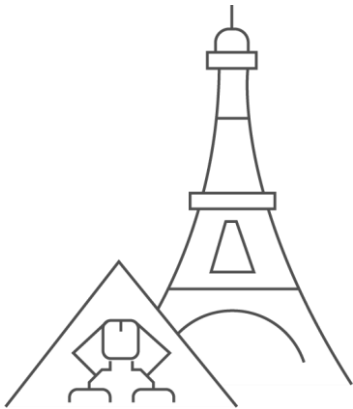


# Attributes and Domains

Working with Attributes and Domains

**GeneXus**<sup>™</sup>

## New Transaction



**ATTRACTIONS**

**NAME:** Louvre Museum

**COUNTRY:** France

**IMAGE:** 

**CATEGORY:** Museum

We will create a transaction to record tourist attractions.

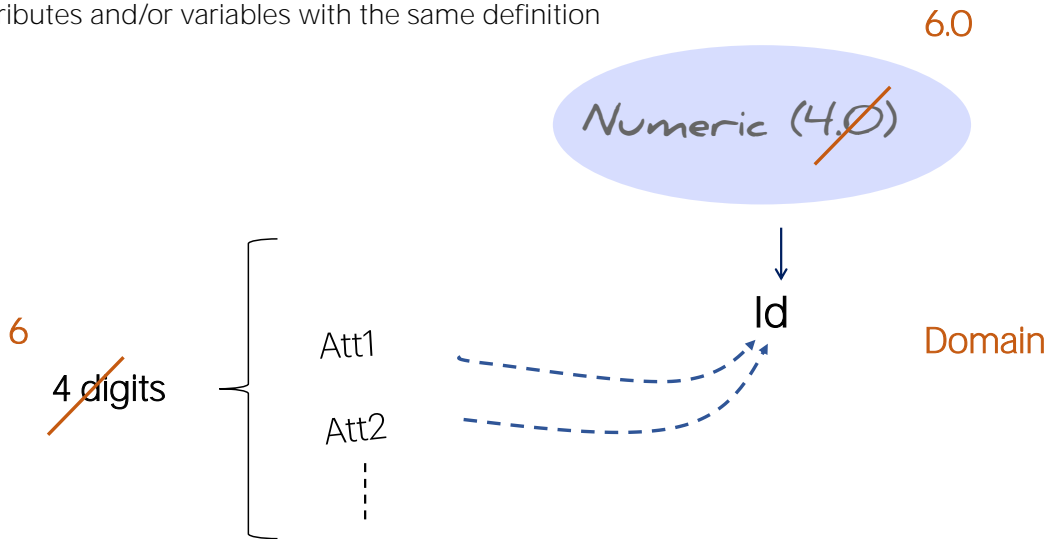
This transaction will be called: **"Attraction"**.

For each tourist attraction, we have been asked to enter:

- its name
- the country where it is located
- a photo
- and a category that describes if it is a monument, a museum, a show, and so on.

## Domains

- Objective: Make generic definitions
- When should we use domains?
  - Attributes and/or variables with the same definition



The key attribute of this transaction will be called “AttractionId”. We should always remember to type a dot so that GeneXus suggests a prefix and we can avoid typing mistakes.

We complete the key attribute name and set its type as numeric of 4 digits, just like we did with the Customer identifier: CustomerId.

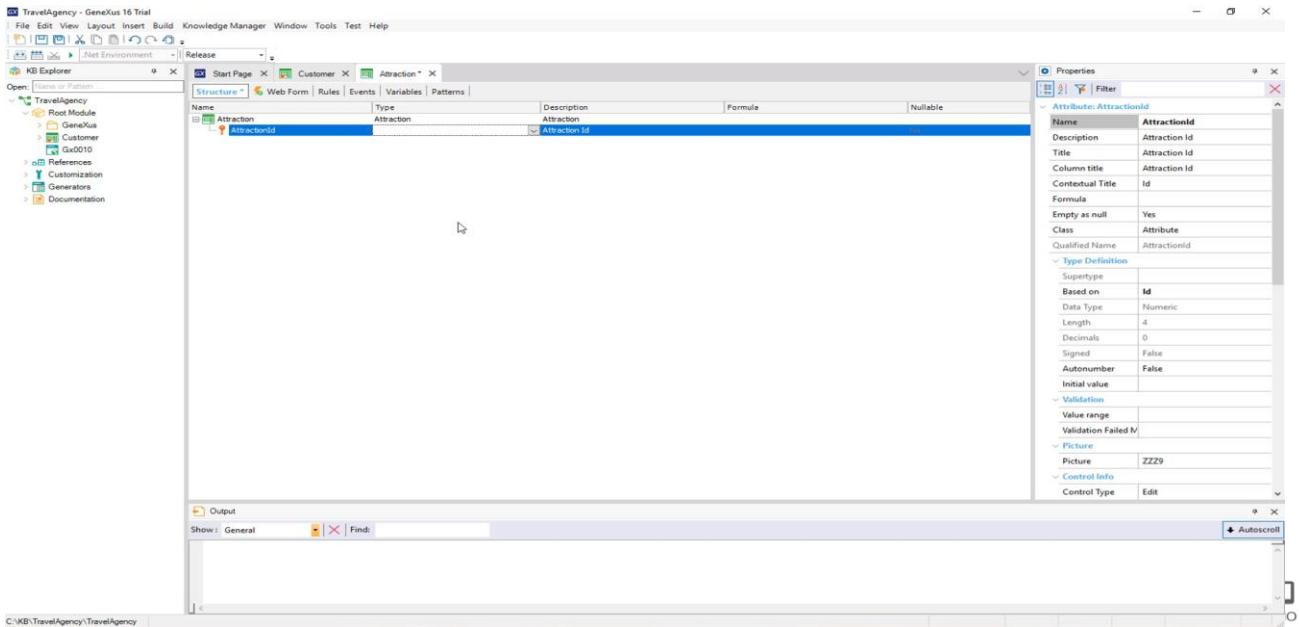
As we will probably need to create more identifiers, it would be a good idea to create a common data type for all identifiers; for instance, a type that we may call “Id” and is a numeric value of 4 digits.

