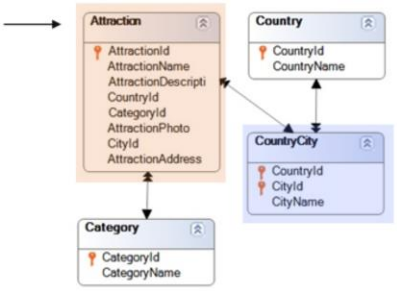
Ya que primero se ejecutará el Refresh del grid externo, y luego, posicionados en la primera ciudad, Rio de Janeiro, se la cargará en el GridI, e inmediatamente se ejecutará el Refresh del grid anidado, y a continuación se cargarán las atracciones de Rio de Janeiro, que en este caso es solo una, el Cristo Redentor.

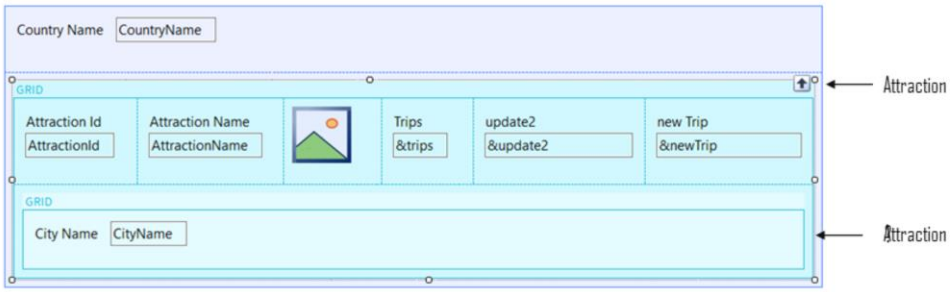
Luego se pasará a la siguiente ciudad, Sao Paulo, se la cargará y se hará el Refresh del grid anidado. Pero cuando se vaya a recorrer la tabla de atracciones para cargar únicamente las de Sao Paulo, no se encontrará ninguna.

Para que solo se muestren las ciudades con atracciones, necesitamos implementar un **corte de control** y no un **join**.

With several Grids: nested



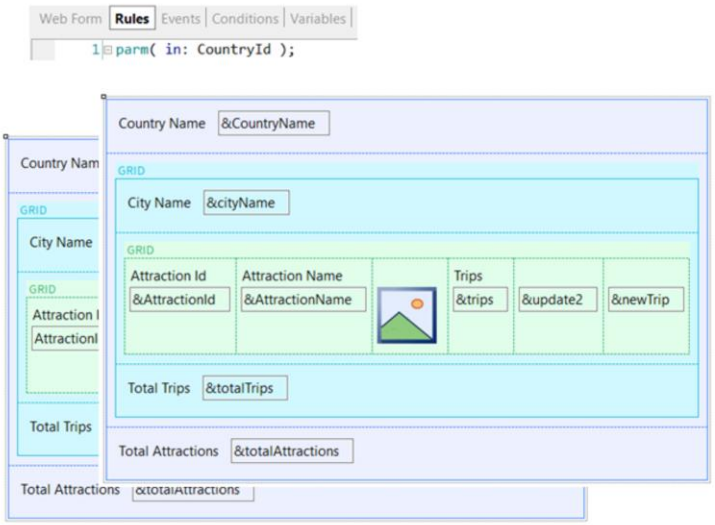


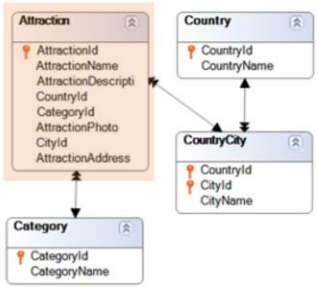


Habíamos dejado esbozado un caso donde se produciría un corte de control indeseado, en lugar de un join. Pero solo habíamos atendido a ese efecto a partir de no especificar transacción base para el grid anidado. Que parecía ser CountryCity, pero en verdad sería Attraction. Pero no nos habíamos adentrado en la navegación.

Ahora lo haremos, con nuestro ejemplo.

With several Grids: nested






```
For each Attraction order CountryId, CityId

   print CountryName
   print CityName

   For each Attraction
      print AttractionName, AttractionPhoto, etc.
   endfor

endfor
```

Necesitamos, como para el caso de un listado, que ambos for eachs (sean implícitos, es decir, provengan de grids **con** tabla base, o sean explícitos, es decir, provengan de grids **sin** tabla base) tengan la misma tabla base, Attraction. Y que el primero conforme el grupo que va a realizar el corte por país/ciudad.

Para ello, entonces, alcanza con modificar la **transacción base** y agregar el **order** al primer grid.

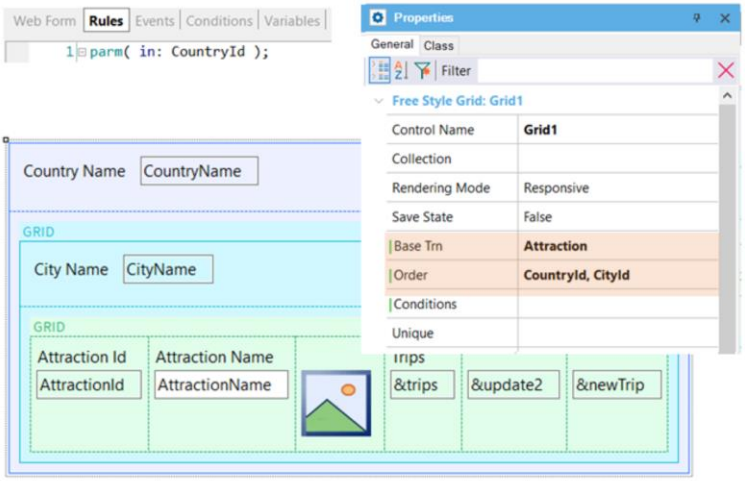
Dependerá de si se implementaron los grids con o sin tabla base, para ver cómo hacerlo.

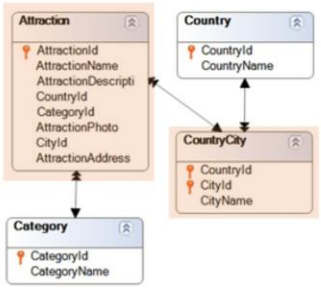
WITH Base Tables

Empecemos por el caso en el que el web panel se implementó con ambos grids con tabla base

With several Grids: nested

Grid1 and Grid2 with Base Tables

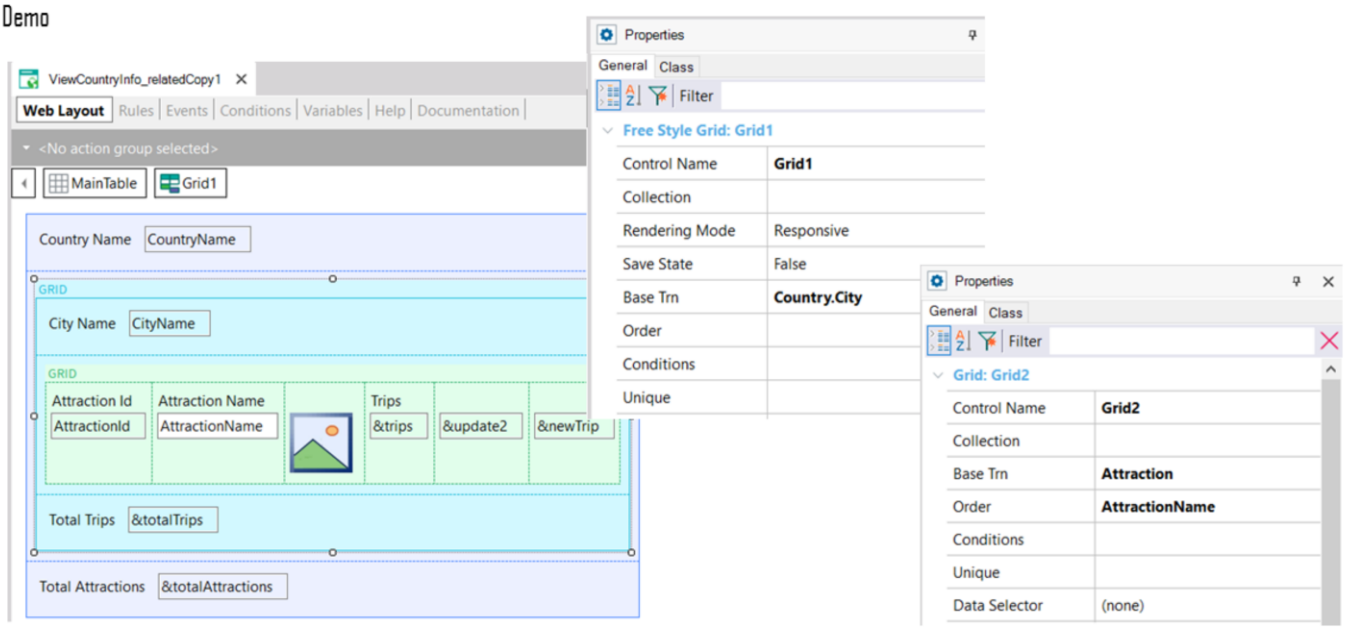





```
For each Attraction order CountryId, CityId

   print CountryName
   print CityName

   For each Attraction
      print AttractionName, AttractionPhoto, etc.
   endfor

endfor
```

En este caso el objetivo se consigue modificando las propiedades que vemos del grid externo, por estas otras.

