# Web Screens

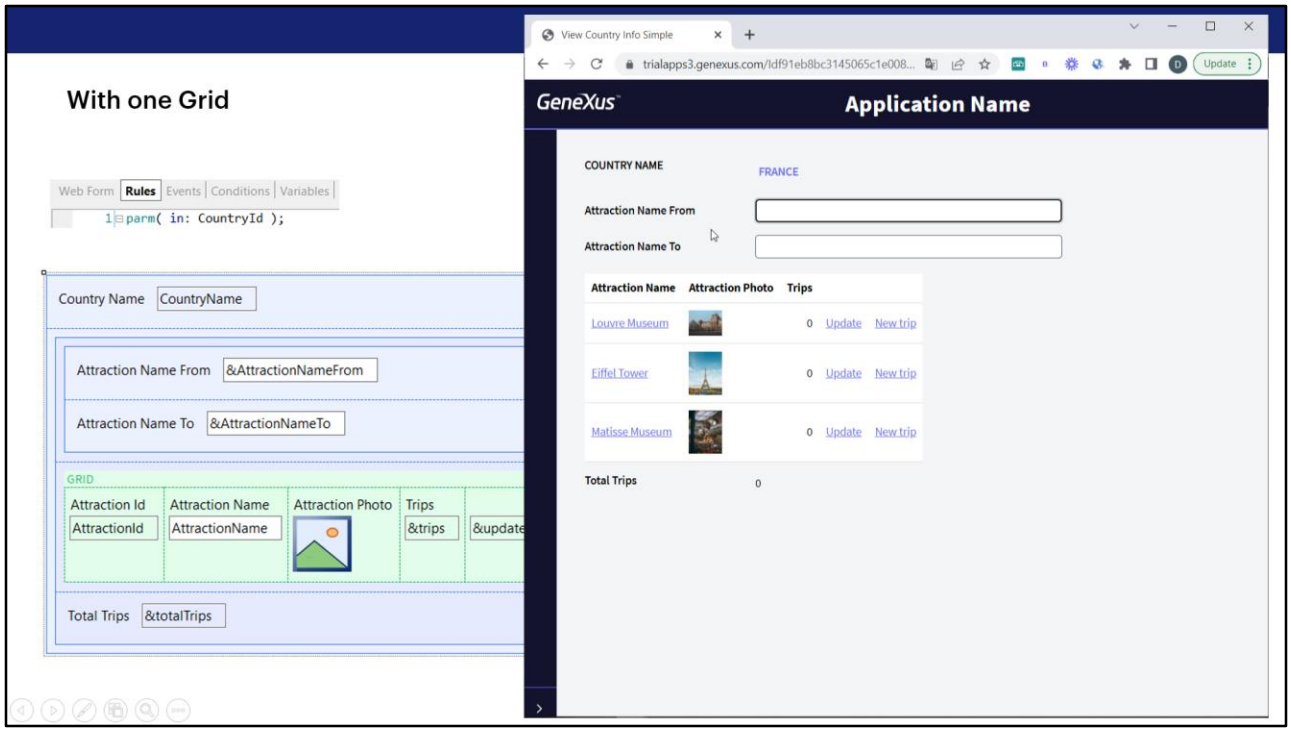
*Base Table and Navigation of Web Panel with ONE Grid.*

GeneXus™

GeneXus

## Web Panel with ONE Grid

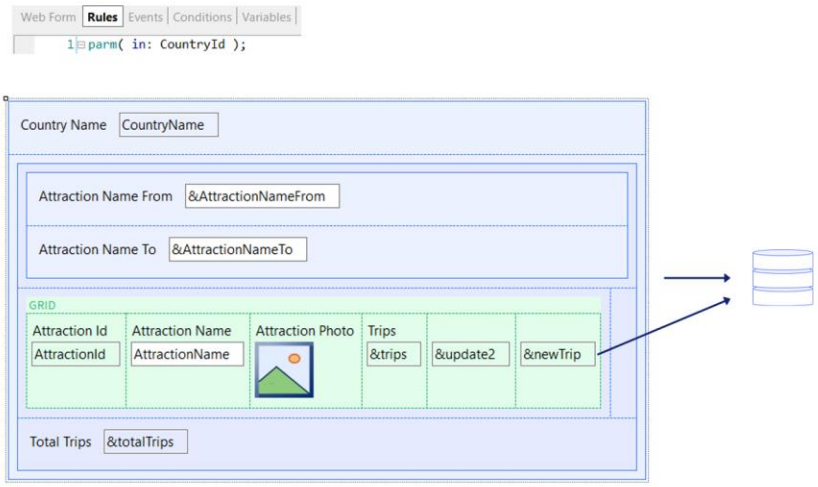
Now let's see the case of a web panel with one grid.




