Descriptions instead of codes

GeneXus

Applications

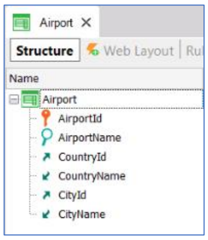
- Need for a friendly user interface (UI)

  Simple and fast data entry → Input Type

One of the challenges facing application developers today is to develop a user-friendly user interface.

Among other things, a friendly user interface implies that data entry should be as simple and fast as possible. GeneXus provides configurable properties to simplify data entry and help design more attractive interfaces. One of them is the InputType property.

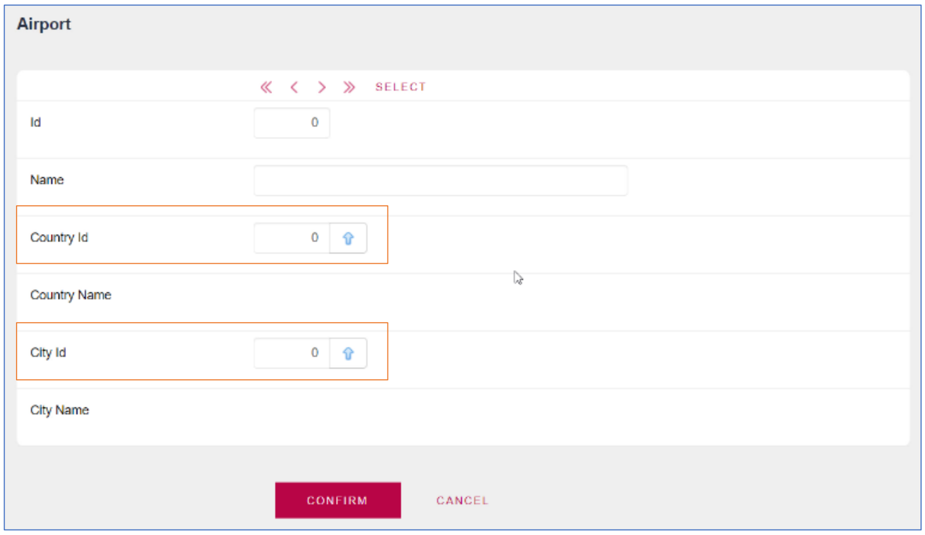
Its function is to replace what the user needs to enter in an attribute with a more user-friendly value.


Airport ✕

Structure | Web Layout | Ru

Name

- Airport
  - AirportId
  - AirportName
  - CountryId
  - CountryName
  - CityId
  - CityName

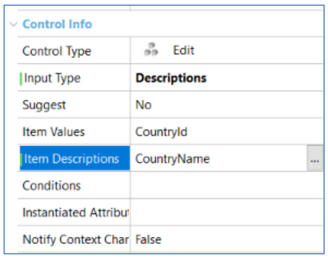

Airport

Id	0
Name	
Country Id	0
Country Name	
City Id	0
City Name	

For example, consider an application for a travel agency, which has a transaction to enter the airports and each of them has a country and a city.

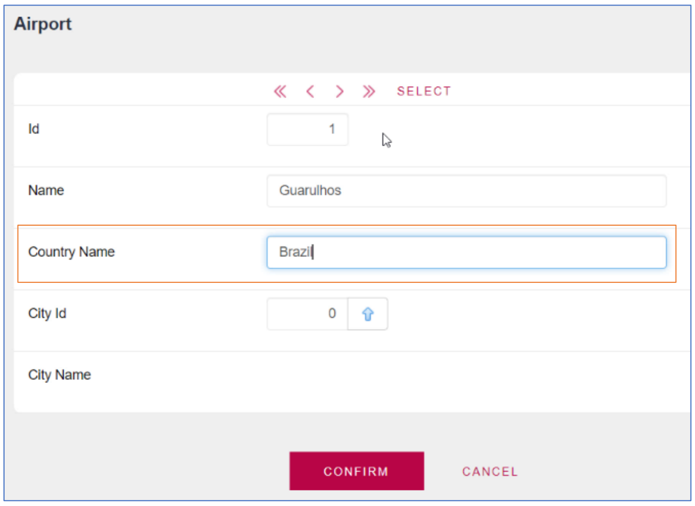
If we declare the transaction as it is defined, by default the form will be this one, where we will have to enter a code for the country and another code for the city.