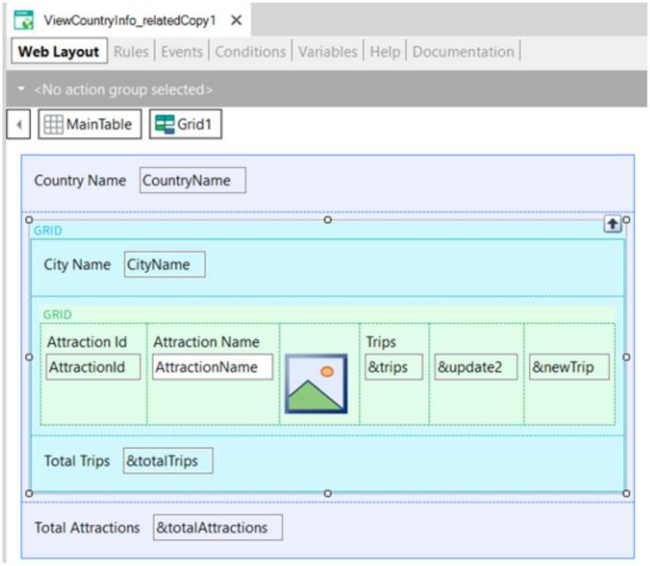
Demo



Antes de hacerlo, vayamos a GeneXus. Vemos las propiedades de ambos grids: el primero tiene Base Trn Country.City, y ningún order.

El segundo tiene Base Trn Attraction y un order por AttractionName.

Demo

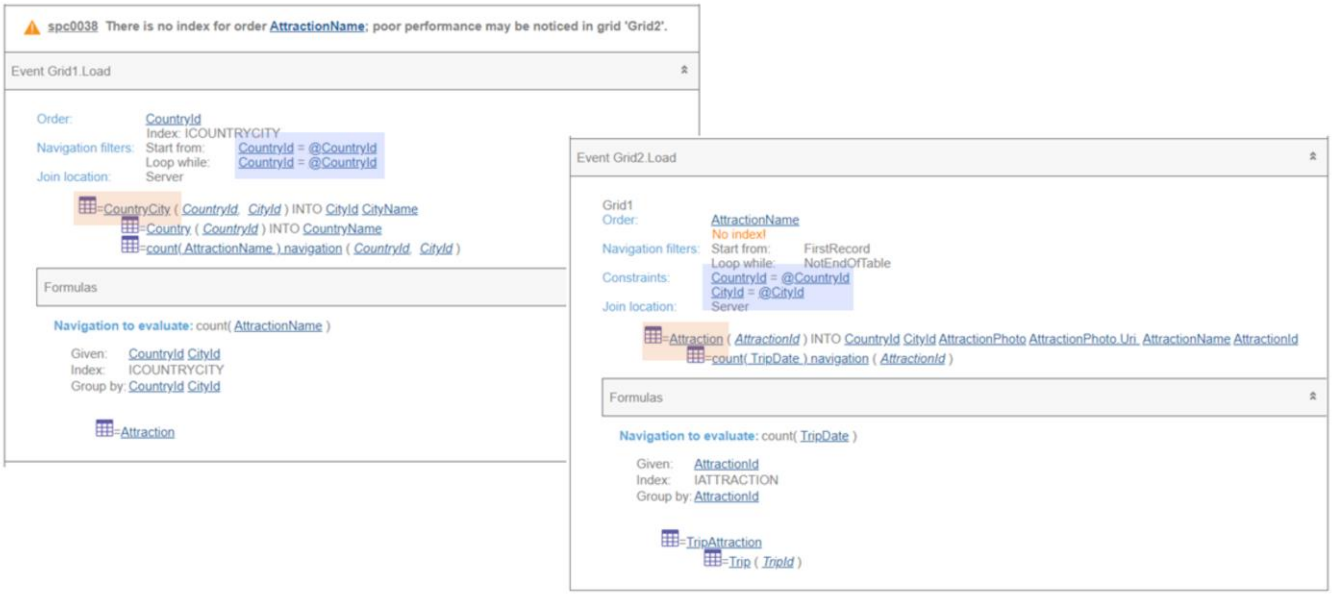



```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid2.Refresh
     &totalTrips = 0
Endevent

Event Grid1.Load
     &attractions = Count(AttractionName)
     &totalAttractions = &totalAttractions + &attractions
Endevent

Event Grid2.Load
     &trips = Count(TripDate)
     &totalTrips = &totalTrips + &trips
Endevent

Event &update2.Click
     Attraction(trnMode.Update, AttractionId)
Endevent

Event &newTrip.Click
     &trips = NewTrip(AttractionId)
     Refresh
endevent

Event AttractionName.Click
     ViewAttractionFromScratch(AttractionId)
Endevent
```

Y en los eventos vemos que se han programado los Refresh y Load de cada grid, únicamente para incializar y sumarizar o contar en variables (las que mostrarán el total de atracciones cargadas y el total de excursiones –trips– en los que esas atracciones se encuentran).

Después tenemos eventos de usuario para llamar a diversos objetos. Aquí no importan para nada.

Demo



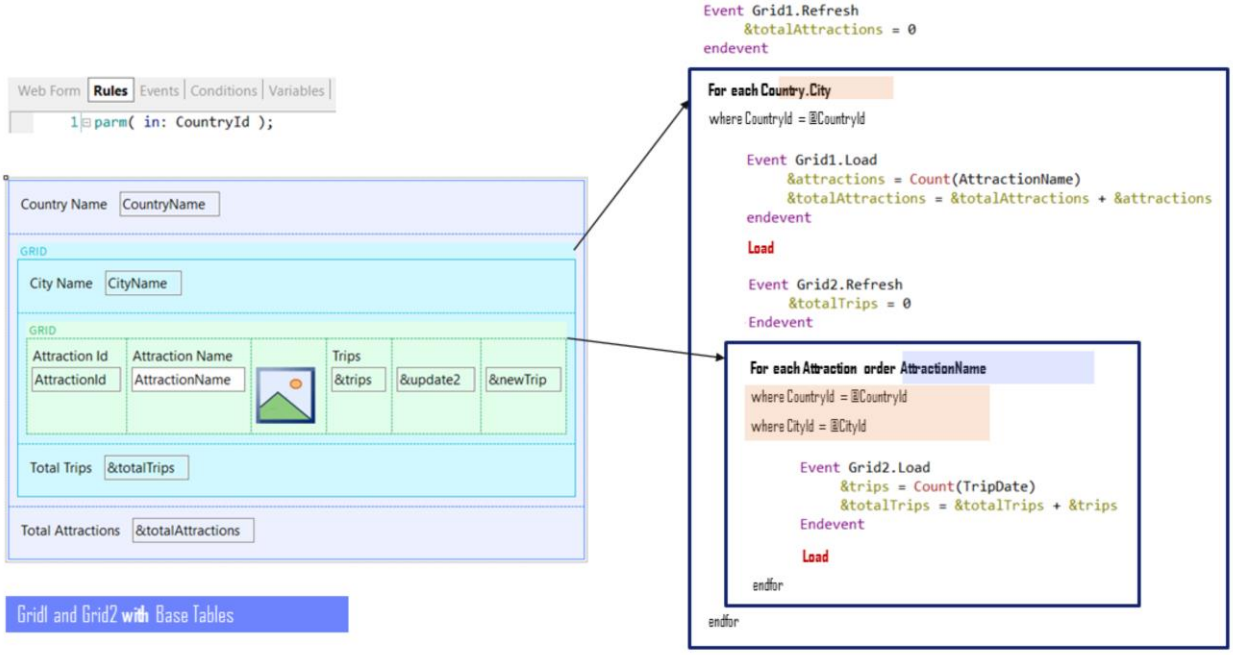
Observemos en GeneXus el listado de navegación del web panel.

Vemos claramente la tabla base del primer grid: CountryCity, y que va a filtrar por el país recibido por parámetro. Y luego vamos al Load anidado, que tiene como tabla base Attraction, y filtra por el país y la ciudad del grid exterior.

Vemos, claro, el join.

Observemos, además, que ordena el primer For each implícito por CountryId, y el segundo por AttractionName, para el que informa que no hay un índice.