Here we are receiving by parameter the identifier of a country. Also, we're showing in a grid all its tourist attractions, and in the fixed part the name of that country.

GeneXus

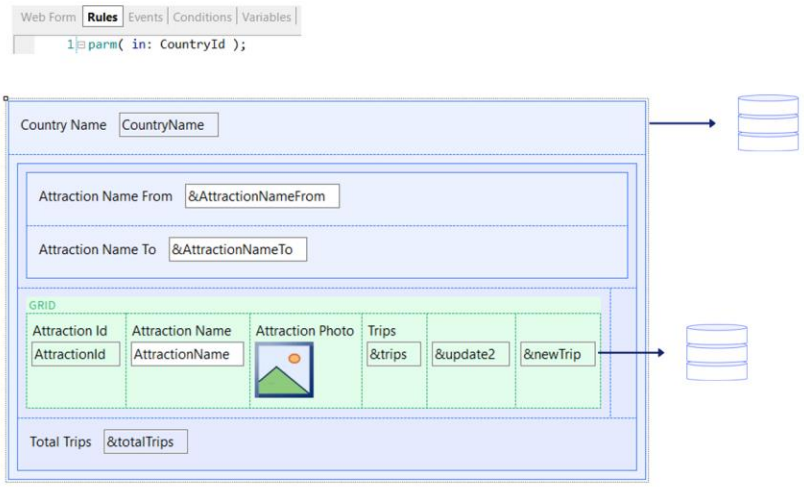
## With one Grid




When the web panel has at most one grid, we can say that if there is a base table, there is no difference between talking about the base table of the web panel and about the base table of the grid. Why?

GeneXus

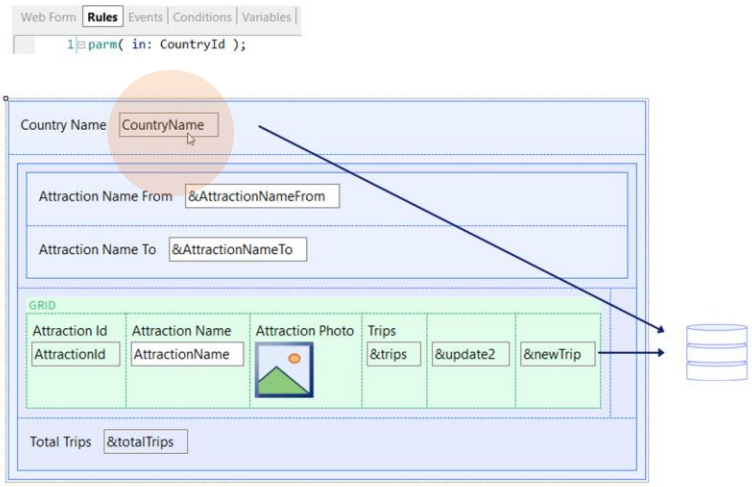
## With one Grid




Because it will not be possible, as it happens in the case of plain Panels – that is, those developed to run with Angular, for example, or with Android or Apple– that there is a navigation for the fixed part of the panel and a different one for the grid.

