

Web Panels

Como implementar cortes de controle em Grids aninhados

GeneXus™

Web Panel with SEVERAL Grids

Em outro vídeo estudamos como se determinavam as tabelas base e a navegação de um web panel com vários grids.

With several Grids: nested

```
Web Form | Rules | Events | Conditions | Variables |
1 param( in: CountryId );
```

Country Name

GRID

City Name

GRID

Attraction Id	Attraction Name	Trips			
<input type="text" value="AttractionId"/>	<input type="text" value="AttractionName"/>	<input type="text" value="Trips"/>	<input type="text" value="update2"/>	<input type="text" value="newTrip"/>	

Total Trips

Total Attractions

Country Name

GRID

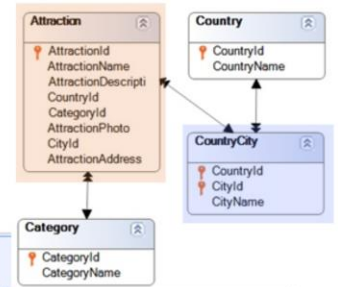
City Name

GRID

Attraction Id	Attraction Name	Trips			
<input type="text" value="AttractionId"/>	<input type="text" value="AttractionName"/>	<input type="text" value="Trips"/>	<input type="text" value="update2"/>	<input type="text" value="newTrip"/>	

Total Trips

Total Attractions



E, em particular, vimos um exemplo de grids aninhados que realizavam um **join**.

Ou seja, o grid externo percorria uma tabela com uma relação 1 para N com a tabela percorrida pelo grid interno, independentemente desses grids terem sido implementados com ou sem tabela base.

Em ambas as soluções, o web panel recebia por parâmetro um identificador de país, e no grid externo eram mostradas as cidades daquele país, ou seja, ia percorrer CountryCity; e no interno eram mostradas as atrações turísticas daquela cidade. Quer dizer, ia percorrer Attraction.

With several Grids: nested

Web Form **Rules** Events Conditions Variables
 1 parm(in: CountryId);

The screenshot shows a web form with the following structure:

- Country Name:
- GRID (light blue background):
 - City Name:
 - GRID (light green background):

Attraction Id	Attraction Name	Trips		
AttractionId	AttractionName	&trips	&update2	&newTrip
 - Total Trips:
- Total Attractions:

Grid1 and Grid2 with Base Tables

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

Pseudo-code

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 o caso de grids com tabela base, esse join se estabelecia automaticamente, sem ter que fazer nada. GeneXus o detectava e adicionava o filtro em seu programa fonte.

With several Grids: nested

Web Form **Rules** Events Conditions Variables

1 parm(in: CountryId);

Country Name &CountryName

GRID

City Name &cityName

GRID

Attraction Id	Attraction Name	Trips	Trips	
&AttractionId	&AttractionName	&trips	&update2	&newTrip

Total Trips &totalTrips

Total Attractions &totalAttractions

Grid1 and Grid2 **without** Base Tables

Pseudo-code

```

Event Grid1.Refresh
    &totalAttractions = 0
endevent

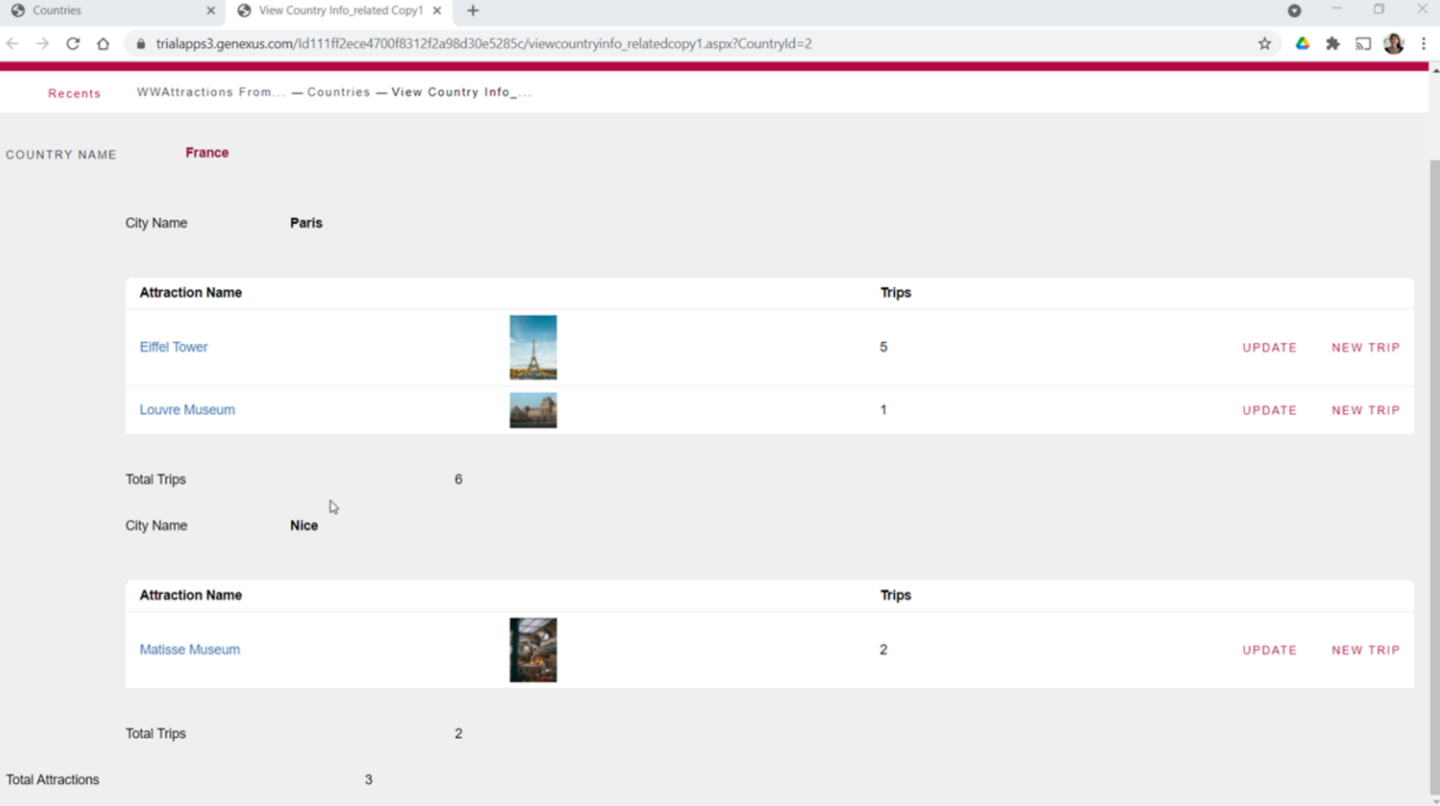
Event Grid1.Load
    For each Country.City
        &CountryName = CountryName
        &cityName = CityName
        &attractions = Count(AttractionName)
        &totalAttractions = &totalAttractions + &attractions
    Load
    endfor
endevent

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