With several Grids: nested



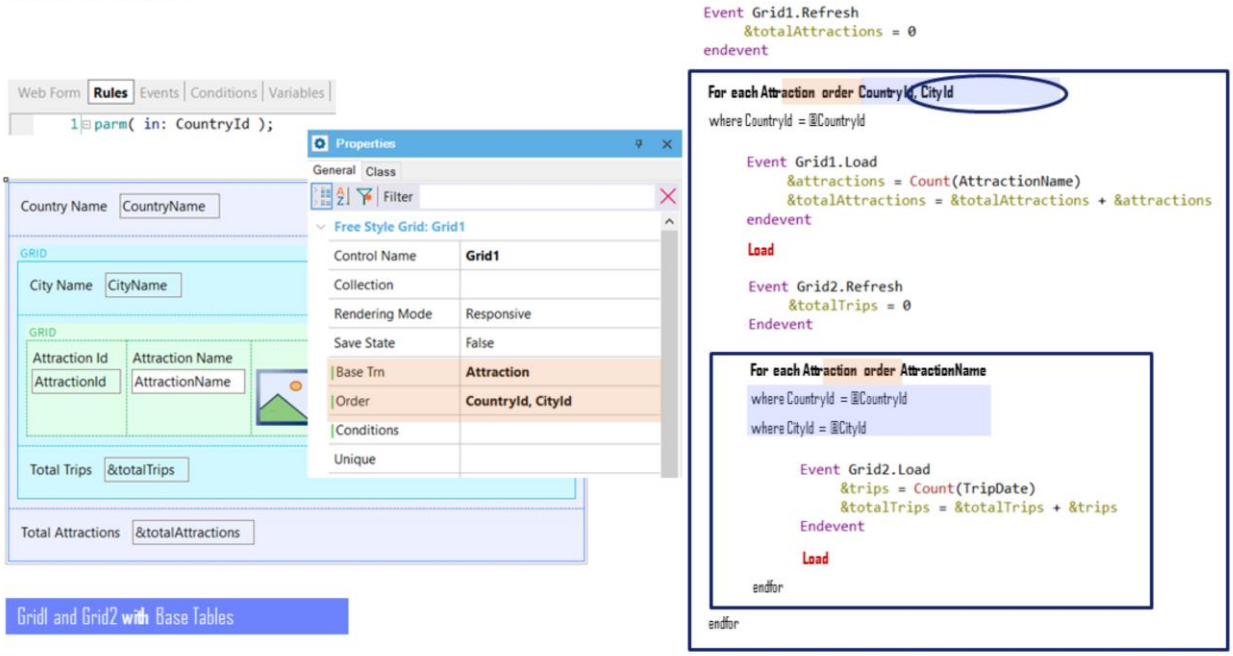
Grid1 and Grid2 with Base Tables

El pseudo-código del fuente que programará GeneXus será más o menos como vemos. Donde la propiedad Base Transaction del Grid1 se utilizó para programar la base Transaction del For each implícito.

En el segundo for each se colocó como cláusula order lo que había en la propiedad Order del grid, que era AttractionName, y por eso vimos los índices elegidos que vimos.

En definitiva, se va a disparar el Refresh del grid externo y a continuación el For each implícito que estamos viendo, que con el interno conformarán **un join**.

With several Grids: nested



Grid1 and Grid2 with Base Tables

Ahora sí, modifiquemos las propiedades del Grid1, el externo.

Al hacerlo, la propiedad Base Transaction del grid hará que se modifique la Base Transaction del For each implícito. Y la propiedad Order se convertirá en la cláusula order de ese for each. Con eso ya alcanzará.

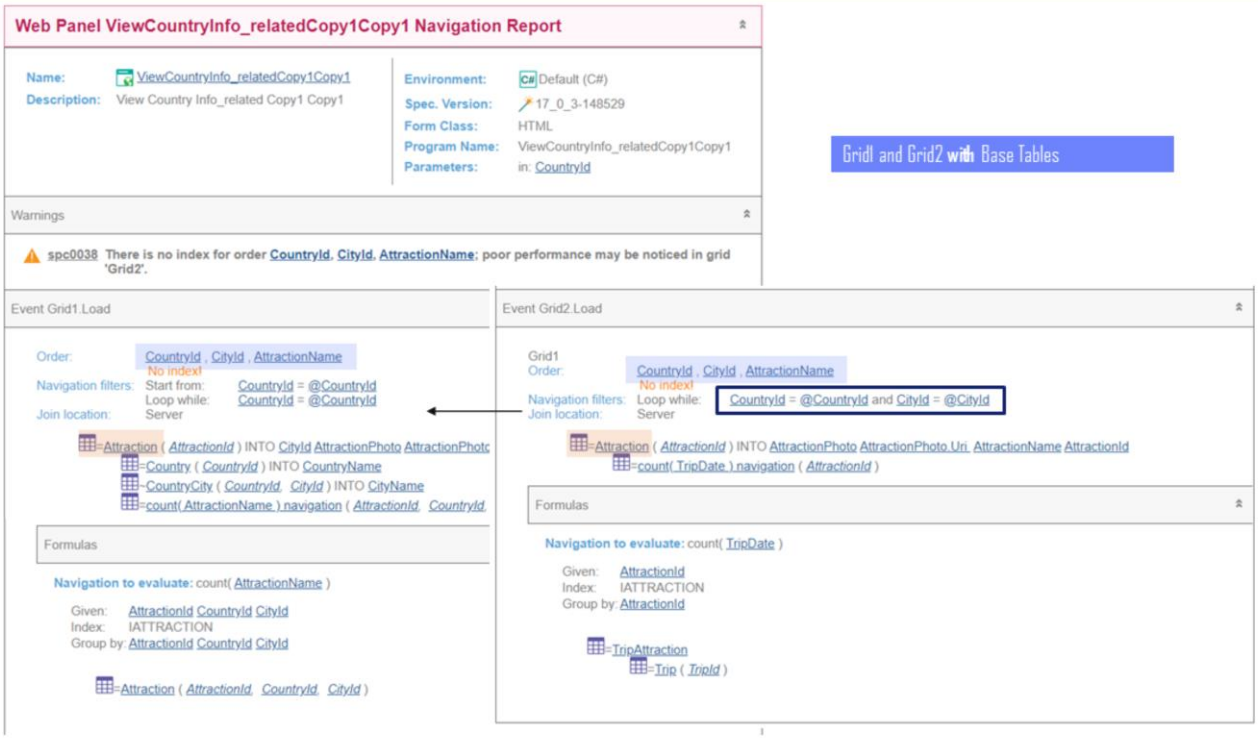
Porque se va a disparar el Refresh del grid externo y a continuación el for each implícito que estamos viendo, que con el interno conformarán un **corte de control**. Por tanto, se van a agrupar las atracciones turísticas por ciudades del país (el filtro por CountryId se debe al parámetro).

Por tanto, por **cada grupo** de atracciones turísticas conformado por cada ciudad se va a disparar una vez el evento Load. Luego se va a cargar en el grid la primera línea, con el CityName de la ciudad del primer grupo de atracciones (y por supuesto, su CountryName, que será, para todos, el mismo).

Y luego se va a disparar el evento Refresh del grid anidado, a continuación del cual se ejecutará el for each implícito, que va a recorrer únicamente las atracciones del grupo, es decir, las de ese país y ciudad. Por cada una de esas atracciones del grupo va a disparar el evento Load y el comando Load para cargarla en el grid anidado.

Y así sucesivamente con todos los grupos.

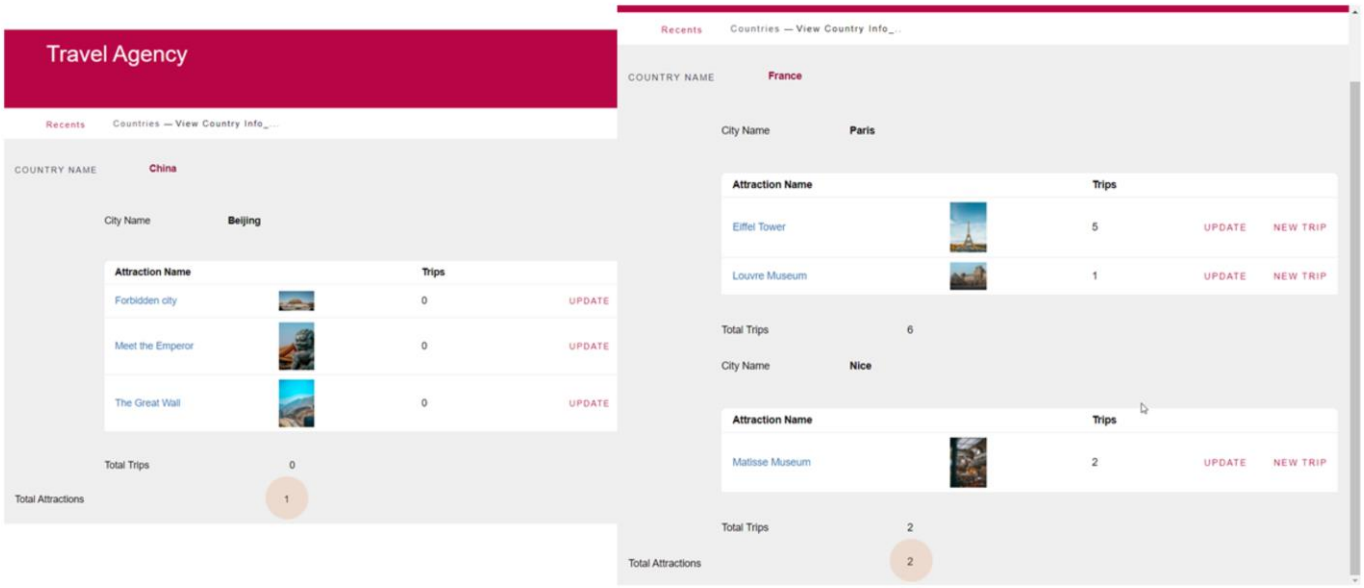
Demo



Si observamos el listado de navegación, vemos claramente que ambos grids navegarán la misma tabla, Attraction, utilizando un índice compuesto por los atributos de la propiedad Order del primer grid, más el order del grid anidado.