GeneXus

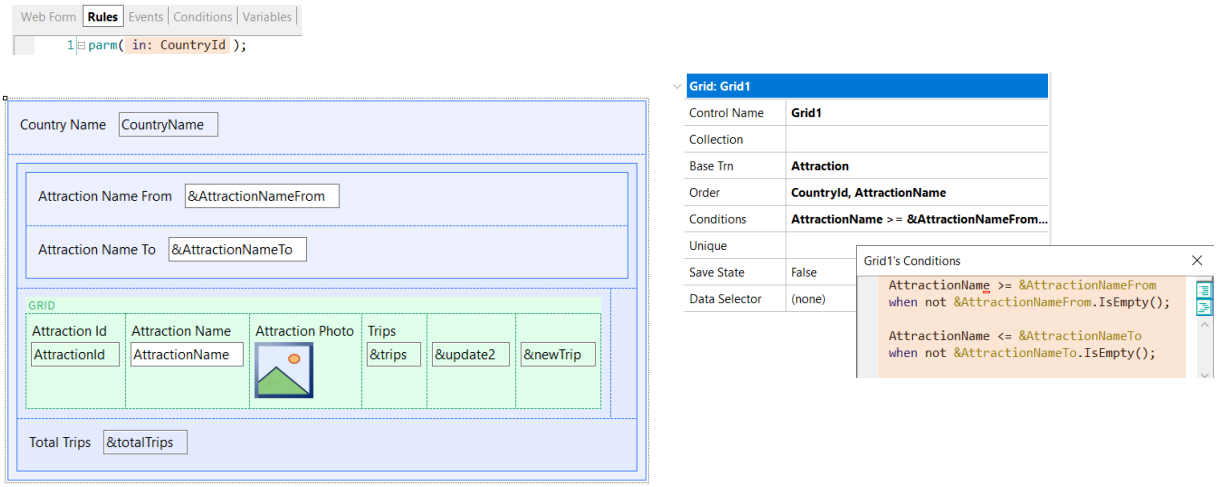
## With one Grid




For web panels with one grid, if there is a base table, it will be unique, and what should happen if it is correctly implemented is that the attributes of the fixed part must have the same value for all the records of the base table that are going to be loaded into the grid, because otherwise they should be in the grid and not in the fixed part!

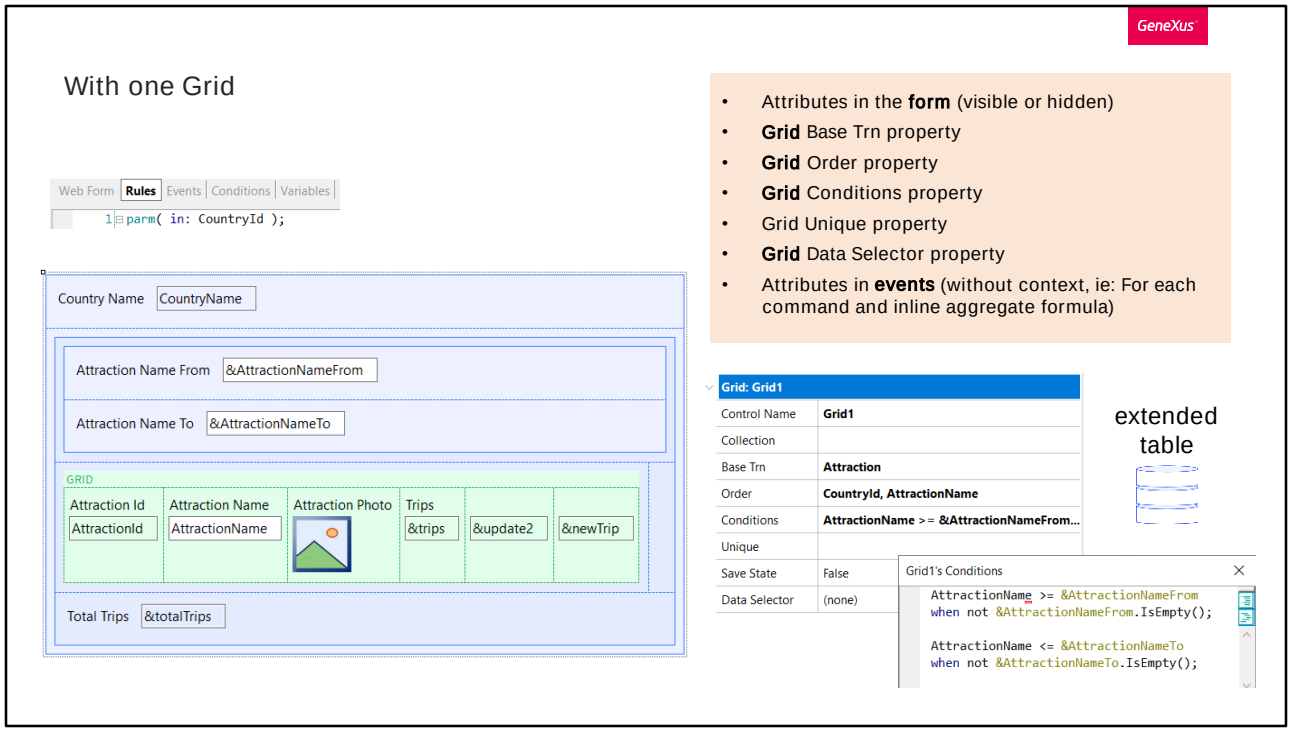
GeneXus

## With one Grid




Here all the attractions that pass the conditions' filters and belong to the country received by parameter will be loaded into the grid. Therefore, for all of them, the CountryName will be the same and that is why it is in the fixed part.

Note that we will be receiving by parameter in the CountryId attribute that DOESN'T PARTICIPATE in determining the possible base table, but it does act as a filter on the navigations that GeneXus finds, as we saw in the previous case.