```




Em vez disso, para o caso de grids sem tabela base, tínhamos que especificá-lo no for each que implementávamos para carregar o grid aninhado (e apenas filtrávamos por cidade, porque o filtro por país já estava implícito por recebê-lo por parâmetro no atributo CountryId).



Aqui temos o web panel com ambos os grids **com tabela base**.
Adicionamos uma ação no pattern Work With de países para invocar este web panel.

Se escolhermos França: aqui vemos as atrações de Paris e as de Nice, que são as duas cidades temos inseridas no sistema.



COUNTRY NAME	Brazil								
City Name	Rio de Janeiro								
<table><thead><tr><th>Attraction Name</th><th></th><th>Trips</th><th></th></tr></thead><tbody><tr><td>Christ the Redemmer</td><td></td><td>0</td><td>UPDATE NEW TRIP</td></tr></tbody></table>		Attraction Name		Trips		Christ the Redemmer		0	UPDATE NEW TRIP
Attraction Name		Trips							
Christ the Redemmer		0	UPDATE NEW TRIP						
Total Trips	0								
City Name	Sao Paulo								
<table><thead><tr><th>Attraction Name</th><th></th><th>Trips</th><th></th></tr></thead><tbody></tbody></table>		Attraction Name		Trips					
Attraction Name		Trips							
Total Trips	0								
Total Attractions	1								

Agora, vejamos o que acontece se em vez de escolher França escolhermos Brasil, por exemplo, que também tem duas cidades inseridas.

Vemos que para a primeira, Rio de Janeiro, é mostrada uma atração turística, mas para a segunda, São Paulo, nenhuma é mostrada. Isso ocorre justamente por se tratar de um **join**.

With several Grids: nested

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


1 parm(in: CountryId);

Country Name

GRID

City Name

GRID

Attraction Id	Attraction Name		Trips	&update2	&newTrip
<input type="text" value="AttractionId"/>	<input type="text" value="AttractionName"/>		<input type="text" value="trips"/>	<input type="text" value="update2"/>	<input type="text" value="newTrip"/>

Grid1.Refresh()

Grid1 → Load → Rio de Janeiro

Grid2.Refresh()

Grid2 → Load → Christ the Redemmer

Grid1 → Load() → Sao Paulo

Grid2.Refresh()

Control Break instead of Join

Visto que primeiro será executado o Refresh do grid externo, e depois, posicionados na primeira cidade, Rio de Janeiro, será carregada no Grid1, e imediatamente será executado o Refresh do grid aninhado, e em seguida serão carregadas as atrações do Rio de Janeiro, que neste caso é apenas uma, o Cristo Redentor.

Em seguida, passará para a próxima cidade, São Paulo, a carregará e será feito o Refresh do grid aninhado. Mas quando for percorrer a tabela de atrações para carregar apenas as de São Paulo, não encontrará nenhuma.

Para que sejam mostradas apenas as cidades com atrações, precisamos implementar um **corte de controle** e não um **join** .


With several Grids: nested

Web Form **Rules** Events Conditions Variables

1 param(in: CountryId);

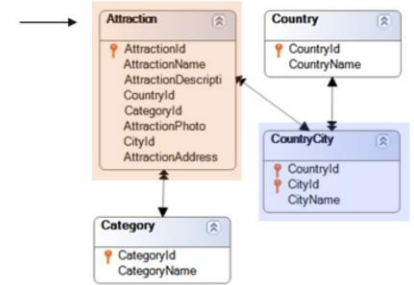
Country Name

GRID

Attraction Id <input type="text" value="AttractionId"/>	Attraction Name <input type="text" value="AttractionName"/>		Trips <input type="text" value="&trips"/>	update2 <input type="text" value="&update2"/>	new Trip <input type="text" value="&newTrip"/>
--	--	---	--	--	---

GRID

City Name



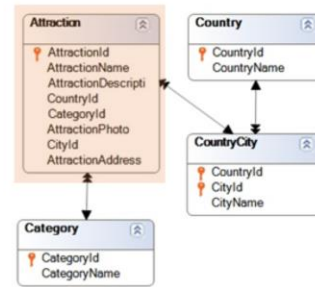
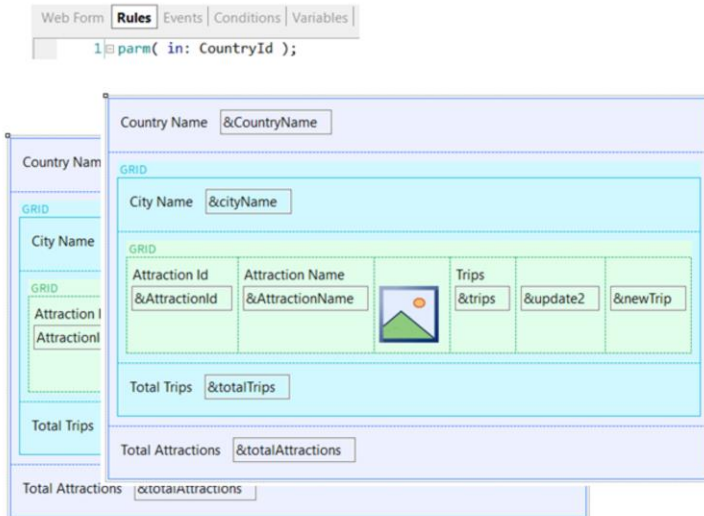
Attraction