Y vemos claramente que para el grid anidado solamente se recorrerán las atracciones que correspondan al país y ciudad del primer grid.

Demo



Si lo probamos ahora en ejecución... vemos exactamente lo que queríamos.

Observemos por ejemplo, China. Perfecto.

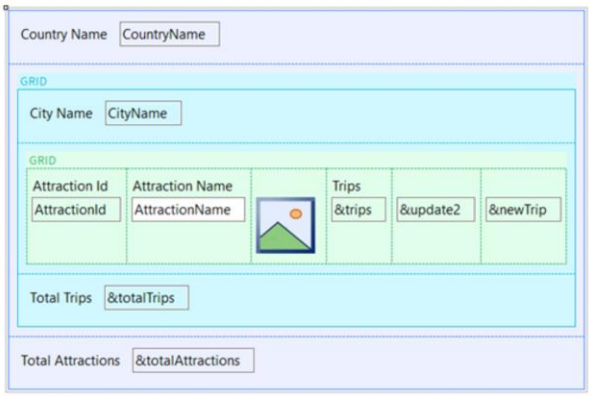
Y si vamos a Francia... en este caso no notamos ninguna diferencia con el caso de join.

Pero las atracciones no se están contando bien. ¿Por qué?

With several Grids: nested

Web Form | **Rules** | Events | Conditions | Variables

```
1 parm( in: CountryId );
```



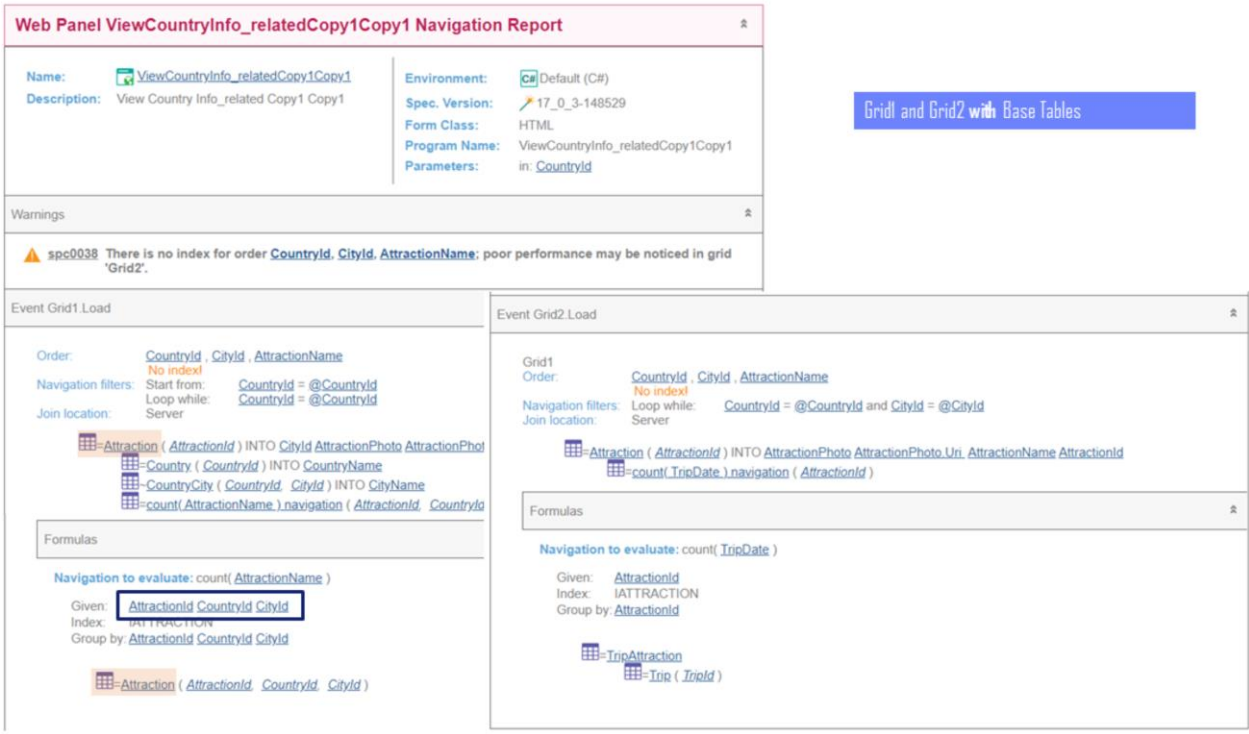
Grid1 and Grid2 with Base Tables

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent
```

```
For each Attraction order CountryId, CityId
where CountryId = &CountryId

     Event Grid1.Load
          &attractions = Count(AttractionName)
          &totalAttractions = &totalAttractions + &attractions
     endevent

     Load

     Event Grid2.Refresh
          &totalTrips = 0
     Endevent

     For each Attraction order AttractionName
     where CountryId = &CountryId
     where CityId = &CityId

          Event Grid2.Load
               &trips = Count(TripDate)
               &totalTrips = &totalTrips + &trips
          Endevent

          Load

     endfor

endfor
```

Es que en el Grid1 estamos utilizando la fórmula Count para contar las atracciones que corresponden a ese país y ciudad. Esto funcionaba cuando la tabla base del for each era CountryCity, pero no ahora que es Attraction.

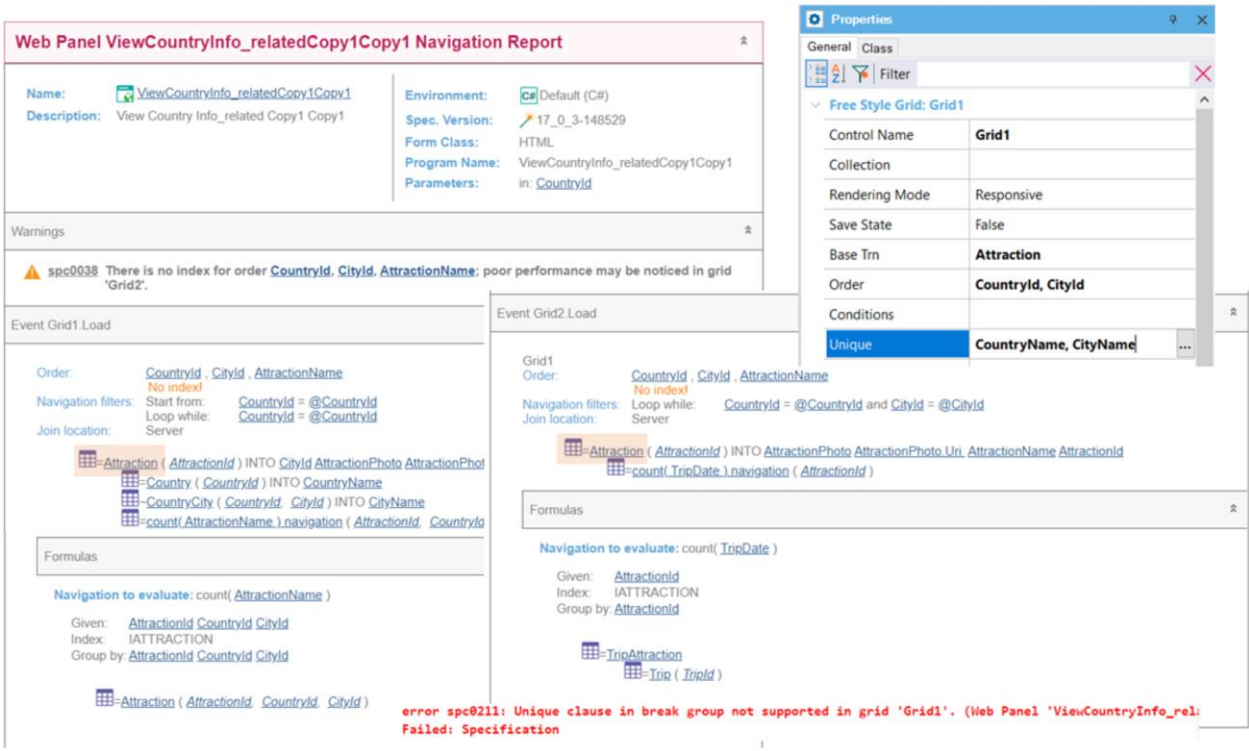
Demo



El problema con la fórmula del primer grid se ve claro en el listado de navegación.

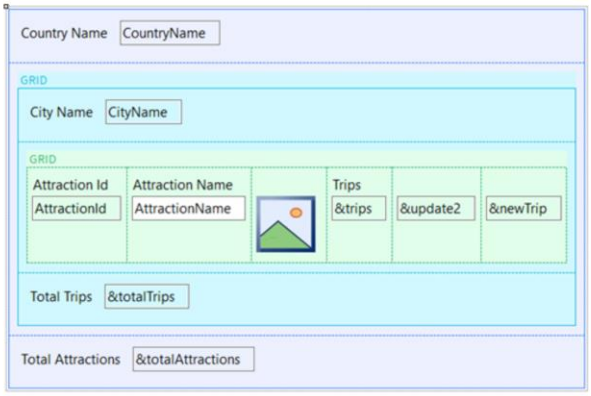
No podemos recorrer una tabla y realizar una agregación sobre la misma tabla.

Demo



Para que eso nos funcione tendríamos que utilizar la **cláusula unique**, que en este caso no nos sirve, porque no se soporta para cortes de control (nos referimos al corte de control entre el grid 1 y el grid 2).

With several Grids: nested



Grid1 and Grid2 with Base Tables

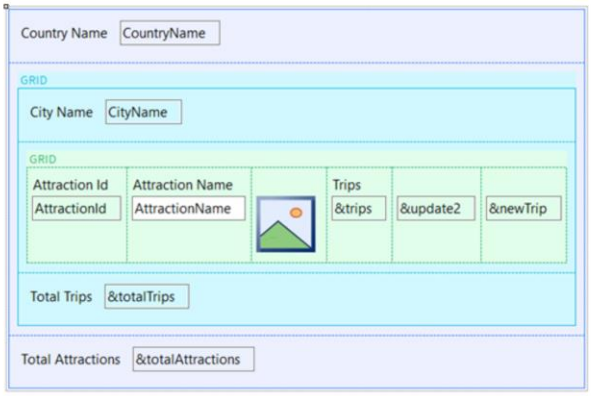
```
Event Grid1.Refresh
     &totalAttractions = 0
endevent
```