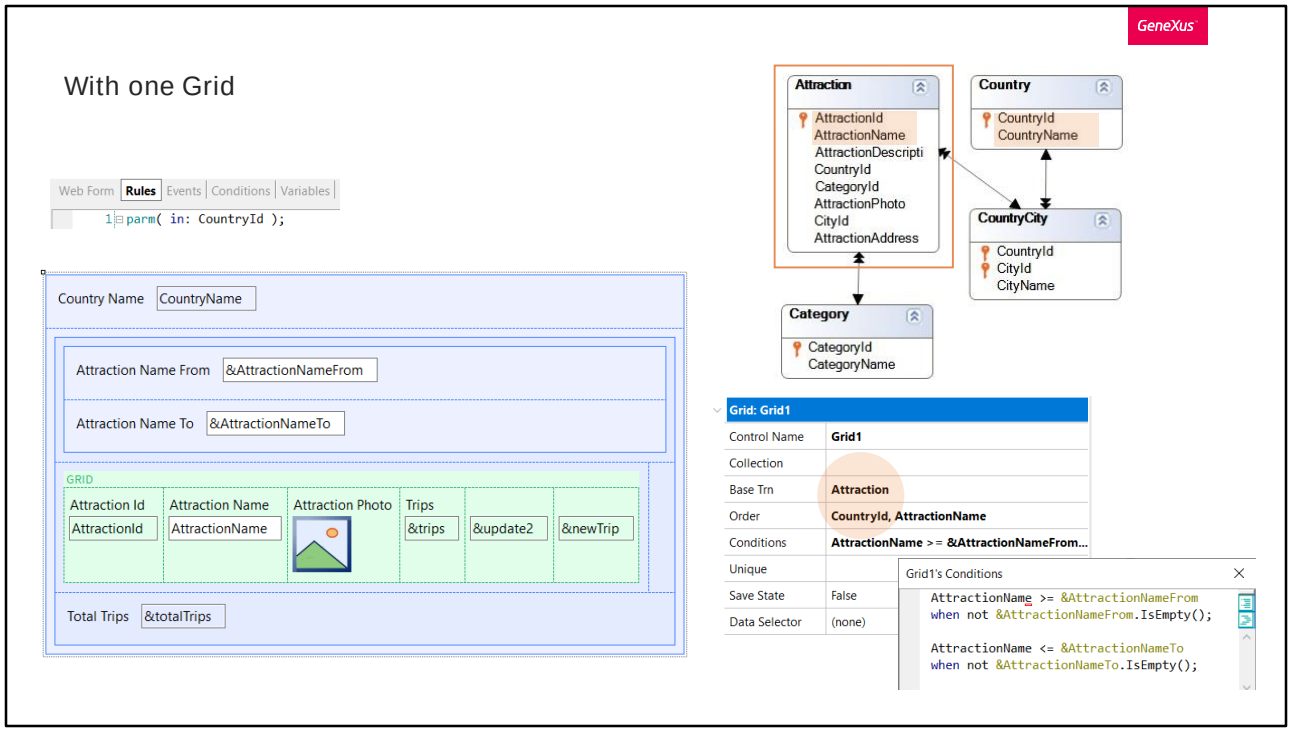



What parts of the web panel are taken into account to determine the base table? Obviously, when there is a grid, in addition to the places we had already seen for the previous case, everything related to the grid is added.

Therefore, if attributes appear there or in the Order properties, conditions, Unique, Data Selector and, of course, if a base transaction is specified in the corresponding property, then the grid (and therefore the Web panel) will have an implicit base table.

Obviously, if there is a base transaction, all the attributes of the form and those that are in “loose” events must belong to its extended table. Otherwise, GeneXus will warn us about which attributes will not be “attainable” and will have no value.

If no base transaction was specified, then, as in the case of the web panel without a grid, GeneXus will find it by looking for the minimum extended table containing all the attributes that appear in the places we defined. Its base table will be the selected base table.




Before looking at the events, we can say that the base table will be Attraction because there is a base transaction. And we confirm that all these other attributes belong to its extended table. So, if there wasn't a single event programmed, we would still know that the Attraction table will be run through, ordering it by CountryId and AttractionName, filtering by the attractions whose AttractionName meets these two conditions, and also whose country matches the one received by parameter.

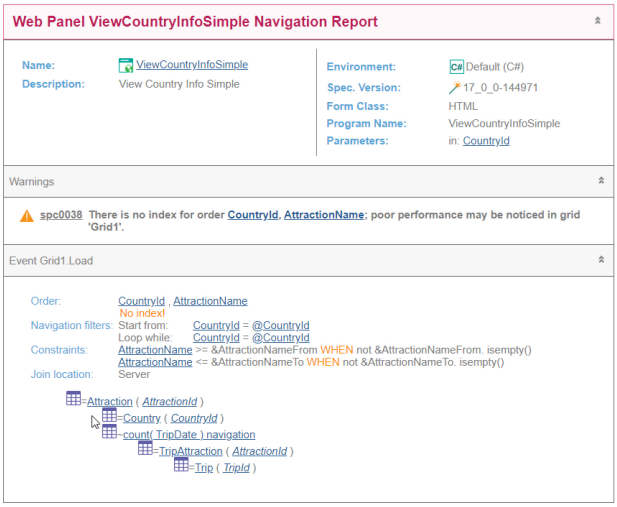
Let's say it one last time: for web panels, if there is a grid, the attributes of the fixed part correspond to the navigation of the grid. Then we will see that if there are several grids, they will correspond to the navigation of one of them.

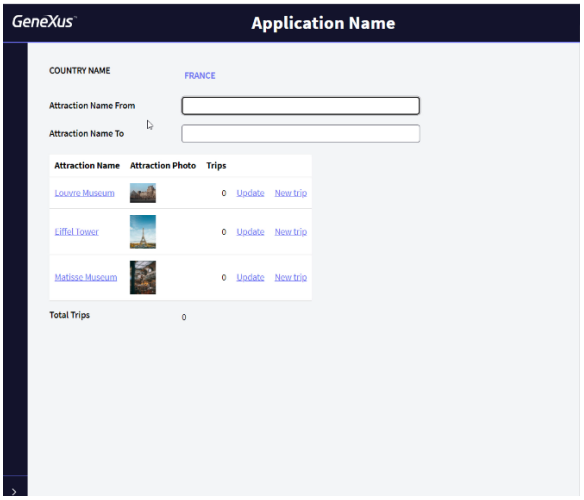
If we were not receiving data in the CountryId attribute, then, since attractions from different countries will be loaded into the grid, what value will the CountryName attribute take on screen? One could easily suspect that it will be either the first or the last one of the records loaded into the grid.

Let’s try it.

GeneXus

## With one Grid

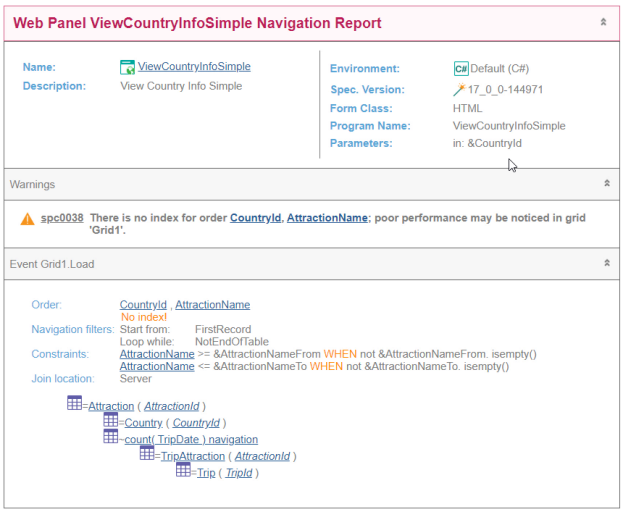


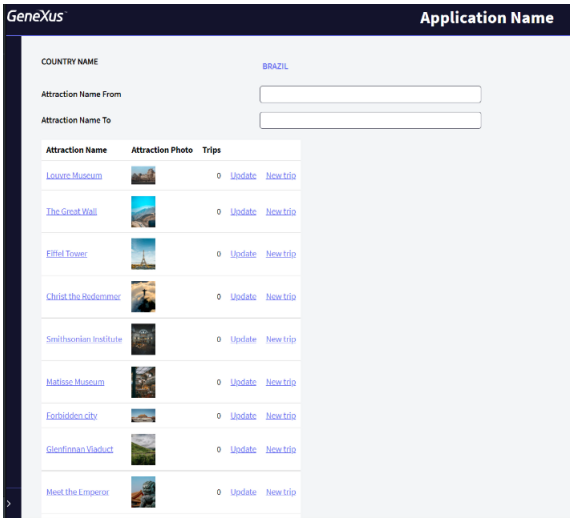




Here we have the web panel, correctly programmed. First, let's see its navigation list.
We see that, indeed, there is a base table, which is Attraction. We see it clearly identified at the level of the Load event of the grid.
And now we will run it before modifying the parameter. From this other web panel we invoke ours.

GeneXus

With one Grid



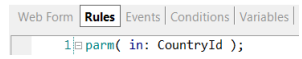

Now let's make changes to our panel so that instead of receiving data in the CountryId attribute, it is received in the &CountryId variable. And so there is no automatic filter.

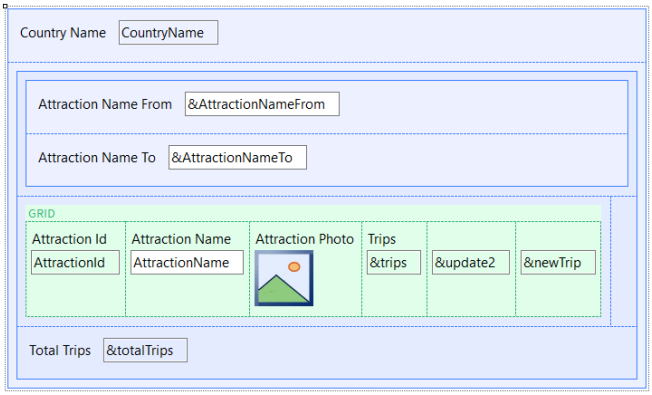
Let's look at the navigation list. We see that now there will be no filter.

Let’s see it at runtime. Note that it loaded as country name that of the first of the records, not of the last one.

GeneXus

## With one Grid




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

Event Grid1.Refresh
     &totalTrips = 0
Endevent

Event Start
     &newTrip = "New trip"
     &update2 = "UPDATE"
     CountryName.ForeColor = RGB(147,4,55) //DarkBase
     CountryName.FontBold = True
Endevent

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

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

Well, let's leave the parameter as we had it.

If we now look at the events, we notice that obviously the Load event was programmed already knowing that the grid would have a base table. If we analyze as GeneXus does which attributes will participate in determining the base table, we see that they are all “loose.”

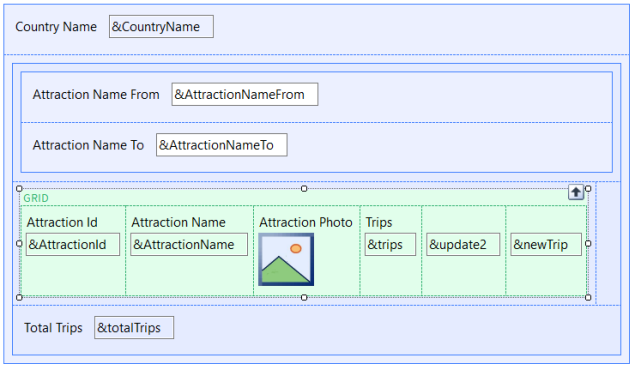
Clearly TripDate does not. It only participates in the determination of the table to be navigated by the Count formula. Of course, and just like with the web panel in general, if the grid has a base table then this Count formula can also have an implicit condition, which in this case is that it will count the trips of the tourist attraction that is about to be loaded.

GeneXus

## With one Grid

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

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




```
Event Grid1.Load
     For each
          &CountryName = CountryName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          Load
          &totalTrips = &totalTrips + &trips
     endfor
Endevent

Event Grid1.Refresh
     &totalTrips = 0
Endevent

Event Start
     &newTrip = "New trip"
     &update2 = "UPDATE"
     &CountryName.ForeColor = RGB(147,4,55) //DarkBase
     &CountryName.FontBold = True
Endevent

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

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

As we've seen before, it is not enough to replace all the attributes we show with variables everywhere if we forget to do it in one of these places. For example, here we made a Save as from the previous panel, to implement it without a base table.

We changed all the attributes of the form to variables, and also those same attributes in the events.

In fact, we programmed the Load event of the grid by specifying the For Each command to run through the Attraction table and load all the variables of the grid (and also the one of the fixed part, &CountryName), thinking that we are programming the web panel without a base table.

We did not specify a base transaction to the For each, but we could have. We know that it will be filtered by CountryId because it is being received by parameter and also that this attribute of the parm rule does not participate in determining the base table.

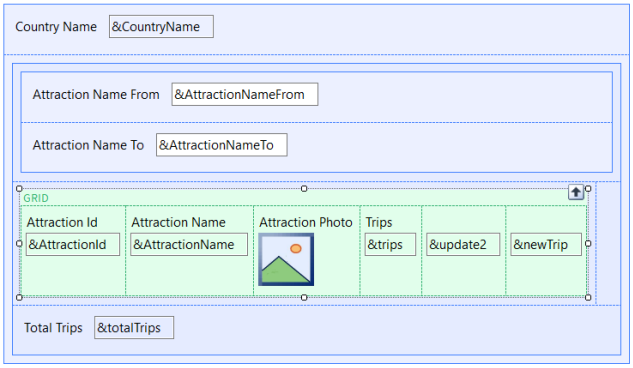
But where are we specifying the filters by attraction name? We forgot about them. We forgot to change the properties of the grid as well! And the properties of the grid do determine the base table.

GeneXus

## With one Grid

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

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




```
Event Grid1.Load
    For each
        &CountryName = CountryName
        &AttractionId = AttractionId
        &AttractionName = AttractionName
        &AttractionPhoto = AttractionPhoto
        &trips = Count(TripDate)
        Load
        &totalTrips = &totalTrips + &trips
    endfor
Endevent

Event Grid1.Refresh
```

**Grid: Grid1**

| | |
|---|---|
| Control Name | **Grid1** |
| Collection | |
| Base Trn | **Attraction** |
| Order | **CountryId, AttractionName** |
| Conditions | **AttractionName >= &AttractionNameFrom...** |
| Unique | |
| Save State | False |
| Data Selector | (none) |

```
endevent

Event &AttractionN
    ViewAttractionFromScratch(&AttractionId)
Endevent
```

Note that these three properties that we omitted ARE ESTABLISHING A BASE TABLE.

## With one Grid

Warnings

spc0038 There is no index for order CountryId, AttractionName; poor performance may be noticed in group starting at line 2.

Event Grid1.Load

Order: CountryId , AttractionName
No index!
Navigation filters: Start from: CountryId = @CountryId
Loop while: CountryId = @CountryId
Constraints: AttractionName >= &AttractionNameFrom WHEN not &AttractionNameFrom. isempty()
AttractionName <= &AttractionNameTo WHEN not &AttractionNameTo. isempty()
Join location: Server

-Attraction ( *AttractionId* )
-Country ( *CountryId* )

Break Attraction (Line: 2)

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

-Attraction ( *AttractionId* )
-Country ( *CountryId* )
-count( TripDate ) navigation
-TripAttraction ( *AttractionId* )
-Trip ( *TripId* )

```
Event Grid1.Load
     For each
          &CountryName = CountryName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          Load
          &totalTrips = &totalTrips + &trips
     endfor
Endevent

Event Grid1.Refresh
     &totalTrips = 0
Endevent

Event Start
     &newTrip = "New trip"
     &update2 = "UPDATE"
     &CountryName.ForeColor = RGB(147,4,55) //DarkBase
     &CountryName.FontBold = True
Endevent

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

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

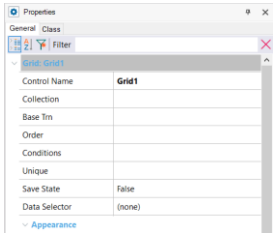
As a matter of fact, let's see what the navigation list says. Precisely, it found the Attraction base table, so it will execute the Load N times, filtering by country received by parameter and also by the variables according to the conditions we specified in the grid.
And for each of those records, it will execute the For Each of the Load, making a control break, because the base table is the same, Attraction.

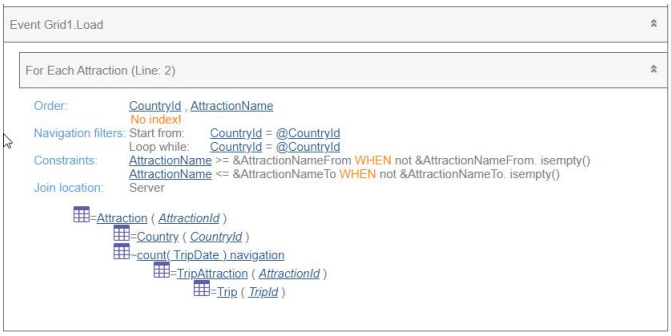
All that because we forgot to check the properties of the grid!

GeneXus

With one Grid

```
Event Grid1.Load
    For each Attraction
        order CountryId, AttractionName
        where AttractionName >= &AttractionNameFrom when not &AttractionNameFrom.IsEmpty()
        where AttractionName <= &AttractionNameTo when not &AttractionNameTo.IsEmpty()
        &CountryName = CountryName
        &AttractionId = AttractionId
        &AttractionName = AttractionName
        &AttractionPhoto = AttractionPhoto
        &trips = Count(TripDate)
        Load
        &totalTrips = &totalTrips + &trips
    endfor
Endevent
```


So, what we must do is bring these properties to the For Each command.
First the base transaction...
Then the order...
And finally the conditions as Where clauses...
We save. It is causing this error because there shouldn't be a semicolon in the syntax of the For Each. We save again. OK.
We check the navigation list.
This time, we managed to ensure that the web panel has no base table.

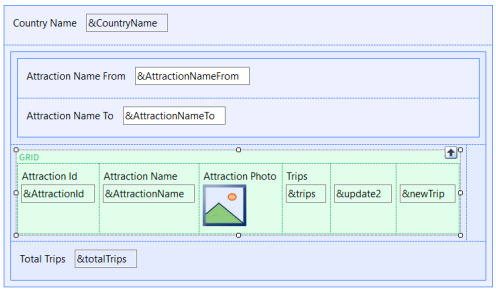
Note that if we placed the Where clauses as Conditions, but not of the grid but as general conditions of the Web panel, as it happened in the case of a panel without a grid, the attributes of these conditions do not participate in determining a base table. Here we can see it clearly. The navigation list is not different from the previous one.

The problem with the general conditions is that they are going to be applied as filters on the entire web panel navigation that involves its attributes, and sometimes we are not interested in that. Let's think, for example, in the case of having multiple grids.

GeneXus

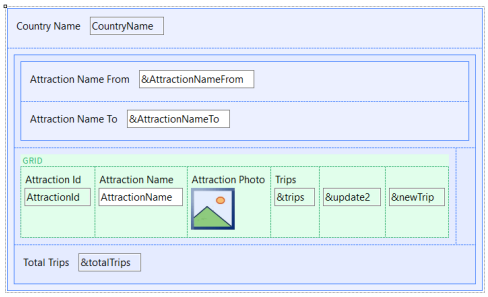
## With one Grid

```
Event Grid1.Load
     For each Attraction
          order CountryId, AttractionName
          &CountryName = CountryName
          &AttractionId = AttractionId
          &AttractionName = AttractionName
          &AttractionPhoto = AttractionPhoto
          &trips = Count(TripDate)
          Load
          &totalTrips = &totalTrips + &trips
     endfor
Endevent
```




Only load an attraction if &trips > 1 ?

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




In general, it is better practice to implement web panels with a base table, because it saves us all this work. Here we had to establish the For Each, assign the values of the attributes to the variables and specify the Load command to load each line.
On the other hand, in the version with a base table, we can see that everything was much simpler.

But what would we do if we only wanted to load attractions that are in more than one trip?

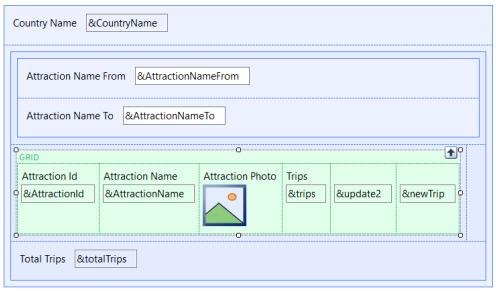
GeneXus

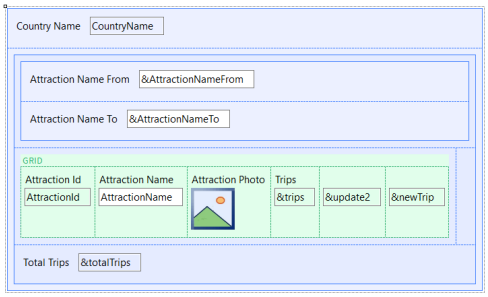
With one Grid

```
Event Grid1.Load
    For each Attraction
        order CountryId, AttractionName
        &CountryName = CountryName
        &AttractionId = AttractionId
        &AttractionName = AttractionName
        &AttractionPhoto = AttractionPhoto
        &trips = Count(TripDate)
        if &trips > 1
            Load
        endif
        &totalTrips = &totalTrips + &trips
    endfor
Endevent
```

Only load an attraction if &trips > 1 ?

```
Event Grid1.Load
    &trips = Count(TripDate)
    If &trips > 1
        Load
    endif
    &totalTrips = &totalTrips + &trips
Endevent
```




This is very simple, in the web panel without a base table: we would condition the Load command to the value of the &trips variable. We only load a line on the grid if the &trips variable is greater than 1. Otherwise, we don't load it.

What about the web panel with a base table? It would seem that we can't do it here, given that the Load is executed when the grid line is about to be loaded. Also, it would seem that we have no way of stopping the load, because the Load command that was explicit in the other one, here is implicit; GeneXus places it in its source code, but not us. However, we can do it. If we program the Load explicitly, then GeneXus understands that it should not place its own. We can say it leaves it in our hands.

Let's try it.

GeneXus™

training.genexus.com
wiki.genexus.com