Attraction

Havíamos deixado esboçado um caso onde seria produzido um corte de controle indesejado, em vez de um join. Mas só atendemos a esse efeito ao não especificar transação base para o grid aninhado. Que parecia ser CountryCity, mas na verdade seria Attraction. Mas não tínhamos entrado na navegação.

Agora o faremos, com nosso exemplo.

With several Grids: nested



```
For each Attraction order CountryId, CityId
```

```
  print CountryName
  print CityName
```

```
  For each Attraction
    print AttractionName, AttractionPhoto, etc.
  endfor
```

```
endfor
```

Necessitamos, como para o caso de uma lista, que ambos os for eachs (sejam implícitos, quer dizer, provenham de grids **com** tabela base, ou sejam explícitos, quer dizer, provenham de grids **sem** tabela base) tenham a mesma tabela base, Attraction. E que o primeiro esteja de acordo com o grupo que irá realizar o corte por país/cidade.

Para isso, então, basta modificar a **transação base** e adicionar a **order** ao primeiro grid.

Dependerá se os grids foram implementados com ou sem tabela base, para ver como fazê-lo.

WITH Base Tables

Comecemos pelo caso em que o web panel foi implementado com ambos os grids com tabela base

With several Grids: nested

Grid1 and Grid2 with Base Tables

Web Form **Rules** Events Conditions Variables


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

Country Name

GRID

City Name

GRID

Attraction Id	Attraction Name	trips	&update2	&newTrip
AttractionId	AttractionName			

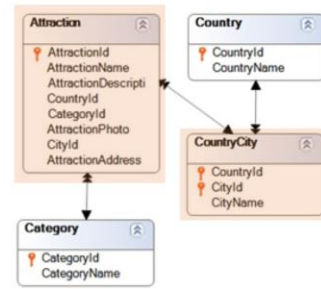
Properties

General Class

Filter

Free Style Grid: Grid1

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



For each **Attraction** order CountryId, CityId

```
print CountryName
print CityName
```

```
For each Attraction
  print AttractionName, AttractionPhoto, etc.
endfor
```

```
endfor
```

Neste caso, o objetivo é alcançado modificando as propriedades que vemos do grid externo, por estas outras.

Demo

The screenshot displays the GeneXus IDE interface. On the left, a web layout titled 'ViewCountryInfo_relatedCopy1' is shown. It contains a 'MainTable' and a 'Grid1'. The layout includes input fields for 'Country Name', 'City Name', 'Attraction Id', 'Attraction Name', 'Trips', 'Total Trips', and 'Total Attractions'. A grid is visible with columns for 'Attraction Id', 'Attraction Name', 'Trips', and two buttons labeled '&update2' and '&newTrip'. On the right, two 'Properties' windows are open. The top window is for 'Free Style Grid: Grid1' and the bottom window is for 'Grid: Grid2'.

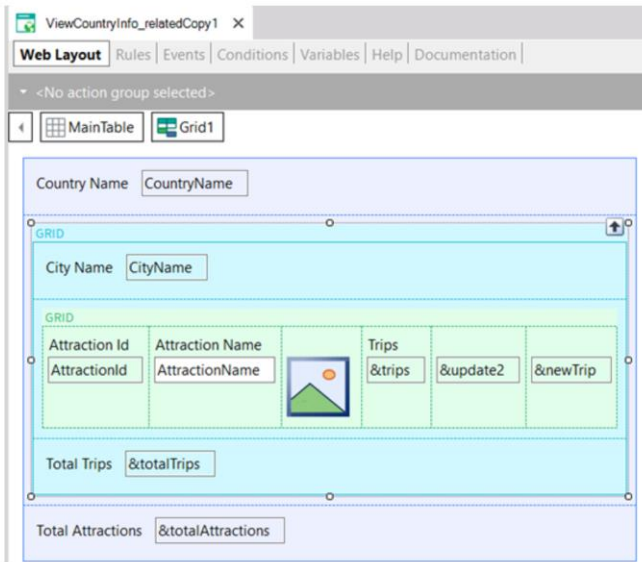
Property	Value
Control Name	Grid1
Collection	
Rendering Mode	Responsive
Save State	False
Base Trn	Country.City
Order	
Conditions	
Unique	

Property	Value
Control Name	Grid2
Collection	
Base Trn	Attraction
Order	AttractionName
Conditions	
Unique	
Data Selector	(none)

Antes de fazer isso, vamos ao GeneXus. Vemos as propriedades de ambos os grids: o primeiro tem Base Trn Country.City e nenhum order.