```
For each Attraction order CountryId, CityId
where CountryId = &CountryId

     Event Grid1.Load
          &attractions = 0
          for each Attraction
               &attractions = &attractions + 1
          endfor
          &totalAttractions = &totalAttractions + &attractions
     endevent

     Load

     Event Grid2.Refresh
          &totalTrips = 0
     Endevent

     For each Attraction order AttractionName
     where CountryId = &CountryId
     where CityId = &CityId

          Event Grid2.Load
               &trips = Count(TripDate)
               &totalTrips = &totalTrips + &trips
          Endevent

          Load

     endfor
endfor
```

Por tanto, se nos podría ocurrir realizar ese cálculo con otro For each, implementando otro corte de control, anidado al for each más externo.

Pero esto no nos sirve. Tendríamos **un** corte de control **partido** en dos instancias, pero el primero agotaría todas las atracciones de la ciudad en su recorrida, y el segundo se quedaría sin atracciones para recorrer.

With several Grids: nested



Grid1 and Grid2 with Base Tables

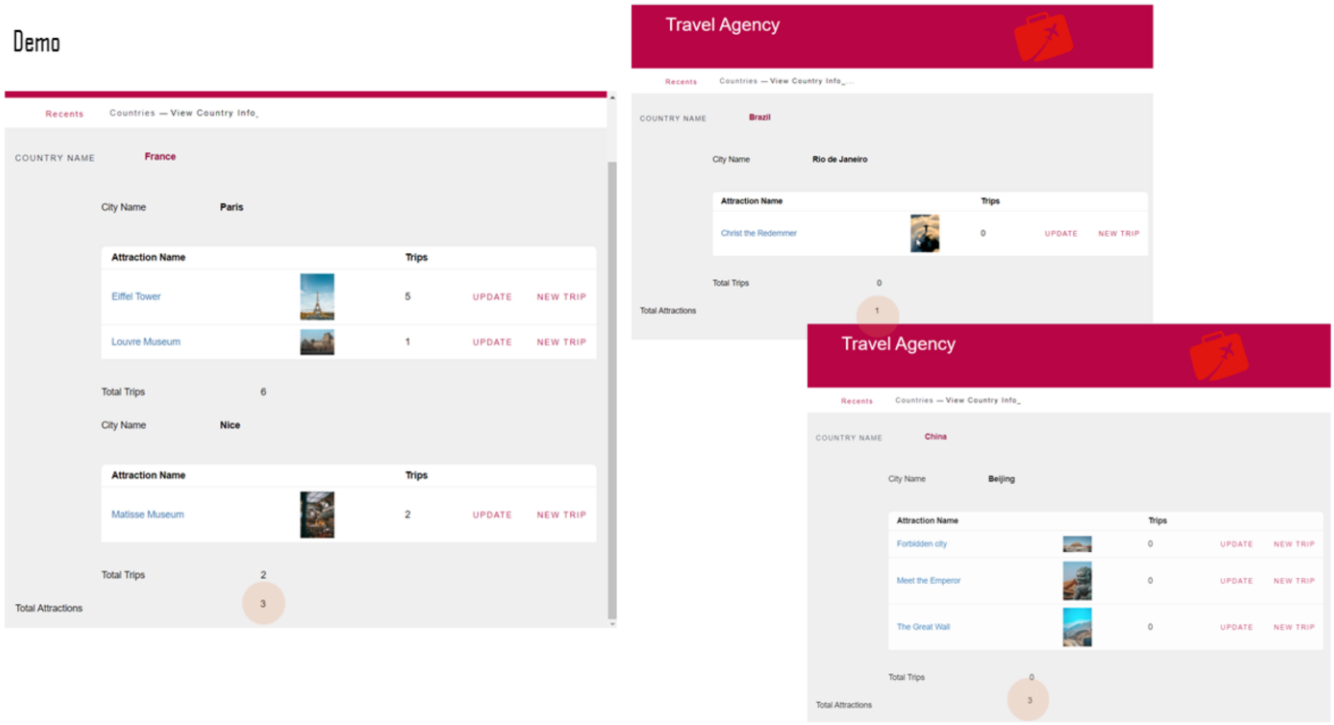
```
Event Grid1.Refresh
    &totalAttractions = 0
endevent
```

```
For each Attraction order CountryId, CityId
where CountryId = &CountryId

    Load

    Event Grid2.Refresh
        &totalTrips = 0
    Endevent

    For each Attraction order AttractionName
    where CountryId = &CountryId
    where CityId = &CityId

        Event Grid2.Load
            &trips = Count(TripDate)
            &totalTrips = &totalTrips + &trips
            &totalAttractions = &totalAttractions + 1
        Endevent

        Load

    endfor

endfor
```

Pero tenemos una solución mucho más sencilla: dado que la cantidad de atracciones será la suma de registros que se carguen en total en el Grid 2, podríamos haber eliminado el evento Load del Grid 1, y haber agregado la suma al Load del Grid 2, sin reinicializar nunca más que en el Refresh la variable...

Así quedó mucho más simple.

Demo



Si ejecutamos... vemos para Francia, por ejemplo, con sus dos ciudades que ni se percibe una diferencia con la implementación con join, porque ambas ciudades tienen atracciones.

Pero en cambio si elegimos Brasil sí vemos la diferencia. O China, por ejemplo.

Y ahora sí se están contando bien las atracciones.

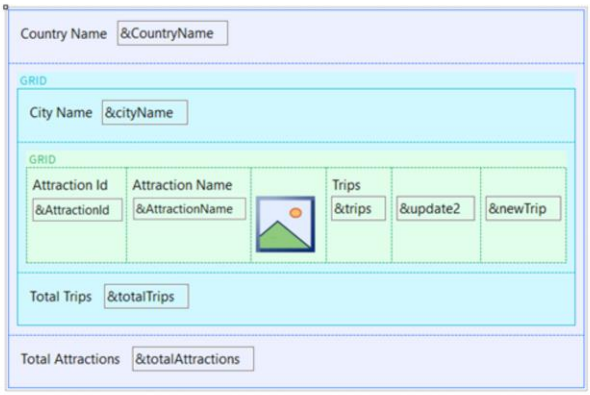
WITHOUT Base Tables

Ahora pasemos al caso de implementación sin tabla base.

With several Grids: nested

Web Form | **Rules** | Events | Conditions | Variables

```
1 parm( in: CountryId );
```



Grid1 and Grid2 **without** Base Tables

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid1.Load
     For each Country.City
          &CountryName = CountryName
          &cityName = CityName
          Load
     endfor
endevent


Event Grid2.Refresh
     &totalTrips = 0
Endevent

Event Grid2.Load
     For each Attraction order AttractionName
     where CityName = &cityName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          &totalTrips = &totalTrips + &trips
          &totalAttractions = &totalAttractions + 1
          Load
     endfor
Endevent
```

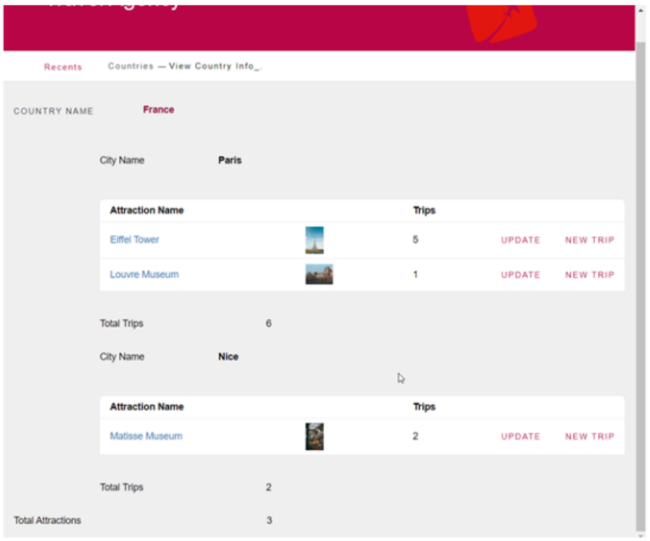
Teníamos este web panel que implementaba el mismo join del principio, pero sin tablas base. Observemos que en la pantalla únicamente tenemos variables y no hay atributos en las propiedades de ninguno de los grids.