Entering data using a code is not the best experience for a user. Even though GeneXus automatically provides a form prompt where we can search by name, we could let the user enter the name of the desired country and/or city and have the program automatically search and assign the corresponding code.



Control Info

Control Type	Edit
Input Type	**Descriptions**
Suggest	No
Item Values	CountryId
Item Descriptions	CountryName
Conditions	
Instantiated Attribu	
Notify Context Char	False



Airport

Id	1
Name	Guarulhos
Country Name	Brazil
City Id	0
City Name	

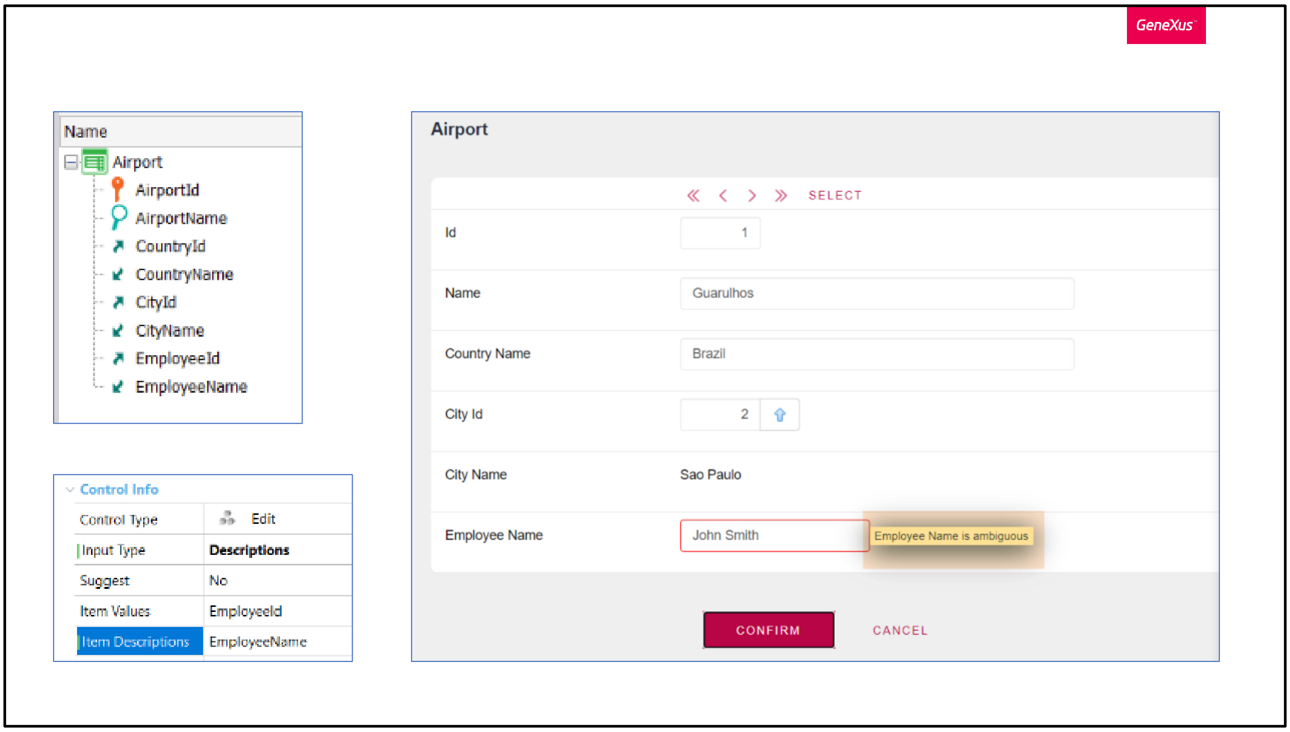
We achieve this by changing the value of the InputType property of the attribute to the descriptions value, as long as the control type is edit.

We will do it with CountryId, by changing from Values to Descriptions, and it will enable new properties. The one we must fill in is Item descriptions.

Since what we want is to enter the name instead of the key, in the Item descriptions property we select CountryName.

At runtime, it seems that CountryId has disappeared and only CountryName is left in editable mode, but actually the attribute that shows those values is not CountryName but CountryId “in disguise.”
If, for example, we want to select the country Brazil, we type it and internally a search is performed in the Country table to retrieve the corresponding code, which in this case will be “1,” but this will be transparent for the user. If we go to the Web Layout of the transaction, we see that the attribute we have is indeed CountryId and not CountryName.

It is important that the attribute we choose for the Item Descriptions property is a candidate key in the transaction to which it belongs; otherwise, an ambiguity could arise.



Name
- Airport
  - AirportId
  - AirportName
  - CountryId
  - CountryName
  - CityId
  - CityName
  - EmployeeId
  - EmployeeName

Control Info

Control Type	Edit
Input Type	**Descriptions**
Suggest	No
Item Values	EmployeeId
Item Descriptions	EmployeeName

Airport

Id	1
Name	Guarulhos
Country Name	Brazil
City Id	2
City Name	Sao Paulo
Employee Name	John Smith

Employee Name is ambiguous

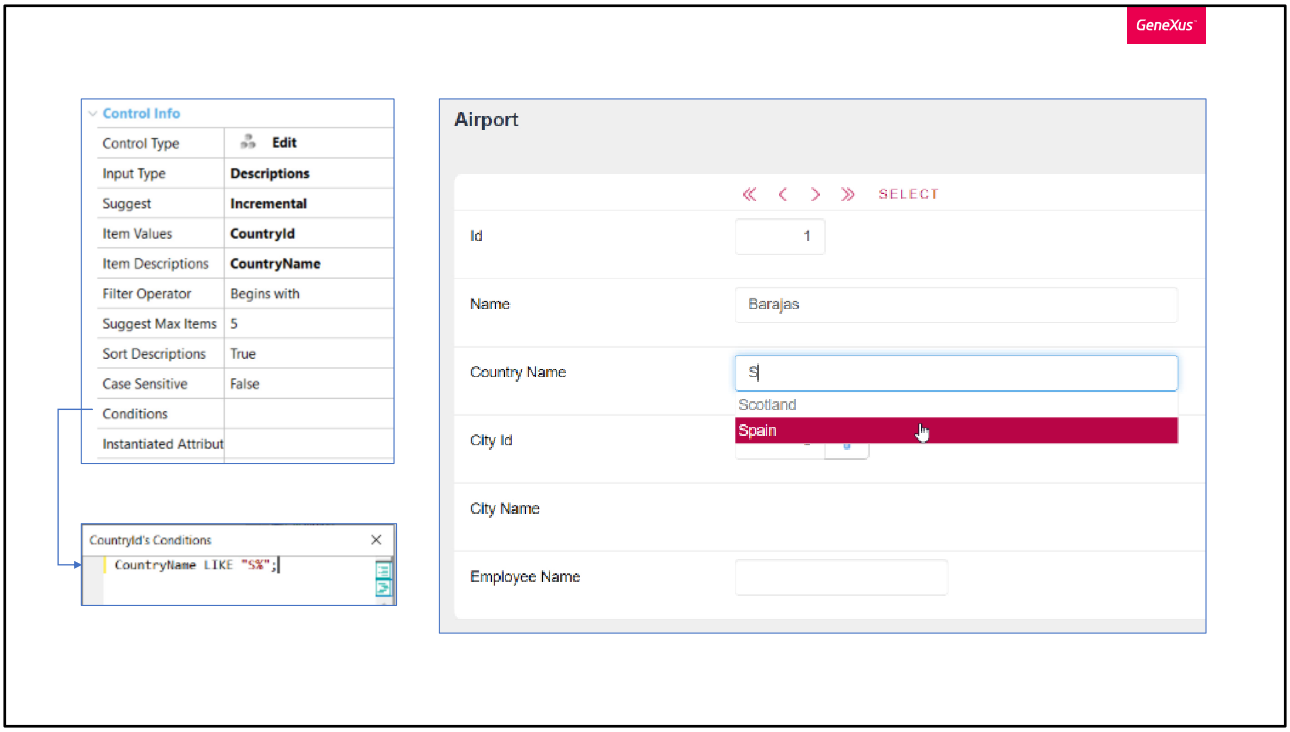
For example, let's suppose that in the Airport transaction we have to assign an employee of the agency who is responsible for the coordination of flights. To do so, we enter the EmployeeId and EmployeeName attributes of the Employee transaction. If we change the InputType property of EmployeeId to Descriptions, and in Item Descriptions we place EmployeeName, to enter the employee's name instead of the key, we may get an ambiguity error; since EmployeeName is probably not a candidate key, there can be two employees with the same name.

When it is run, we can see that the output shows a warning about this situation. If we enter a name and there is more than one record with the same name, an error will be displayed because it will not know which one to keep.

Therefore, it is always important to take this into account. Another consideration is that the attribute entered in Item Descriptions must be in the transaction where we use this property; otherwise, an error will be thrown. And it must also be of char or varchar domain.

We can configure this Input Type property and its associated Item Descriptions at the attribute level as we have just seen. If it is applied in this way, it will have this behavior in each transaction where the attribute appears, except in its own transaction—in this case, Country. However, we can also configure these properties directly in the Web Layout of a transaction, and it would only apply to that form; in this case, we must take into account that the dynamism is lost and that you will have to manually change the name of the label and delete the description attribute that will no longer be necessary. We can also use this functionality in any Panel or Web Panel we have.

This behavior is similar to when we configure an attribute that is a foreign key, with the Dynamic Combo Box type. The difference here is that we cannot type the desired value but we choose it among the displayed options.



Control Info

Control Type	Edit
Input Type	**Descriptions**
Suggest	**Incremental**
Item Values	**CountryId**
Item Descriptions	**CountryName**
Filter Operator	Begins with
Suggest Max Items	5
Sort Descriptions	True
Case Sensitive	False
Conditions	
Instantiated Attribut	

CountryId's Conditions

```
CountryName LIKE "S%";
```

Airport

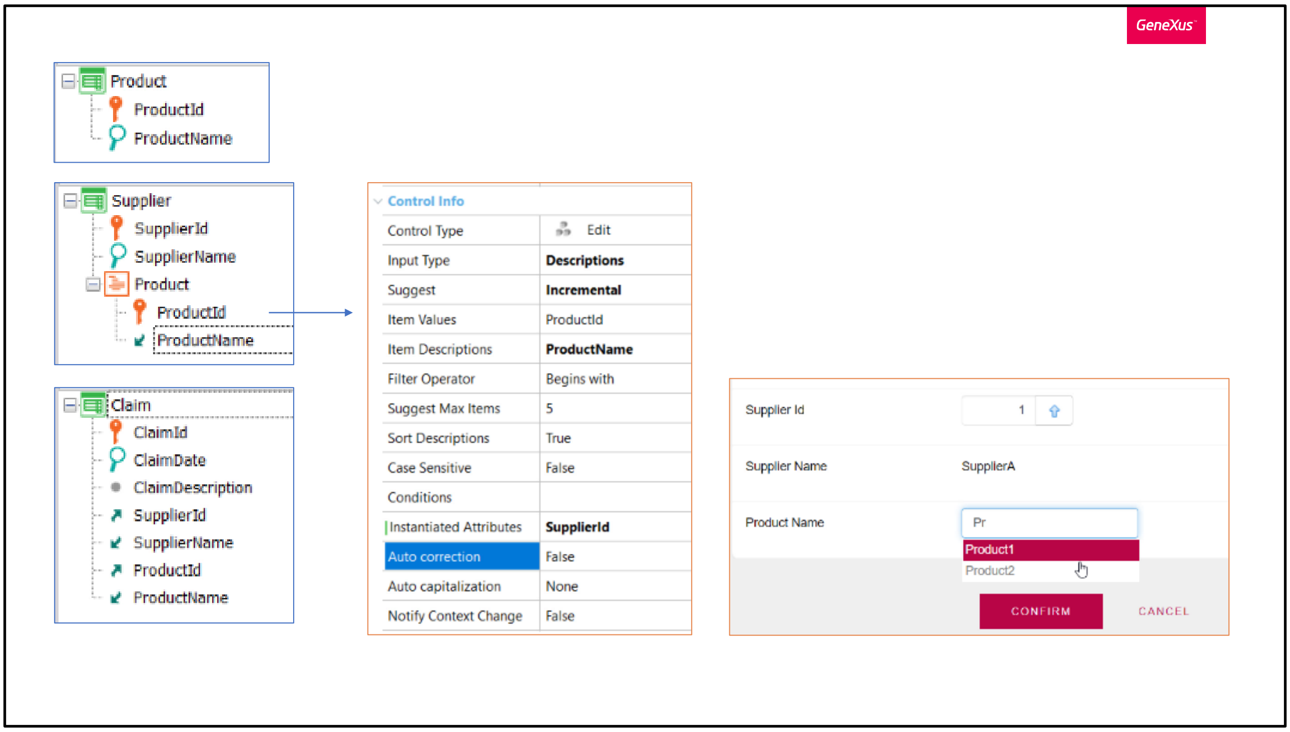
Id	1
Name	Barajas
Country Name	S
City Id	
City Name	
Employee Name	

The disadvantage of having to type the descriptive name of a record we want to select is that we may not remember it exactly or we may make a mistake when typing it. To help the user when entering values, next to the input Type property set to descriptions, the Suggest property is displayed. By default it is set to “No,” but if we change it to “Incremental” the user will be offered matching values as he/she enters letters.

Let's see it running; as we type letters, different options are suggested.
As we saw with the Input Type property, Suggest can be configured at the attribute level or at the field level.

Once the property value has been changed, new properties are enabled; for example, Filter Operator, to configure whether the records to be suggested should match from the beginning of the input or simply contain those letters. Also, if we want the filter to be case sensitive or not, or even to be able to set a limit of records to be suggested, by default it is set to 5. The latter can help reduce loading times in case there are many records.

Something else that this allows is to set conditions to this search; for example, if for some reason we only want to display the countries that begin with the letter “S” we can enter the following condition. At runtime, if we enter any other letter it will not suggest the matching records; even if we enter the name of a country that exists in the database it will not find it—that is, it neither suggests it nor allows us to register it.



- Product
  - ProductId
  - ProductName

- Supplier
  - SupplierId
  - SupplierName
  - Product
    - ProductId
    - ProductName

- Claim
  - ClaimId
  - ClaimDate
  - ClaimDescription
  - SupplierId
  - SupplierName
  - ProductId
  - ProductName

Control Info	
Control Type	Edit
Input Type	**Descriptions**
Suggest	**Incremental**
Item Values	ProductId
Item Descriptions	**ProductName**
Filter Operator	Begins with
Suggest Max Items	5
Sort Descriptions	True
Case Sensitive	False
Conditions	
Instantiated Attributes	**SupplierId**
Auto correction	False
Auto capitalization	None
Notify Context Change	False

Supplier Id	1
Supplier Name	SupplierA
Product Name	Pr

Another property that appeared is “Instantiated Attributes;” here we have another way to limit the set of possible values that will be suggested for entry. It is used to filter a set of attributes based on other attributes that cannot be directly inferred.
To clarify this, let's look at it in an example:
Suppose we are developing an application for a company, where we have a transaction to enter the products that are sold there, and another transaction to enter the suppliers of those products. In turn, for each supplier we record the products that this supplier provides and the company sells.