O segundo tem Base Trn Attraction e uma 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

```

Enos eventos vemos que foram programados os Refresh e Load de cada grid, apenas para inicializar e sumarizar ou contar em variáveis (as que mostrarão o total de atrações carregadas e o total de excursões -trips- nas quais essas atrações se encontram).

Depois, temos eventos de usuário para chamar diversos objetos. Aqui não importam para nada.

Demo

▲ spc0038 There is no index for order **AttractionName**; poor performance may be noticed in grid 'Grid2'.

Event Grid1.Load

Order: **CountryId**
 Index: ICOUNTRYCITY

Navigation filters: Start from: **CountryId = @CountryId**
 Loop while: **CountryId = @CountryId**

Join location: Server

CountryCity (**CountryId** , **CityId**) INTO **CityId** **CityName**
 Country (**CountryId**) INTO **CountryName**
 count(**AttractionName**).navigation (**CountryId** , **CityId**)

Formulas

Navigation to evaluate: count(**AttractionName**)

Given: **CountryId** **CityId**
 Index: ICOUNTRYCITY
 Group by: **CountryId** **CityId**

Attraction

Event Grid2.Load

Grid1

Order: **AttractionName**
 No index

Navigation filters: Start from: FirstRecord
 Loop while: NotEndOfTable

Constraints: **CountryId = @CountryId**
CityId = @CityId

Join location: Server

Attraction (**AttractionId**) INTO **CountryId** **CityId** **AttractionPhoto** **AttractionPhoto.Uri** **AttractionName** **AttractionId**
 count(**TripDate**).navigation (**AttractionId**)

Formulas

Navigation to evaluate: count(**TripDate**)

Given: **AttractionId**
 Index: IATTRACTION
 Group by: **AttractionId**

TripAttraction
 Trip (**TripId**)

Observemos em GeneXus a lista de navegação do web panel.

Vemos claramente a tabela base do primeiro grid: CountryCity, e que filtrará pelo país recebido por parâmetro. E então vamos para o Load aninhado, que tem como tabela base Attraction, e filtra pelo país e a cidade do grid externo.

Vemos, claro, o join.

Observemos, além disso, que ordena o primeiro For each implícito por CountryId, e o segundo por AttractionName, para o qual informa que não há um índice.

With several Grids: nested

Web Form **Rules** Events Conditions Variables

1) parm(in: CountryId);

Country Name

GRID

City Name

GRID

Attraction Id	Attraction Name	Trips		
AttractionId	AttractionName	&trips	&update2	&newTrip

Total Trips

Total Attractions

Grid1 and Grid2 with Base Tables

```
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
```

O pseudocódigo do fonte que programará GeneXus será mais ou menos como vemos. Onde a propriedade Base Transaction do Grid1 foi utilizada para programar a base Transaction do For each implícito.

No segundo for each foi colocada como cláusula order o que havia na propriedade Order do grid, que era AttractionName, e por isso vimos os índices escolhidos que vimos.

Em suma, será disparado o Refresh do grid externo e a seguir o For each implícito que estamos vendo, que com o interno formarão **um join**.

With several Grids: nested

Web Form **Rules** Events Conditions Variables

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

Country Name CountryName

GRID

City Name CityName

GRID

Attraction Id	Attraction Name
AttractionId	AttractionName

Total Trips &totalTrips

Total Attractions &totalAttractions

Properties

General Class

Filter

Free Style Grid: Grid1

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

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
endfor
```

```
For each Attraction order AttractionName
  where CountryId = @CountryId
  where CityId = @CityId

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

  Load
endfor
```

```
endfor
```

Agora sim, modifiquemos as propriedades do Grid1, o externo.

Ao fazer isso, a propriedade Base Transaction do grid fará com que seja modificada a Base Transaction do For each implícito. E a propriedade Order se tornará a cláusula order desse for each. Com isso, já será o suficiente.

Porque será disparado o Refresh do grid externo e, em seguida o for each implícito que estamos vendo, que com o interno formarão um **corte de controle**. Portanto, serão agrupadas as atrações turísticas por cidades do país (o filtro por CountryId se deve ao parâmetro).

Portanto, para **cada grupo** de atrações turísticas formado por cada cidade, será disparado uma vez o evento Load. Em seguida, será carregada no grid a primeira linha, com o CityName da cidade do primeiro grupo de atrações (e, claro, seu CountryName, que será o mesmo para todos).

E então será disparado o evento Refresh do grid aninhado, após o qual será executado o for each implícito, que percorrerá apenas as atrações do grupo, ou seja, as desse país e cidade. Para cada uma dessas atrações do grupo, disparará o evento Load e o comando Load para carregá-la no grid aninhado.

E assim sucessivamente com todos os grupos.

Web Panel ViewCountryInfo_relatedCopy1Copy1 Navigation Report

Name: ViewCountryInfo_relatedCopy1Copy1
Description: View Country Info_related Copy1 Copy1

Environment: Default (C#)
Spec. Version: 17_0_3-148529
Form Class: HTML
Program Name: ViewCountryInfo_relatedCopy1Copy1
Parameters: in: CountryId

Warnings

▲ spe0038 There is no index for order CountryId, CityId, AttractionName; poor performance may be noticed in grid "Grid2".

Event Grid1.Load

Order: CountryId, CityId, AttractionName
No index!

Navigation filters: Start from: CountryId = @CountryId
 Loop while: CountryId = @CountryId
 Join location: Server

Attraction (AttractionId) INTO CityId AttractionPhoto AttractionPhoto
 Country (CountryId) INTO CountryName
 CountryCity (CountryId, CityId) INTO CityName
 count(AttractionName).navigation (AttractionId, CountryId)

Formulas

Navigation to evaluate: count(AttractionName)

Given: AttractionId CountryId CityId
 Index: IATTRACTION
 Group by: AttractionId CountryId CityId

Attraction (AttractionId, CountryId, CityId)

Grid1 and Grid2 with Base Tables

Event Grid2.Load

Grid1
 Order: CountryId, CityId, AttractionName
No index!

Navigation filters: Loop while: CountryId = @CountryId and CityId = @CityId
 Join location: Server

Attraction (AttractionId) INTO AttractionPhoto AttractionPhoto.Uri, AttractionName AttractionId
 count(TripDate).navigation (AttractionId)

Formulas

Navigation to evaluate: count(TripDate)

Given: AttractionId
 Index: IATTRACTION
 Group by: AttractionId

TripAttraction
 Trip (TripId)

Se observamos a lista de navegação, vemos claramente que ambos os grids navegarão a mesma tabela, Attraction, utilizando um índice composto pelos atributos da propriedade Order do primeiro grid, mais a order do grid aninhado.