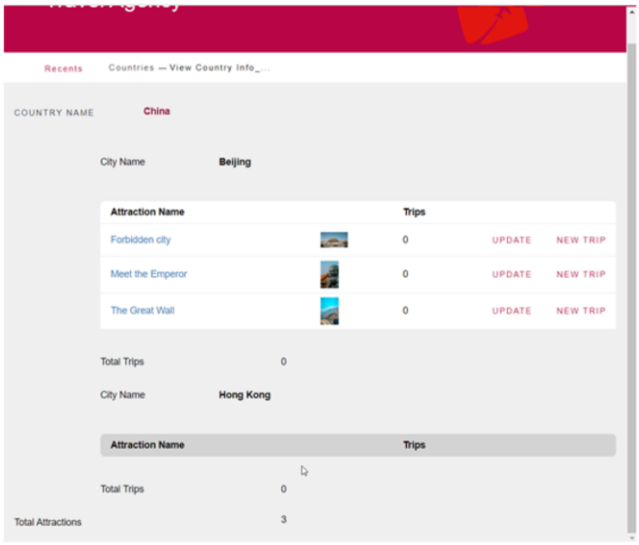
Era en los eventos donde realizábamos, explícitamente, la carga de la base de datos. Hagamos un Save us para dejar este como estaba, con join. E implementar el corte de control en otro. Ya aprovechamos y quitamos el Count de attractions del primer Load, y hacemos la cuenta en el segundo, para dejarlo todo más simple.

Modifiquemos ahora la acción del Work With Country para invocar ahora a este web panel.

Demo

Grid1 and Grid2 **without** Base Tables



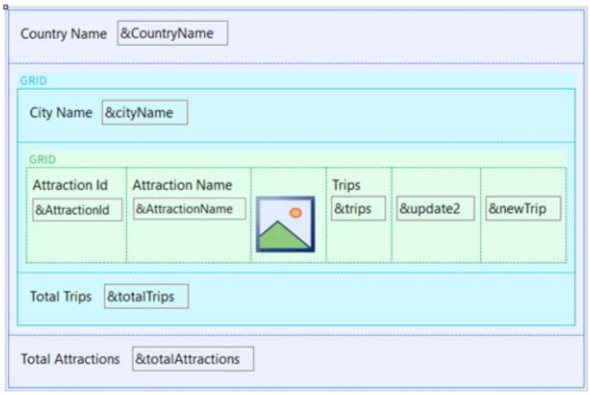


Ejecutemos... y veamos las atracciones de Francia. Y ahora las de China. Se percibe claramente el **join** y no el **corte de control**.

With several Grids: nested

Web Form | **Rules** | Events | Conditions | Variables

```
1  parm( in: CountryId );
```



Grid1 and Grid2 **without** Base Tables

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid1.Load
     For each Country.City
          &CountryName = CountryName
          &cityName = CityName
          Load
     endfor
endevent
```

Control Break?

Do 'Grid2'

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

Event Grid2.Load
     For each Attraction order AttractionName
     where CityName = &cityName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          &totalTrips = &totalTrips + &trips
          &totalAttractions = &totalAttractions + 1
          Load
     endfor
Endevent
```

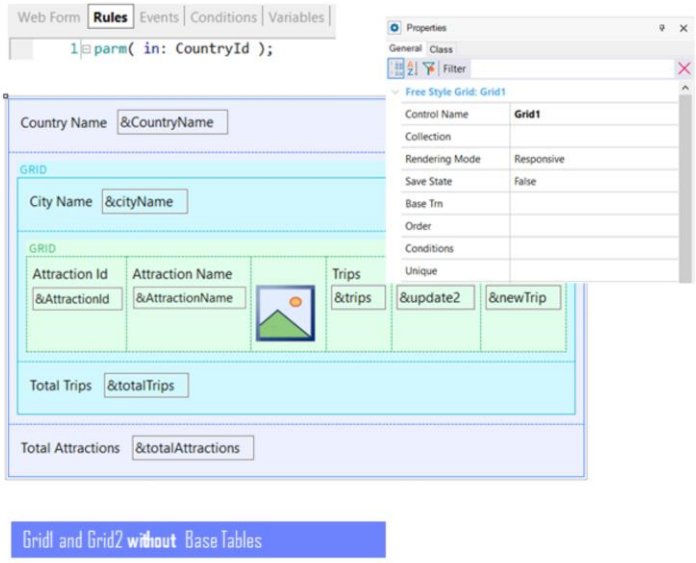
Aunque el comando Load del primer grid dispare el evento Refresh e inmediatamente el Load del seguro grid, verdaderamente no anida los for eachs. Es como si se invocara a una subrutina, digamos.

Es como si el For each del grid anidado se ejecutara aislado. Es por ello que GeneXus no está encontrando un join automático y es por ello que tuvimos que filtrar explícitamente las atracciones de la ciudad que quedó cargada en la variable &cityName, que sí fue cargada por el evento Load que invocó al Load del grid anidado. No tuvimos que colocar también filtro por CountryId ya que se instancia por venir en el parámetro.

Tengamos esto en mente, porque será lo que hará a este caso menos evidente de lo que a primera vista podría parecer.

La pregunta es: ¿cómo hacemos para que el For each correspondiente al grid1 cambie su tabla base a Attraction y establezca un corte de control por país, ciudad?

With several Grids: nested



Grid1 and Grid2 **without** Base Tables

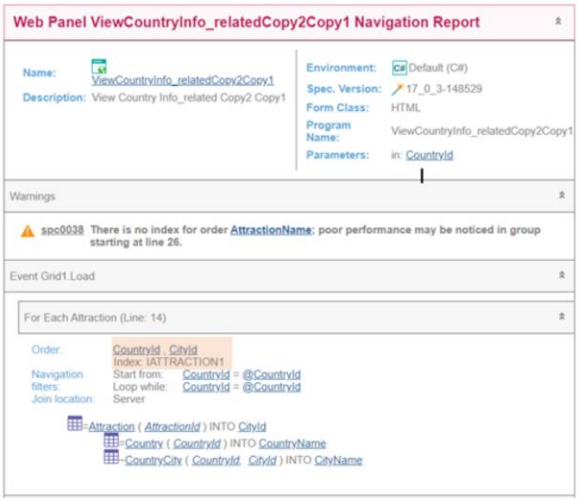
```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid1.Load
     For each Attraction order CountryId, CityId
          &CountryName = CountryName
          &cityName = CityName
          Load
     endfor
endevent


Event Grid2.Refresh
     &totalTrips = 0
Endevent

Event Grid2.Load
     For each Attraction order AttractionName
     where CityName = &cityName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          &totalTrips = &totalTrips + &trips
          &totalAttractions = &totalAttractions + 1
          Load
     endfor
Endevent
```

No será agregando Base Trn ni Order a las propiedades del Grid 1 (porque si lo hiciéramos, transformaríamos la implementación en una con tabla base), sino al For each explícito del evento Load del Grid1.

Por tanto parece obvio que lo primero será modificar el primer for each para que tenga como Base Transaction a Attraction...

Y también parecería evidente que debemos colocar cláusula order para establecer los criterios de agrupamiento por los que queremos que se establezca el corte de control en relación al for each del Grid2.