The data type created by us is called Domain.

For example, once **we’ve** created the Id domain, we can set several attributes to Id type... and they will all be 4 digit numeric values.

One advantage it provides is that, if later on we need identifiers to be “Numeric” of length 6 instead of “Numeric” of 4, changing the “Domain” definition will be enough to update all the attributes based on that “Domain” in a single step.

## DEMO



[DEMO: <https://youtu.be/GimYAs2M7ek> ]

To create it, we press the Tab key and in the Type column we write: Id=N (it is autocompleted with Numeric). We leave the 4-digit value suggested by default.

After pressing Enter, we can see that the AttractionId attribute has now been set as Id type.

Let's take a look at the properties of the AttractionId attribute.

If we are positioned on the AttractionId attribute and press F4, this window will display several settings made for this attribute and will allow us to change them.

We can see that it is "Based on" the Id domain and for this reason it is a 4-digit numeric value.

Note that properties can be sorted alphabetically. Here we can see the Autonumber property.

This property is set to False by default and if we change it to True, all the new attractions entered will be automatically numbered in sequence.

That is to say, every time that a new attraction is added, the AttractionId attribute will be automatically assigned a new number that's bigger than the

last existing number.

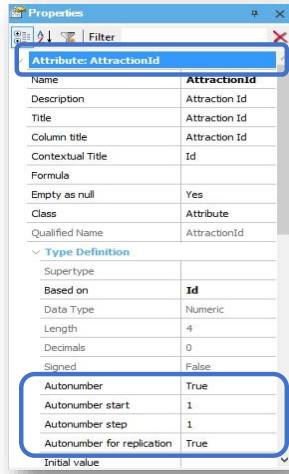
So, we're setting the Autonumber property specifically for this AttractionId identifier attribute.

Another option could be to set the same property for the Id domain that **we've** created...  
... so that when we create more transaction identifier attributes, we can assign them the Id domain. In this way, they would inherit all the domain definitions (such as the data type and all the properties configured).

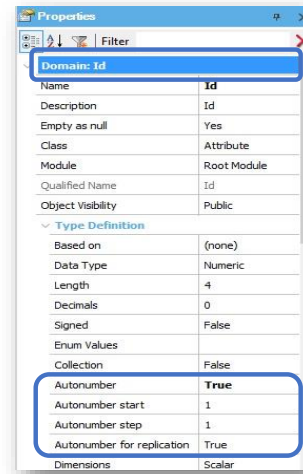
## Automatic numbering

- Autonumber property (values: True, False)
- Where is it defined? Two possible options:

### 1. In the [attribute](#) definition.

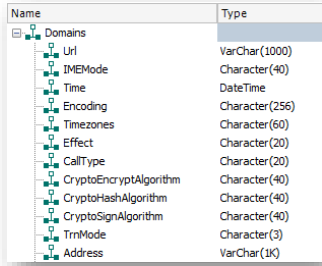


### 2. In the [domain](#) definition.

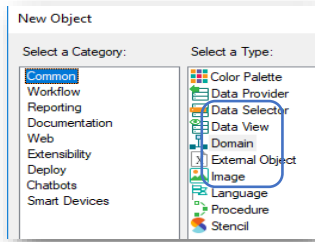


# How do we create a Domain?

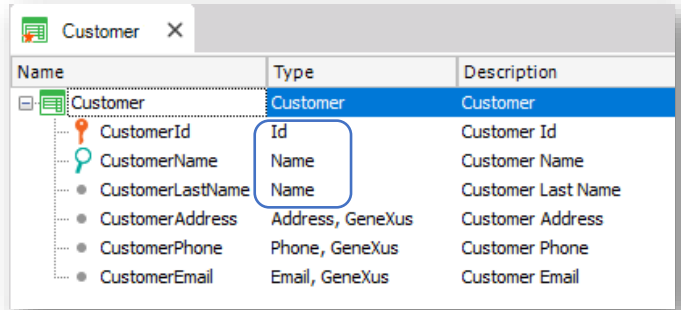
- View / Domains



- File / New / Object



- Inline definition in the transaction structure:



- Properties can be set for them → they are inherited by the attributes and variables based on them.

To see the domains created, we edit the Domains window.

Here we can create and edit domains, in a similar way to how we create attributes.

We click on the Id domain and the properties window is refreshed to show this **domain's** properties.

We find the Autonumber property and set it to True... this will cause all Id type attributes to be automatically autonumbered in sequence.

Semantic domains  
(Default)

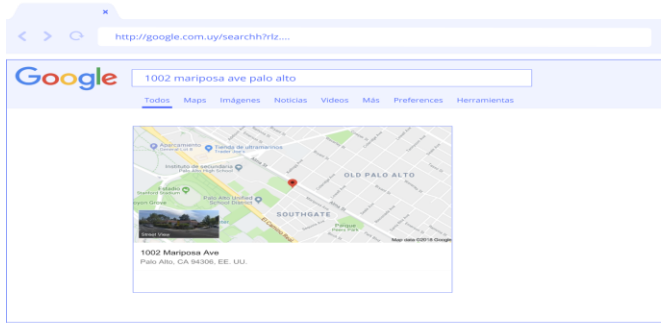
Email  
Address  
Phone  
URL  
Geolocation

Work With Customers

Name

| id | Name       | Address                                   | Phone          | Email              |
|----|------------|---|----------------|--------------------|
| 1  | Mary Brown | 1002 Mariposa Ave Palo Alto CA 94306-1050 | (650) 329-1704 | mbrown@hotmail.com |

| Name             | Type             | Description        |
|------------------|------------------|--------------------|
| Customer         | Customer         | Customer           |
| Customerid       | id               | CustomerId         |
| CustomerName     | Name             | Customer Name      |
| CustomerLastName | Name             | Customer Last Name |
| CustomerAddress  | Address, GeneXus | Customer Address   |
| CustomerPhone    | Phone, GeneXus   | Customer Phone     |
| CustomerEmail    | Email, GeneXus   | Customer Email     |



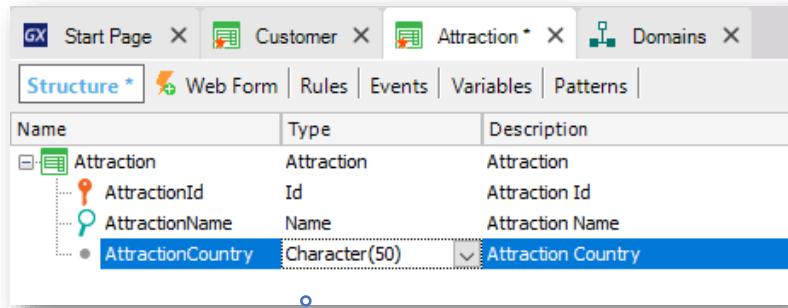
SEND Save Now Discard

To:  Add Cc Add Bcc

Subject:



## Domains



We go back to the structure window of the Attraction transaction and start to create its second attribute.

We add the **AttractionName** attribute. We also create the Name domain of Character type, length 50.... And set the Name type for the AttractionName attribute.

We could create an attribute called AttractionCountry as Character(50) and enter the country name when adding details...

What happens if we want to enter two tourist attractions from the same country?

We should enter the same country name twice... and be careful to type it exactly the same! Later on, we might need to search for all the attractions in a certain country... and to get them, the country must have been typed the same every time.

Transaction:

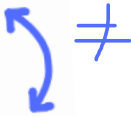
Attraction
⌵

📍 AttractionId

AttractionName

AttractionCountry

AttractionId    AttractionName    AttractionCountry

|   |               |        |  |
|---|---------------|--------|--|
| 1 | Louvre Museum | France |  |
| 2 | Great Wall    | China  |  |
| 3 | Eiffel Tower  | France |  |

Let's take a look at this case...

Suppose that **we've** entered several attractions with their corresponding countries.

For example, we have an attraction with identifier 1, called Louvre, located in France, an attraction with identifier 2, called the Great Wall and located in China...and another attraction with Id=3, the Eiffel Tower, which is also in France.

We know that the Louvre is located in France and that the Eiffel Tower is also in France... but due to a typing or spelling mistake, we type the country name differently.

Here we typed France with two Ns!!... so, for the system, this country is not the same as this other one!

For this reason, this solution can't be used...

It seems more reasonable to enter the country only once, in a single location, and then for each attraction make reference to the corresponding country.

## Attraction

| AttractionId | AttractionName | AttractionCountry | CountryId |
|--------------|----------------|-------------------|-----------|
| 1            | Louvre Museum  | France            | 2         |
| 2            | Great Wall     | China             | 3         |
| 3            | Eiffel Tower   | France            | 2         |

*Note: The 'AttractionCountry' column is crossed out with a blue 'X'. A blue arrow points from the 'France' entry in the first row to the 'France' entry in the third row. A blue arrow points from the 'China' entry in the second row to the 'France' entry in the third row. A blue '≠' symbol is placed between the 'AttractionCountry' and 'CountryId' columns.*

## Country

| CountryId | CountryName |
|-----------|-------------|
| 1         | Brazil      |
| 2         | France      |
| 3         | China       |

That is to say, we should define something like this:

One location where countries are stored... and in attractions we make reference to the corresponding country identifiers.

The Louvre is in France

Country 2...

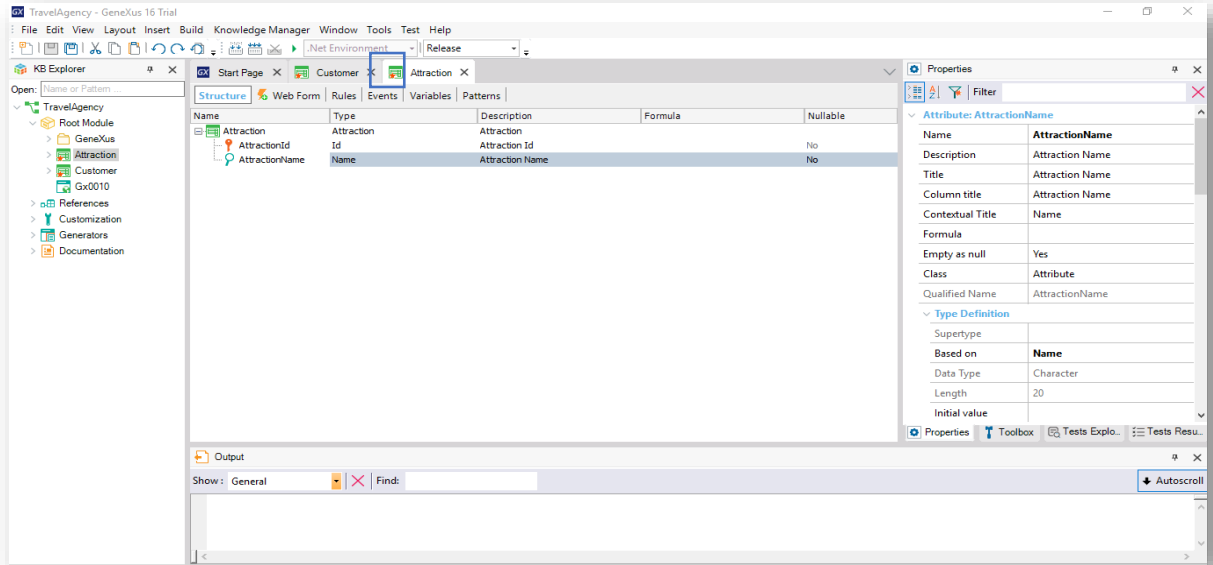
The Great Wall is in China

3...

And the Eiffel Tower is in France... Also 2.

To do this in GeneXus we will create a transaction to record the countries... and then we will see how to assign a country to each transaction.

## GeneXus Server



Meanwhile, we save the Attraction structure that we haven't completed. Note that it is graphically indicated that this object is pending to be uploaded to GeneXus server.

We leave it like this for now because we will need to upload this change together with other changes that will be made in the following video.

## DEMO

- Create **Country** Transaction
  - Attributes: CountryId, CountryName
- Create **Attraction** Transaction
  - Attributes: AttractionId, AttractionName, CountryId, CountryName
  - Define Id and Name domains
  - Autonumber in Id domain
- Execute the application (F5):
  - Read the “Impact Analysis”
    - Note that CountryName is not in the Attraction table.
  - Explain “**Reorganize**” concept
  - Data entry: countries and tourist attractions.
    - Note automatic numbering
    - Show automatic controls for data consistency (CountryId)

*GeneXus*<sup>TM</sup>

[training.genexus.com](http://training.genexus.com)  
[wiki.genexus.com](http://wiki.genexus.com)