E vemos claramente que para o grid aninhado, apenas serão percorridas as atrações que correspondem ao país e cidade do primeiro grid.




Demo

Travel Agency

Recents Countries — View Country Info_...

COUNTRY NAME **China**

City Name **Beijing**

Attraction Name		Trips	
Forbidden city		0	UPDATE
Meet the Emperor		0	UPDATE
The Great Wall		0	UPDATE



Total Trips 0

Total Attractions 1

Recents Countries — View Country Info_...


COUNTRY NAME **France**

City Name **Paris**

Attraction Name		Trips		
Eiffel Tower		5	UPDATE	NEW TRIP
Louvre Museum		1	UPDATE	NEW TRIP

Total Trips 6

City Name **Nice**

Attraction Name		Trips		
Matisse Museum		2	UPDATE	NEW TRIP

Total Trips 2

Total Attractions 2

Se o testamos agora em execução... vemos exatamente o que queríamos.

Observemos por exemplo, China. Perfeito.

E se vamos para França... neste caso não notamos nenhuma diferença com o caso de join.

Mas as atrações não estão sendo corretamente contadas. Por quê?

With several Grids: nested

Web Form **Rules** Events Conditions Variables


1 parm(in: CountryId);

Country Name

GRID

City Name

GRID

Attraction Id	Attraction Name		Trips		
AttractionId	AttractionName		&trips	&update2	&newTrip

Total Trips

Total Attractions

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

É que no Grid1 estamos utilizando a fórmula Count para contar as atrações que correspondem a esse país e cidade. Isto funcionava quando a tabela base do for each era CountryCity, mas não agora que é Attraction.

Demo

Web Panel ViewCountryInfo_relatedCopy1Copy1 Navigation Report

Name: ViewCountryInfo_relatedCopy1Copy1
Description: View Country Info_related Copy1

Environment: Default (C#)
Spec. Version: 17_0_3-148529
Form Class: HTML
Program Name: ViewCountryInfo_relatedCopy1Copy1
Parameters: in: CountryId

Warnings

▲ spc0038 There is no index for order **CountryId, CityId, AttractionName**; poor performance may be noticed in grid 'Grid2'.

Event Grid1.Load

Order: CountryId, CityId, AttractionName
No index!
Navigation filters: Start from: CountryId = @CountryId
 Loop while: CountryId = @CountryId
Join location: Server

Attraction (AttractionId) INTO CityId AttractionPhoto AttractionPhotoUri
 Country (CountryId) INTO CountryName
 CountryCity (CountryId, CityId) INTO CityName
 count(AttractionName) navigation (AttractionId, CountryId)

Formulas

Navigation to evaluate: count(AttractionName)

Given: AttractionId CountryId CityId
Index: IATTRACTION
Group by: AttractionId CountryId CityId

Attraction (AttractionId, CountryId, CityId)

Event Grid2.Load

Grid1
Order: CountryId, CityId, AttractionName
No index!
Navigation filters: Loop while: CountryId = @CountryId and CityId = @CityId
Join location: Server

Attraction (AttractionId) INTO AttractionPhoto AttractionPhotoUri AttractionName AttractionId
 count(TripDate) navigation (AttractionId)

Formulas

Navigation to evaluate: count(TripDate)

Given: AttractionId
Index: IATTRACTION
Group by: AttractionId

TripAttraction
 Trip (TripId)

Grid1 and Grid2 with Base Tables

O problema com a fórmula do primeiro grid é claro na lista de navegação.

Não podemos percorrer uma tabela e realizar uma agregação na mesma tabela.

Demo

Web Panel ViewCountryInfo_relatedCopy1Copy1 Navigation Report

Name: ViewCountryInfo_relatedCopy1Copy1
 Description: View Country Info_related Copy1

Environment: Default (C#)
 Spec. Version: 17_0_3-148529
 Form Class: HTML
 Program Name: ViewCountryInfo_relatedCopy1Copy1
 Parameters: in: CountryId

Warnings

▲ spc0038 There is no index for order **CountryId, CityId, AttractionName**; poor performance may be noticed in grid 'Grid2'.

Event Grid1.Load

Order: CountryId, CityId, AttractionName
 No index!

Navigation filters: Start from: CountryId = @CountryId
 Loop while: CountryId = @CountryId
 Join location: Server

=Attraction (AttractionId) INTO CityId AttractionPhoto AttractionPhoto Un. AttractionName AttractionId
 =Country (CountryId) INTO CountryName
 =CountryCity (CountryId, CityId) INTO CityName
 =count(AttractionName) navigation (AttractionId, CountryId)

Formulas

Navigation to evaluate: count(AttractionName)

Given: AttractionId CountryId CityId
 Index: IATTRACTION
 Group by: AttractionId CountryId CityId

=Attraction (AttractionId, CountryId, CityId)

Event Grid2.Load

Grid1
 Order: CountryId, CityId, AttractionName
 No index!

Navigation filters: Loop while: CountryId = @CountryId and CityId = @CityId
 Join location: Server

=Attraction (AttractionId) INTO AttractionPhoto AttractionPhoto Un. AttractionName AttractionId
 =count(TripDate) navigation (AttractionId)

Formulas

Navigation to evaluate: count(TripDate)

Given: AttractionId
 Index: IATTRACTION
 Group by: AttractionId

=TripAttraction
 =Trip (TripId)

error spc0211: Unique clause in break group not supported in grid 'Grid1'. (Web Panel 'ViewCountryInfo_rel:
 Failed: Specification

Properties

General Class

Free Style Grid: Grid1

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

Para que isso funcione teríamos que utilizar a **cláusula unique**, que neste caso não nos serve, pois não é suportada para cortes de controle (nos referimos ao corte de controle entre o grid 1 e o grid 2).