Demo

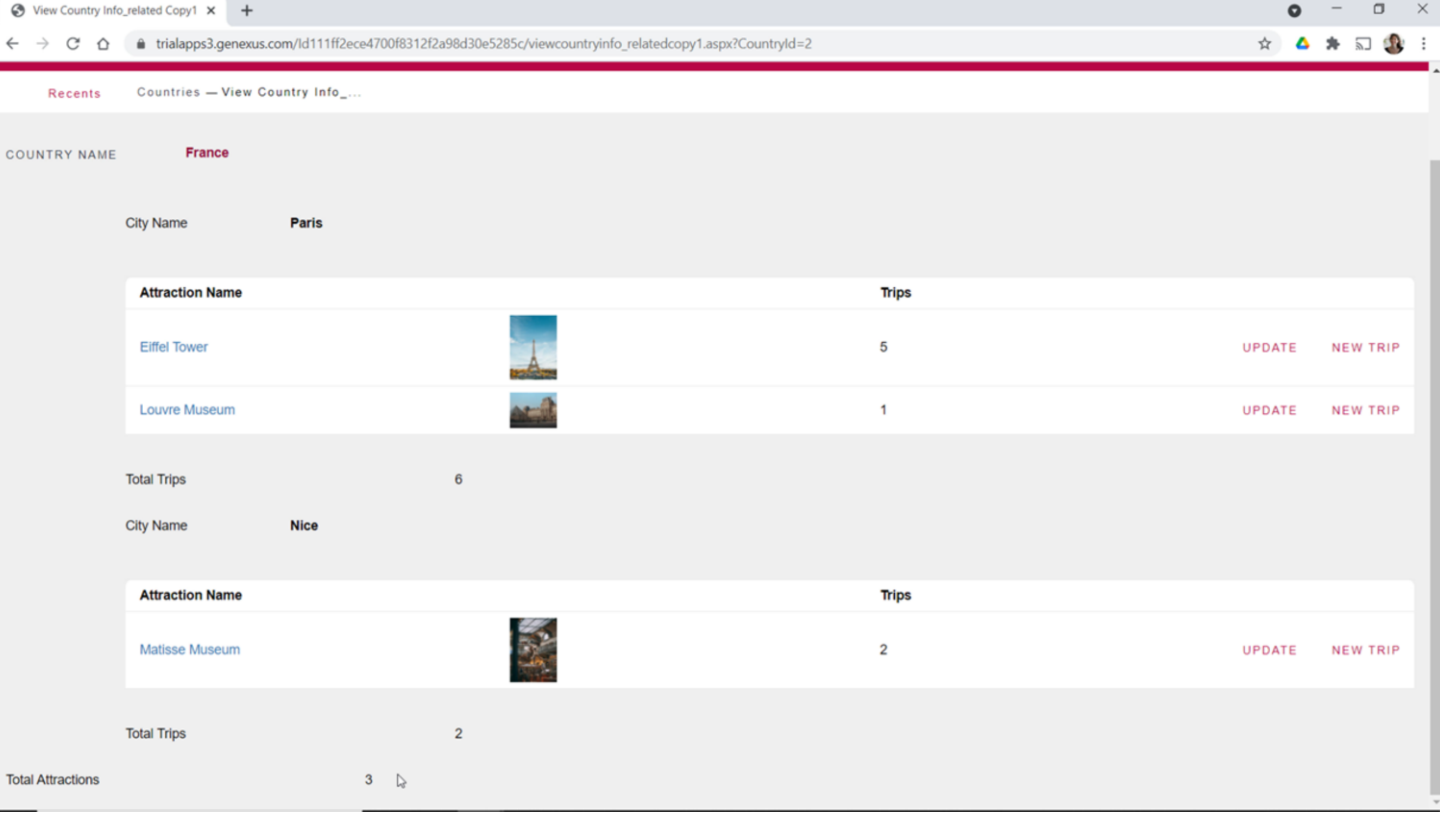


This is not a Control Break!

Sin embargo, si observamos el listado de navegación...

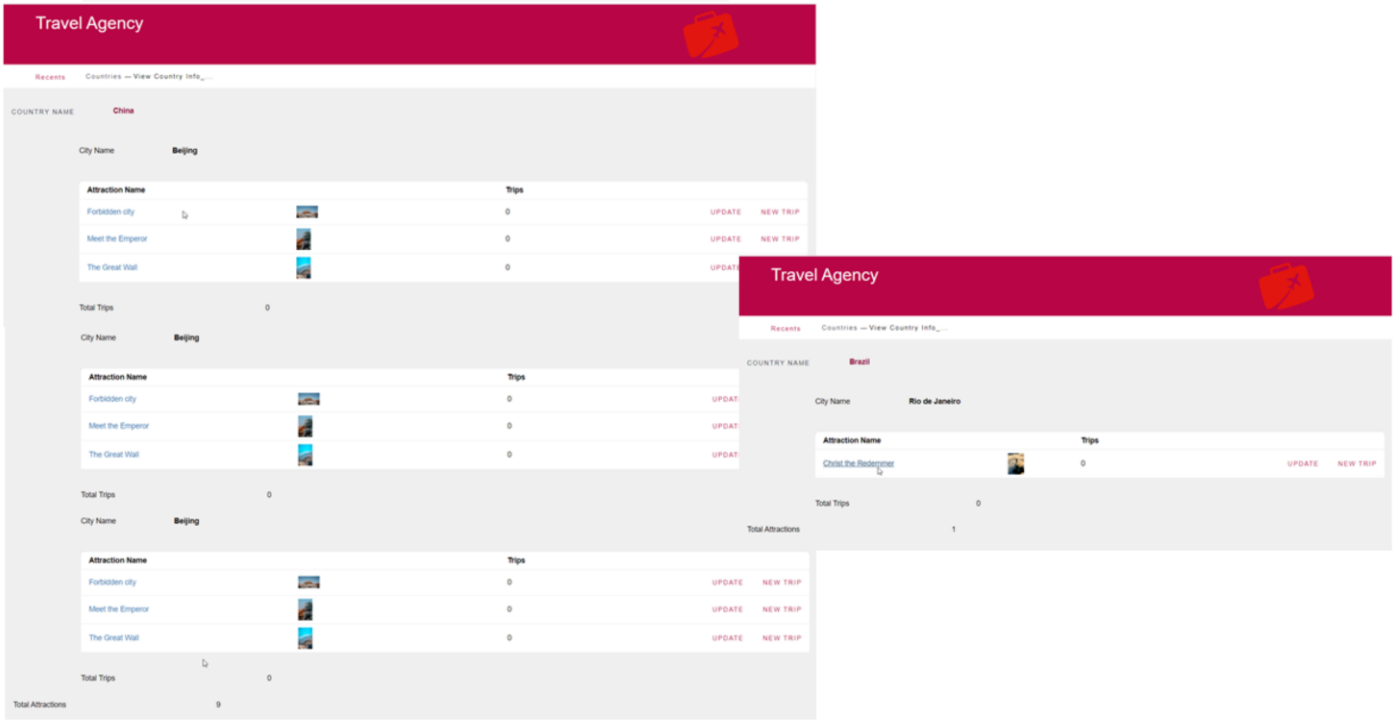
Aparece algo raro, y es que si bien cada for each en apariencia hace lo que debe hacer, no eligió como order de cada uno el mismo, para utilizar un único índice y hacer eficiente la recorrida. Algo no anda bien.

Evidentemente no entendió que deberá hacer un corte de control.



Y esto lo constatamos si ejecutamos.

Veamos que para las atracciones de Francia sale dos veces París, que coincide con las dos atracciones de París que hay.



Para las de China, sale tres veces Beijing, que coincide con las tres atracciones de Beijing que hay.

Y para Brasil solo una vez, coincidiendo con las atracciones de Río que hay.

¿Qué está pasando?

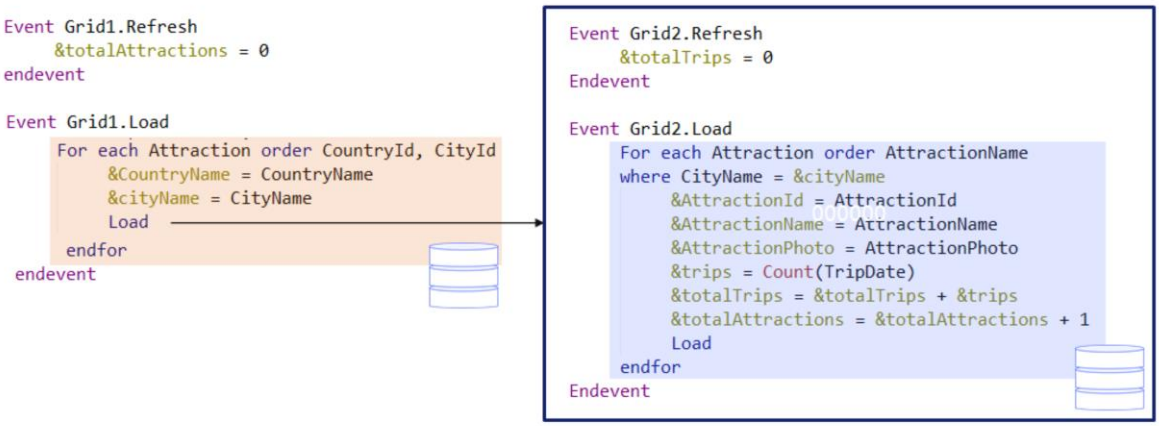
With several Grids: nested

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid1.Load
     For each Attraction order CountryId, CityId
          &CountryName = CountryName
          &cityName = CityName
          Load

          Event Grid2.Refresh
               &totalTrips = 0
          Endevent

          Event Grid2.Load
               For each Attraction order AttractionName
               where CityName = &cityName
                    &AttractionId = AttractionId
                    &AttractionName = AttractionName
                    &AttractionPhoto = AttractionPhoto
                    &trips = Count(TripDate)
                    &totalTrips = &totalTrips + &trips
                    &totalAttractions = &totalAttractions + 1
                    Load
               endfor
          Endevent

     endfor
endevent
```

Evidentemente no está encontrando que debe hacer un corte de control.

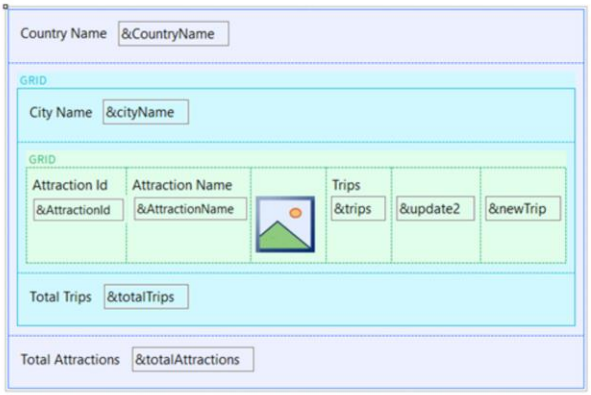
Es que, como ya anticipamos, no anida **realmente** las navegaciones.

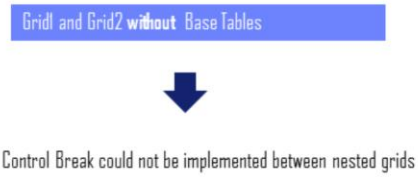
With several Grids: nested



Es como si fueran dos for eachs independientes, solo que desde uno se invoca la ejecución del otro, pero a través de dos consultas separadas a la base de datos.

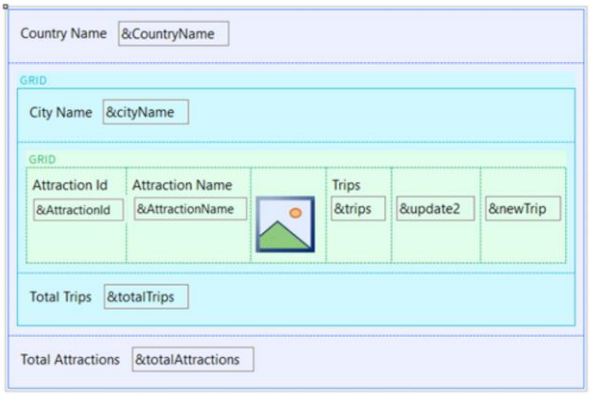
With several Grids: nested





Esto equivale a decir que no podemos implementar **realmente** un corte de control entre dos grids anidados sin tablas base.

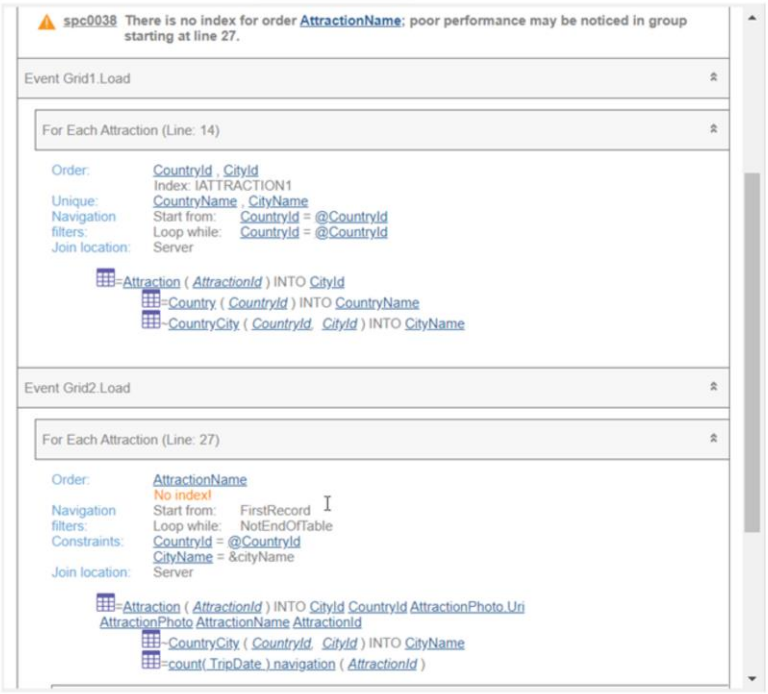
With several Grids: nested




```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid1.Load
     For each Attraction order CountryId, CityId
          unique CountryName, CityName
          &CountryName = CountryName
          &cityName = CityName
          Load
     endfor
endevent

Event Grid2.Refresh
     &totalTrips = 0
Endevent

Event Grid2.Load
     For each Attraction order AttractionName
     where CityName = &cityName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          &totalTrips = &totalTrips + &trips
          &totalAttractions = &totalAttractions + 1
          Load
     endfor
Endevent
```

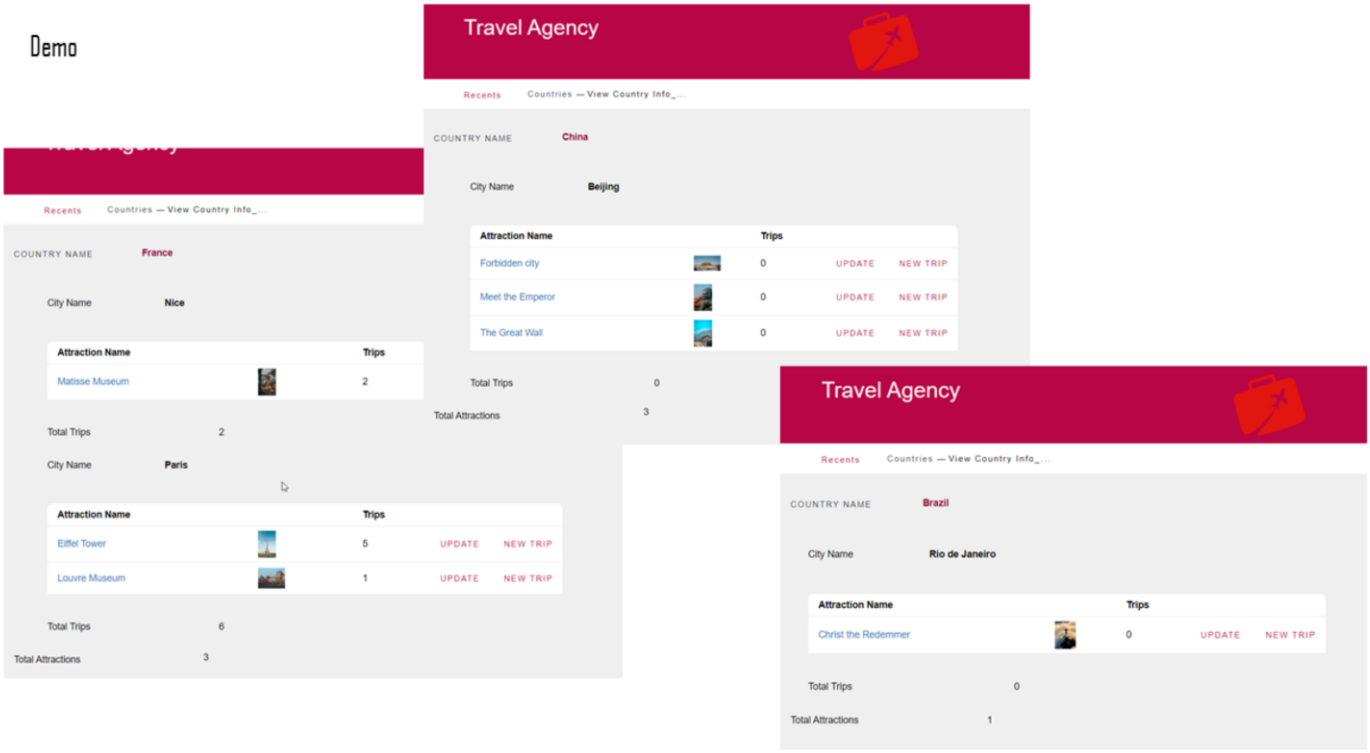
La solución que tenemos por el momento es utilizar, para el primer For each, la cláusula unique. Es decir, que de la tabla Attraction si varios registros tienen el mismo país y ciudad, se quede con uno solo de ellos. Y para éste, cargue las variables y ejecute el evento Refresh e inmediatamente Load del grid2, que ejecutará su for each como si fuera independiente por completo del anterior. Y es justamente por ello que esta vez sí nos permitirá utilizar la cláusula unique.

Demo



Si ahora vemos el listado de navegación... parece que así sí va a funcionar.

Demo



Ejecutemos...

Lo hemos conseguido.

WITH or WITHOUT Base Tables?

Grid1 and Grid2 with Base Tables

Free Style Grid: Grid1

Control Name	**Grid1**
Collection	
Rendering Mode	Responsive
Save State	False
Base Trn	**Attraction**
Order	**CountryId, CityId**
Conditions	
Unique	

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

Event Grid2.Refresh
     &totalTrips = 0
Endevent

Event Grid2.Load
     &trips = Count(TripDate)
     &totalTrips = &totalTrips + &trips
     &totalAttractions = &totalAttractions + 1
Endevent
```

Grid1 and Grid2 without Base Tables

```
Event Grid1.Refresh
     &totalAttractions = 0
endevent

 Event Grid1.Load
      For each Attraction order CountryId, CityId
           unique CountryName, CityName
           &CountryName = CountryName
           &cityName = CityName
           Load
      endfor
 endevent

Event Grid2.Refresh
     &totalTrips = 0
Endevent

Event Grid2.Load
     For each Attraction order AttractionName
     where CityName = &cityName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          &totalTrips = &totalTrips + &trips
          &totalAttractions = &totalAttractions + 1
          Load
     endfor
Endevent
```

Así es que, por el momento, nos está resultando mucho más sencillo implementar un corte de control, cuando los grids **tienen tabla base**.
Estrictamente hablando, solo en ese caso se tratará de un **verdadero corte de control**.

En el segundo caso, cuando los grids no tienen tabla base, tan solo lo estamos simulando, pero en verdad habrá dos consultas independientes a la tabla Attraction y no una sola que resuelva todo, como sucede en el verdadero corte de control.

Te invitamos a que pruebes todo lo que hemos visto.

GeneXus™

training.genexus.com

wiki.genexus.com

training.genexus.com/certifications