In addition, we have another transaction that will be used to enter claims about a certain product and supplier.

Let's suppose that each product will have a unique name, and we are interested in searching for products by typing their name, so we configure the ProductId attribute with Input Type in Descriptions and in Item Descriptions we set ProductName. We leave the Suggest property set to “Incremental” so that it suggests the records.

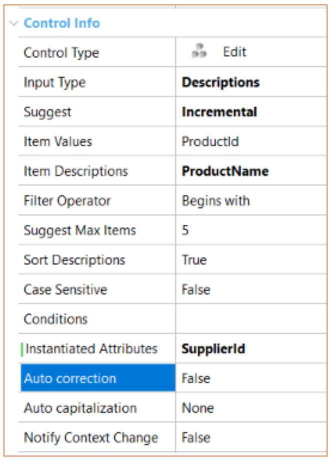
For this example, we are going to have 2 products for supplier A and 2 products for supplier B.
We are going to enter a claim, so we select the supplier. When we type a product, it also suggests the products of supplier B; that is to say, it is showing all the products of the table Product. We only want it to suggest the products from supplier A, which

means that it should search the SupplierProduct table only for the products that have supplier A.

The property that we have just seen (Instantiated Attributes), is precisely for limiting the set of possible values displayed and/or entered, based on information that we have instantiated in the transaction.

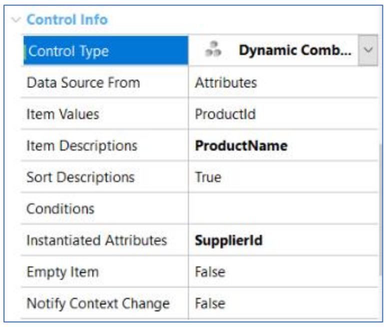
In order to display only the products of the chosen supplier, we must indicate the SupplierId attribute as instantiated in the mentioned property. The attributes indicated here are involved in determining the table to be navigated to suggest and accept the values entered by the user. In this case, when entering SupplierId to retrieve the ProductId, the table navigated will be SupplierProduct.

At runtime, we can see that now it only brings the products of the chosen supplier.



Control Info	
Control Type	Edit
Input Type	**Descriptions**
Suggest	**Incremental**
Item Values	ProductId
Item Descriptions	**ProductName**
Filter Operator	Begins with
Suggest Max Items	5
Sort Descriptions	True
Case Sensitive	False
Conditions	
Instantiated Attributes	**SupplierId**
Auto correction	False
Auto capitalization	None
Notify Context Change	False

VS.



Control Info	
Control Type	**Dynamic Comb...**
Data Source From	Attributes
Item Values	ProductId
Item Descriptions	**ProductName**
Sort Descriptions	True
Conditions	
Instantiated Attributes	**SupplierId**
Empty Item	False
Notify Context Change	False

- It does not allow writing the value; it allows selecting it from a list
- You cannot specify a maximum number of values to display

- It allows you to write the value
- You can specify a maximum number of values to display (Max Items)

For the Dynamic Combo boxes it is also possible to use the Instantiated Attributes property, as well as the Conditions property.

As we have seen, the use of the Dynamic combo box is similar to when we set the Input Type in Descriptions. One difference is that the Edit controls allow typing values, even different from the suggested ones, while the combo controls do not. Another difference is that in the Edit controls, as we saw, you can specify a maximum number of values offered with the Max Items property and in the dynamic combo it is not possible. Therefore, if the number of records is very large it will still load all of them, and that is an advantage of the Edit controls.

This is a summary of the use of the properties shown; to learn more, we recommend you explore our Wiki.

GeneXus™

training.genexus.com
wiki.genexus.com