With several Grids: nested

The screenshot shows a user interface with three nested grid levels. The top level (Grid1) contains a 'Country Name' input field. The middle level (Grid2) contains a 'City Name' input field. The bottom level (Grid3) contains a table with columns: 'Attraction Id', 'Attraction Name', a landscape icon, and 'Trips'. Below the table are 'Total Trips' and 'Total Attractions' input fields.

Grid1 and Grid2 with Base Tables

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

```
For eachAttraction 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 eachAttraction order AttractionName
    where CountryId = @CountryId
    where CityId = @CityId

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

    Load

  endfor

endfor
```

Portanto, poderia nos ocorrer realizar este cálculo com outro For each, implementando outro corte de controle, aninhado no for each mais externo.

Mas isto não nos serve. Teríamos **um** corte de controle **dividido** em duas instâncias, mas o primeiro esgotaria todas as atrações da cidade em seu percurso, e o segundo ficaria sem atrações para percorrer.

With several Grids: nested

Grid1 and Grid2 with Base Tables

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

```
For eachAttraction order CountryId, CityId
  where CountryId = @CountryId
```

Load

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

```
For eachAttraction order AttractionName
```

```
  where CountryId = @CountryId
```

```
  where CityId = @CityId
```

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

Load

```
endfor
```

```
endfor
```

Mas temos uma solução muito mais simples: como a quantidade de atrações será a soma de registros carregados no total no Grid 2, poderíamos ter eliminado o evento Load do Grid 1, e ter adicionado a soma ao Load do Grid 2, sem nunca reinicializar a variável além do que no Refresh...



Assim ficou muito mais simples.

Demo

Recents Countries — View Country Info, ...


COUNTRY NAME **France**

City Name **Paris**

Attraction Name		Trips		
Eiffel Tower		5	UPDATE	NEW TRIP
Louvre Museum		1	UPDATE	NEW TRIP

Total Trips 6

City Name **Nice**

Attraction Name		Trips		
Matisse Museum		2	UPDATE	NEW TRIP

Total Trips 2


Total Attractions 3

Travel Agency

Recents Countries — View Country Info, ...

COUNTRY NAME **Brazil**

City Name **Rio de Janeiro**

Attraction Name		Trips		
Christ the Redeemer		0	UPDATE	NEW TRIP

Total Trips 0




Total Attractions 1

Travel Agency

Recents Countries — View Country Info, ...

COUNTRY NAME **China**

City Name **Beijing**

Attraction Name		Trips		
Forbidden city		0	UPDATE	NEW TRIP
Meet the Emperor		0	UPDATE	NEW TRIP
The Great Wall		0	UPDATE	NEW TRIP

Total Trips 0

Total Attractions 3

Se executamos... vemos para França, por exemplo, com suas duas cidades que nem é percebida uma diferença com a implementação com join, porque ambas as cidades têm atrações.

Mas por outro lado se escolhemos Brasil sim, vemos a diferença. Ou China, por exemplo.

E agora sim estão sendo corretamente contadas as atrações.

WITHOUT Base Tables

Agora, passemos para o caso de implementação sem tabela base.

With several Grids: nested

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

1 param(in: CountryId);

Country Name

GRID

City Name

GRID

Attraction Id	Attraction Name	Trips		
<input type="text" value="&AttractionId"/>	<input type="text" value="&AttractionName"/>	<input type="text" value="&trips"/>	<input type="text" value="&update2"/>	<input type="text" value="&newTrip"/>

Total Trips

Total Attractions

```

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

```

Grid1 and Grid2 without Base Tables

Tínhamos este web panel que implementava o mesmo join do início, mas sem tabelas base. Observemos que na tela temos apenas variáveis e não há atributos nas propriedades de nenhum dos grids.

Era nos eventos onde realizávamos, explicitamente, a carga da base de dados. Fazemos um Save as para deixar este como estava, com join. E implementar o corte de controle em outro. Já aproveitamos e retiramos o Count de attractions do primeiro Load, e fazemos a conta no segundo, para deixar tudo mais simples.

Modifiquemos agora a ação do Work With Country para invocar agora este web panel.



Demo

Grid1 and Grid2 without Base Tables

Recents Countries — View Country Info_...


COUNTRY NAME **France**

City Name **Paris**

Attraction Name		Trips		
Eiffel Tower		5	UPDATE	NEW TRIP
Louvre Museum		1	UPDATE	NEW TRIP

Total Trips 6

City Name **Nice**

Attraction Name		Trips		
Matisse Museum		2	UPDATE	NEW TRIP




Total Trips 2

Total Attractions 3

Recents Countries — View Country Info_...

COUNTRY NAME **China**

City Name **Beijing**

Attraction Name		Trips		
Forbidden city		0	UPDATE	NEW TRIP
Meet the Emperor		0	UPDATE	NEW TRIP
The Great Wall		0	UPDATE	NEW TRIP

Total Trips 0

City Name **Hong Kong**

Attraction Name		Trips		

Total Trips 0

Total Attractions 3

Executemos... e vejamos as atrações da França. E agora as da China. Percebe-se claramente o **join** e não o **corte de controle**.

With several Grids: nested

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

1 param(in: CountryId);

Country Name &CountryName				
GRID				
City Name &cityName				
GRID				
Attraction Id &AttractionId	Attraction Name &AttractionName		Trips &trips	&update2 &newTrip
Total Trips &totalTrips				
Total Attractions &totalAttractions				

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
```

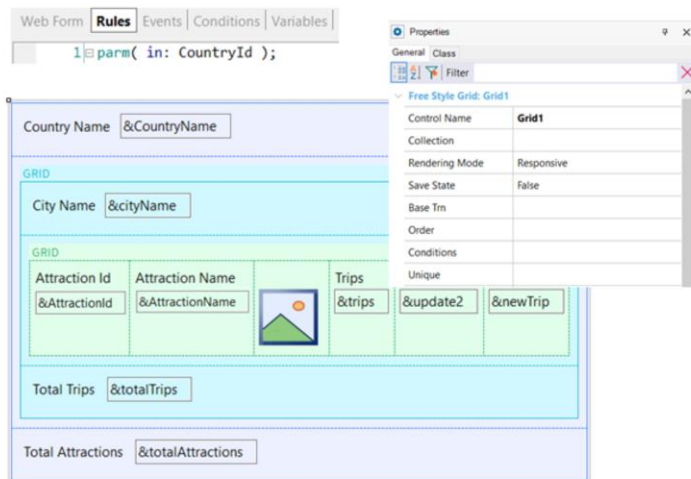
Embora o comando Load do primeiro grid dispare o evento Refresh e imediatamente o Load do segundo grid, na realidade não aninha os for eachs. É como se fosse chamada uma sub-rotina, digamos.

É como se o For each do grid aninhado fosse executado isoladamente. É por isso que GeneXus não está encontrando um join automático e é por isso que tivemos que filtrar explicitamente as atrações da cidade que ficou carregada na variável &cityName, que sim foi carregada pelo evento Load que invocou o Load do grid aninhado. Não tivemos que colocar também filtro por CountryId, já que é instanciado por vir no parâmetro.

Tenhamos isto em mente, porque será o que fará para este caso menos evidente do que poderia parecer à primeira vista.

A questão é: como fazemos para que o For each correspondente ao grid1 mude sua tabela base para Attraction e estabeleça um corte de controle por país, cidade?

With several Grids: nested



```

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

```

Grid1 and Grid2 without Base Tables


Não será adicionando Base Trn ou Order às propriedades do Grid 1 (porque se o fizéssemos, transformaríamos a implementação em uma com tabela base), mas sim ao For each explícito do evento Load do Grid1.


Portanto, parece óbvio que a primeira coisa será modificar o primeiro for each para que tenha como Base Transaction a Attraction...

E também pareceria evidente que devemos colocar cláusula order para estabelecer os critérios de agrupamento pelos quais queremos que seja estabelecido o corte de controle em relação ao for each do Grid2.


Demo

Web Panel ViewCountryInfo_relatedCopy2Copy1 Navigation Report

Name:  ViewCountryInfo_relatedCopy2Copy1
 Description: View Country Info_related Copy2 Copy1

Environment:  Default (C#)
 Spec. Version: 17_0_3-148529
 Form Class: HTML
 Program Name: ViewCountryInfo_relatedCopy2Copy1
 Parameters: in: CountryId

Warnings




 spc0038 There is no index for order **AttractionName**; poor performance may be noticed in group starting at line 26.

Event Grid Load

For Each Attraction (Line: 14)

Order: CountryId, CityId
 Index: IATTRACTION1

Navigation: Start from: CountryId = @CountryId
 filters: Loop while: CountryId = @CountryId
 Join location: Server

 Attraction (AttractionId) INTO CityId
 Country (CountryId) INTO CountryName
 CountryCity (CountryId, CityId) INTO CityName

This is not a Control Break!

Event Grid2 Load

For Each Attraction (Line: 26)

Order: **AttractionName**
 No index

Navigation: Start from: FirstRecord
 filters: Loop while: NotEndOfTable
 Constraints: CountryId = @CountryId
 CityName = @CityName
 Join location: Server

 Attraction (AttractionId) INTO CityId CountryId AttractionPhoto.Uri
 AttractionPhoto AttractionName AttractionId
 CountryCity (CountryId, CityId) INTO CityName
 count TripDate_1.navigation (AttractionId)

Formulas

No entanto, se observamos a lista de navegação...

Aparece algo estranho, e é que embora cada for each aparentemente faça o que deve fazer, não escolheu como order de cada um, o mesmo, para utilizar um único índice e tornar a leitura eficiente. Algo não está certo.

Evidentemente não entendeu que deverá fazer um corte de controle.



View Country Info_related Copy1 x +

trialapps3.genexus.com/ld111ff2ece4700f8312f2a98d30e5285c/viewcountryinfo_relatedcopy1.aspx?CountryId=2

Recents Countries — View Country Info_...


COUNTRY NAME **France**

City Name **Paris**

Attraction Name		Trips		
Eiffel Tower		5	UPDATE	NEW TRIP
Louvre Museum		1	UPDATE	NEW TRIP

Total Trips 6

City Name **Nice**

Attraction Name		Trips		
Matisse Museum		2	UPDATE	NEW TRIP

Total Trips 2

Total Attractions 3

E constatamos isto se executamos.

Vejamus que para as atrações da França sai duas vezes Paris, que coincide com as duas atrações de Paris que existem.



COUNTRY NAME China			
City Name Beijing			
Attraction Name		Trips	
Forbidden city		0	UPDATE NEW TRIP
Meet the Emperor		0	UPDATE NEW TRIP
The Great Wall		0	UPDATE NEW TRIP
Total Trips		0	
City Name Beijing			
Attraction Name		Trips	
Forbidden city		0	UPDATE NEW TRIP
Meet the Emperor		0	UPDATE NEW TRIP
The Great Wall		0	UPDATE NEW TRIP
Total Trips		0	
City Name Beijing			
Attraction Name		Trips	
Forbidden city		0	UPDATE NEW TRIP
Meet the Emperor		0	UPDATE NEW TRIP
The Great Wall		0	UPDATE NEW TRIP
Total Trips		0	
Total Attractions		0	



COUNTRY NAME Brazil			
City Name Rio de Janeiro			
Attraction Name		Trips	
Christ the Redeemer		0	UPDATE NEW TRIP
Total Trips		0	
Total Attractions		1	

Para as da China, sai três vezes Beijing, que coincide com as três atrações de Beijing que existem.

E para Brasil apenas uma vez, coincidindo com as atrações do Rio que existem.

O que está acontecendo?

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

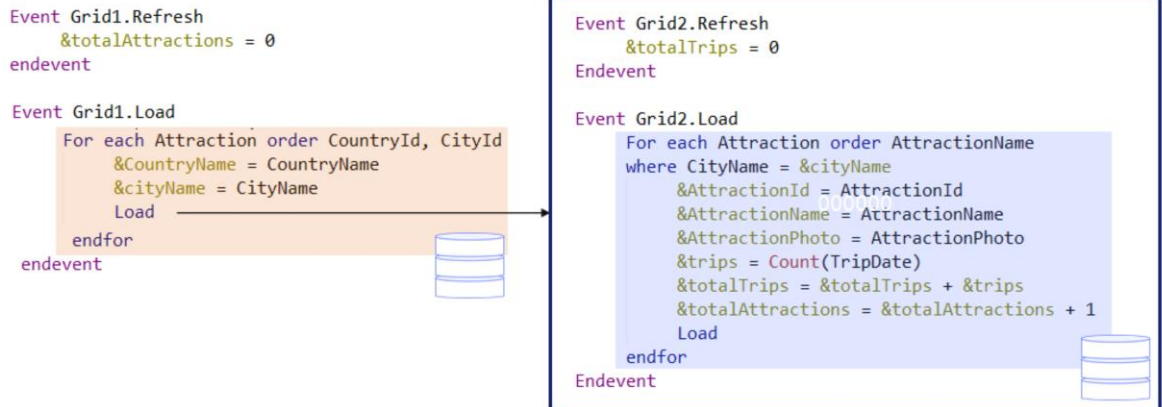
```



Evidentemente não está encontrando que deve fazer um corte de controle.


É que, como já antecipamos, não aninha **realmente** as navegações.

With several Grids: nested



É como se fossem dois for eachs independentes, só que a partir de um é invocada a execução do outro, mas por meio de duas consultas distintas à base de dados.

With several Grids: nested

Country Name &CountryName					
GRID					
City Name &cityName					
GRID					
Attraction Id &AttractionId	Attraction Name &AttractionName		Trips &trips	&update2	&newTrip
Total Trips &totalTrips					
Total Attractions &totalAttractions					

Grid1 and Grid2 **without** Base Tables



Control Break could not be implemented
between nested grids

Isto equivale a dizer que não podemos **realmente** implementar um corte de controle entre dois grids aninhados sem tabelas 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

```

A solução que temos no momento é utilizar, para o primeiro For each, a cláusula unique. Ou seja, que da tabela Attraction se vários registros têm o mesmo país e cidade, se mantenha com apenas um deles. E para este, carregue as variáveis e execute o evento Refresh e imediatamente Load do grid2, que executará seu for each como se fosse completamente independente do anterior. E é justamente por isso que desta vez sim nos permitirá usar a cláusula unique.

Demo

▲ spc0038 There is no index for order AttractionName: poor performance may be noticed in group starting at line 27.

Event Grid1.Load

For Each Attraction (Line: 14)

Order: CountryId , CityId
 Index: IATTRACTION1
 Unique: CountryName , CityName
 Navigation Start from: CountryId = @CountryId
 filters: Loop while: CountryId = @CountryId
 Join location: Server

Attraction (AttractionId) INTO CityId
 Country (CountryId) INTO CountryName
 CountryCity (CountryId , CityId) INTO CityName

Event Grid2.Load

For Each Attraction (Line: 27)

Order: AttractionName
 No index
 Navigation Start from: FirstRecord I
 filters: Loop while: NotEndOfTable
 Constraints: CountryId = @CountryId
CityName = &cityName
 Join location: Server

Attraction (AttractionId) INTO CityId CountryId AttractionPhoto Uri
 AttractionPhoto AttractionName AttractionId
 CountryCity (CountryId , CityId) INTO CityName
 count(TripDate) navigation (AttractionId)

Se agora vemos a lista de navegação... parece que assim vai funcionar.

Demo

Travel Agency 

Recents Countries — View Country Info...

COUNTRY NAME **China**

City Name **Beijing**

Attraction Name	Trips		
Forbidden city	0	UPDATE	NEW TRIP
Meet the Emperor	0	UPDATE	NEW TRIP
The Great Wall	0	UPDATE	NEW TRIP

Total Trips 0
Total Attractions 3

Travel Agency

Recents Countries — View Country Info...

COUNTRY NAME **France**

City Name **Nice**

Attraction Name	Trips
Matisse Museum	2

Total Trips 2
Total Attractions 3

City Name **Paris**

Attraction Name	Trips		
Eiffel Tower	5	UPDATE	NEW TRIP
Louvre Museum	1	UPDATE	NEW TRIP

Total Trips 6
Total Attractions 3

Travel Agency 

Recents Countries — View Country Info...

COUNTRY NAME **Brazil**

City Name **Rio de Janeiro**

Attraction Name	Trips		
Christ the Redemmer	0	UPDATE	NEW TRIP

Total Trips 0
Total Attractions 1

Executemos...

Conseguimos.

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
```



Portanto, no momento, nos está sendo muito mais fácil implementar um corte de controle, quando os grids **têm tabela base**.

A rigor, somente nesse caso se tratará de um **verdadeiro corte de controle**.

No segundo caso, quando os grids não possuem tabela base, estamos apenas simulando, mas na realidade haverá duas consultas independentes à tabela Attraction, e não apenas uma que resolva tudo, como acontece no verdadeiro corte de controle.

Convidamos você a testar tudo o que vimos.

GeneXus™

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications