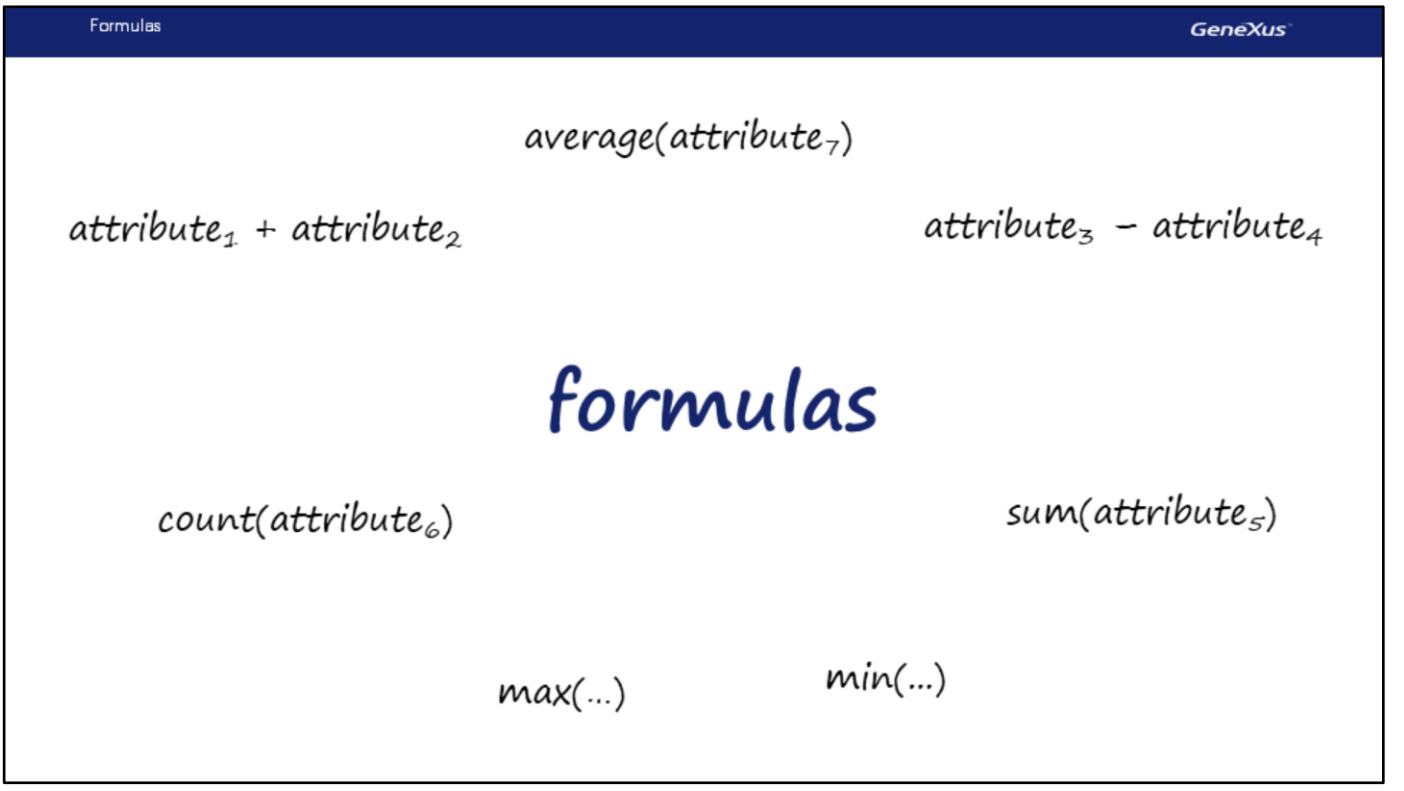
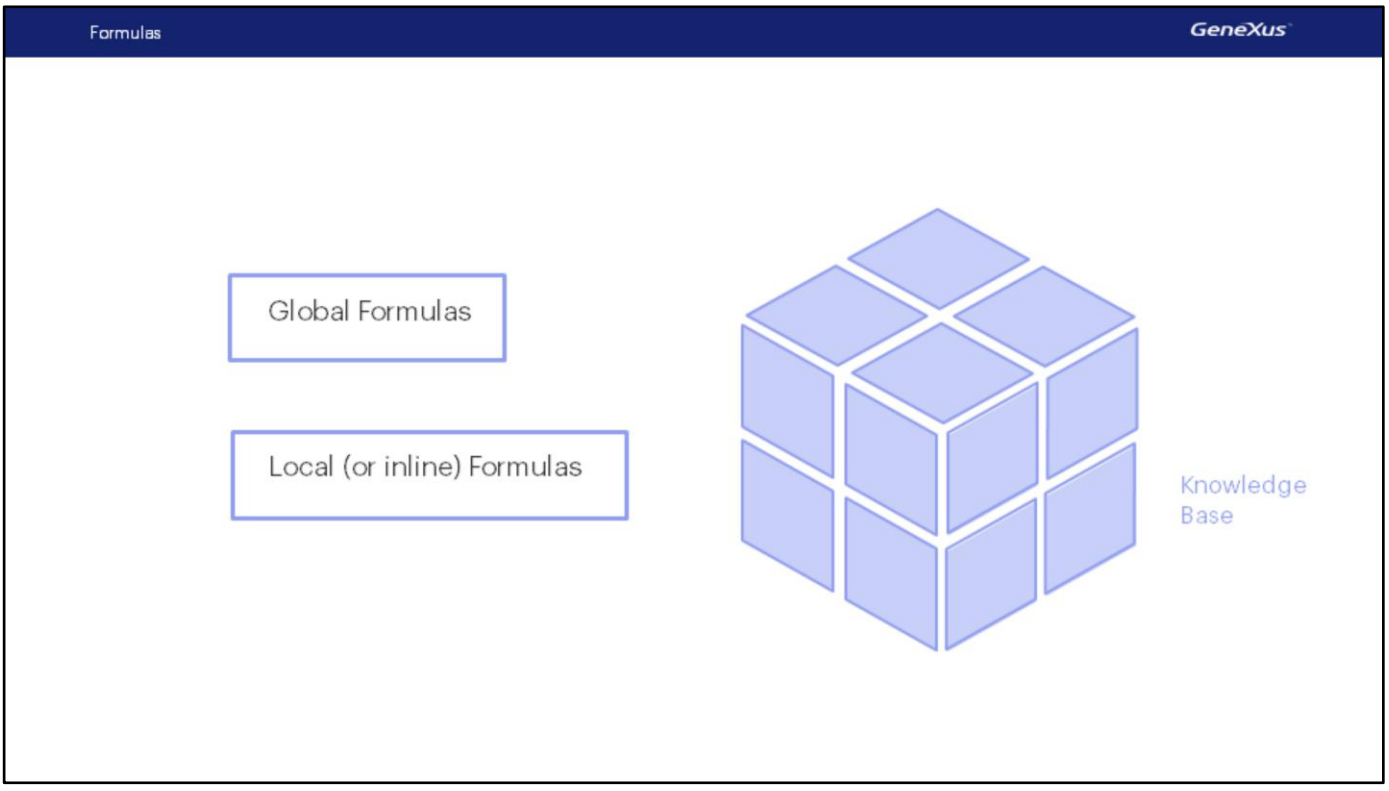
Formulas

GeneXus 16



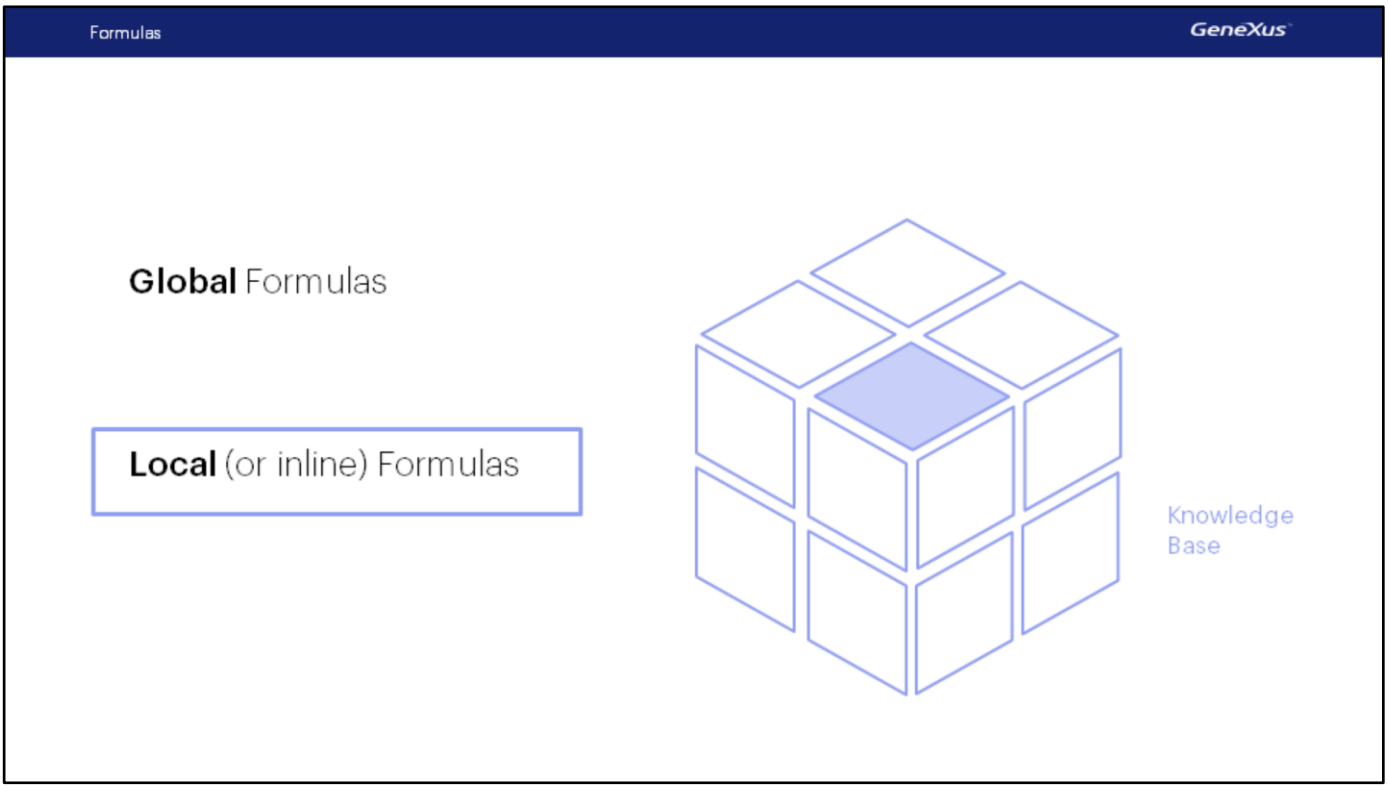
Many times we need the application to solve a calculation involving values from certain attributes, constants, and/or functions.

For these cases, GeneXus provides **Formulas.**



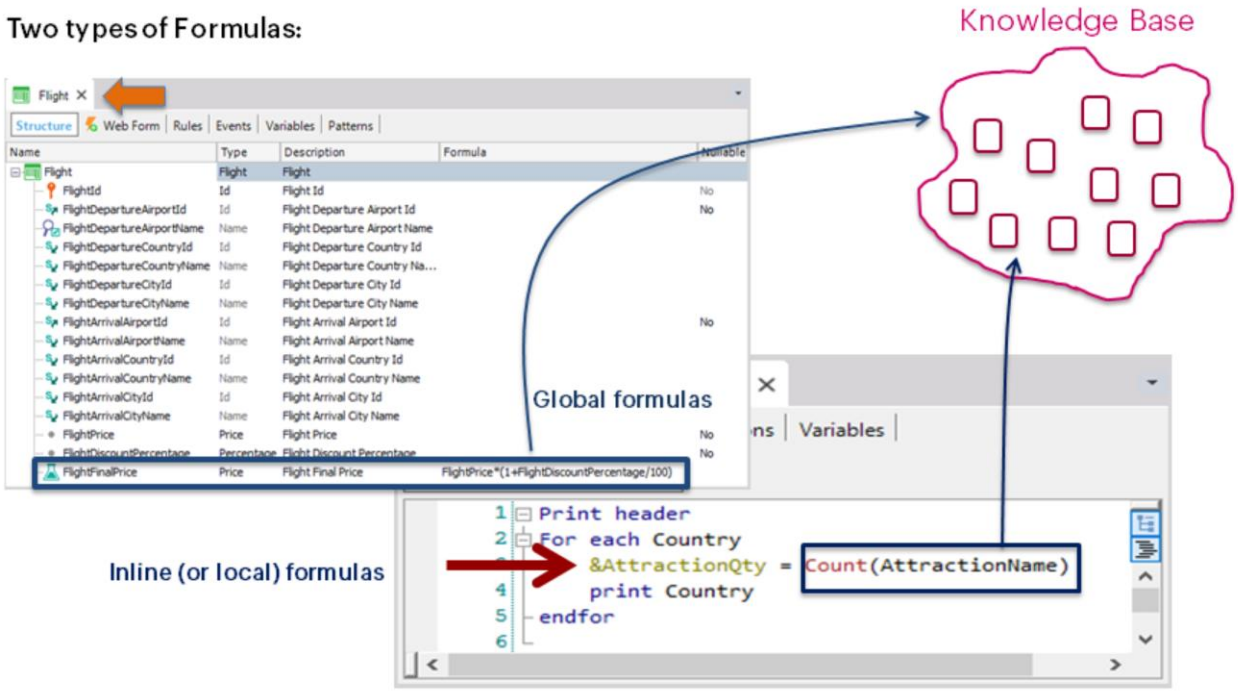
Formulas can be defined in two ways:

GLOBALLY: the calculation will be available throughout the Knowledge Base.



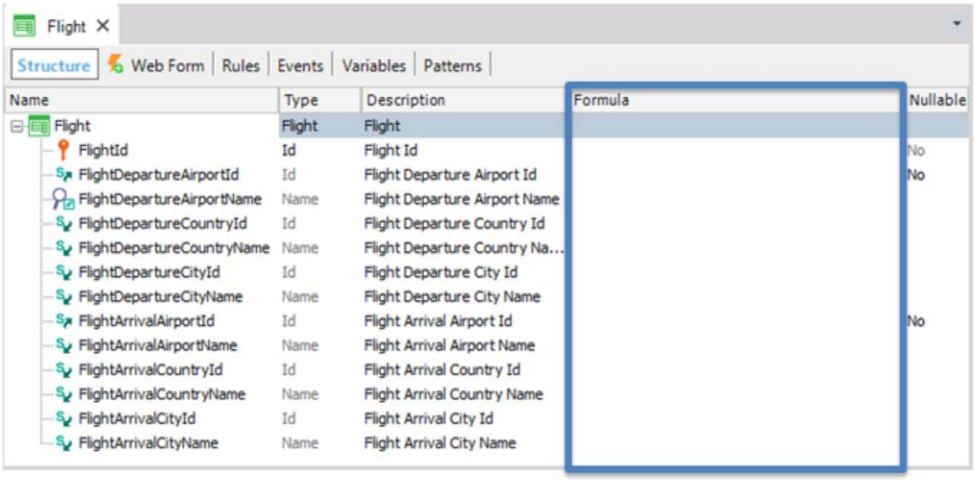
LOCALLY or INLINE: in this case, the calculation will be available only in the object in which it has been defined.

Two types of Formulas:



Global formulas

It is a calculation we define in association with an attribute; from then on it will be "virtual".

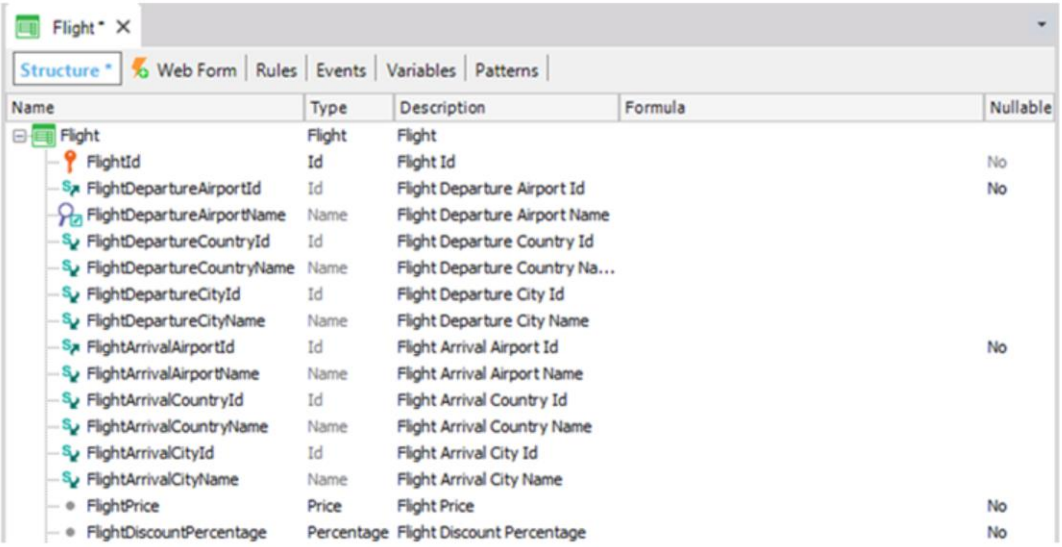


Let's start by explaining **what a global formula is** and how it is defined.

A global formula is a calculation defined in relation to an attribute.

Note that transaction structures contain a column titled "Formula".
If a calculation is defined in this column for an attribute, **GeneXus will understand that this attribute is virtual**; that is to say, it will not have to be physically created as a field in the associated table, because the attribute value will be obtained by making the calculation we've indicated.

Adding a global formula



Name	Type	Description	Formula	Nullable
Flight	Flight	Flight		
FlightId	Id	Flight Id		No
FlightDepartureAirportId	Id	Flight Departure Airport Id		No
FlightDepartureAirportName	Name	Flight Departure Airport Name		
FlightDepartureCountryId	Id	Flight Departure Country Id		
FlightDepartureCountryName	Name	Flight Departure Country Na...		
FlightDepartureCityId	Id	Flight Departure City Id		
FlightDepartureCityName	Name	Flight Departure City Name		
FlightArrivalAirportId	Id	Flight Arrival Airport Id		No
FlightArrivalAirportName	Name	Flight Arrival Airport Name		
FlightArrivalCountryId	Id	Flight Arrival Country Id		
FlightArrivalCountryName	Name	Flight Arrival Country Name		
FlightArrivalCityId	Id	Flight Arrival City Id		
FlightArrivalCityName	Name	Flight Arrival City Name		
FlightPrice	Price	Flight Price		No
FlightDiscountPercentage	Percentage	Flight Discount Percentage		No

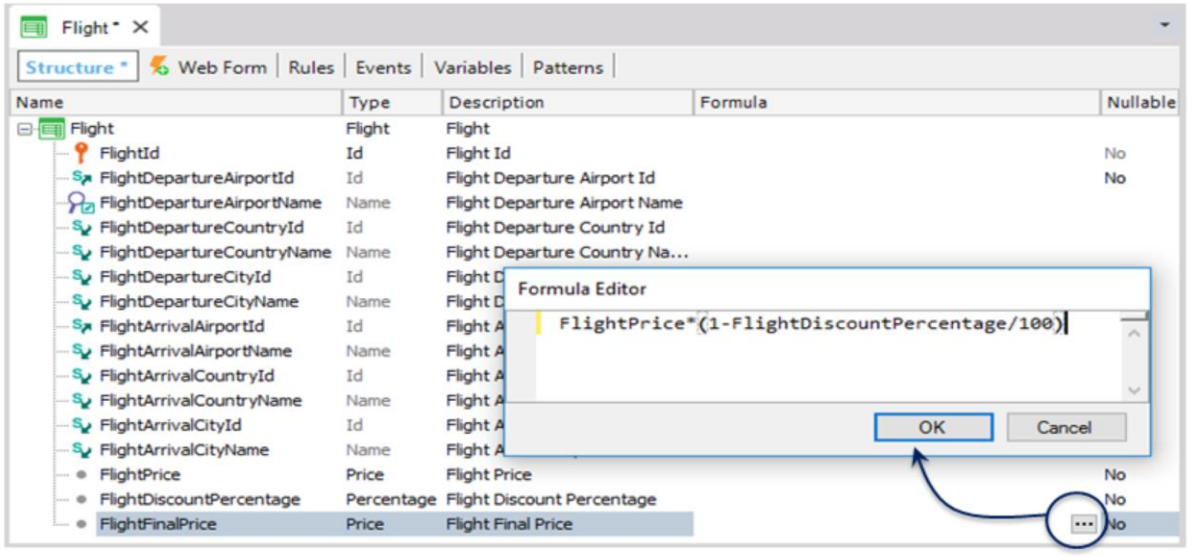
Let's see this with an example.

First, we will define a new attribute in the Flight transaction, **in order to store the price of each flight.**
So, we add the FlightPrice attribute.
And create the Price domain.

We save.

Now we will add another new attribute in the same transaction **to store the discount applied to each flight.** We call it: FlightDiscountPercentage... its data type will be a domain also called Percentage, numeric of 3 digits.

Adding a global formula

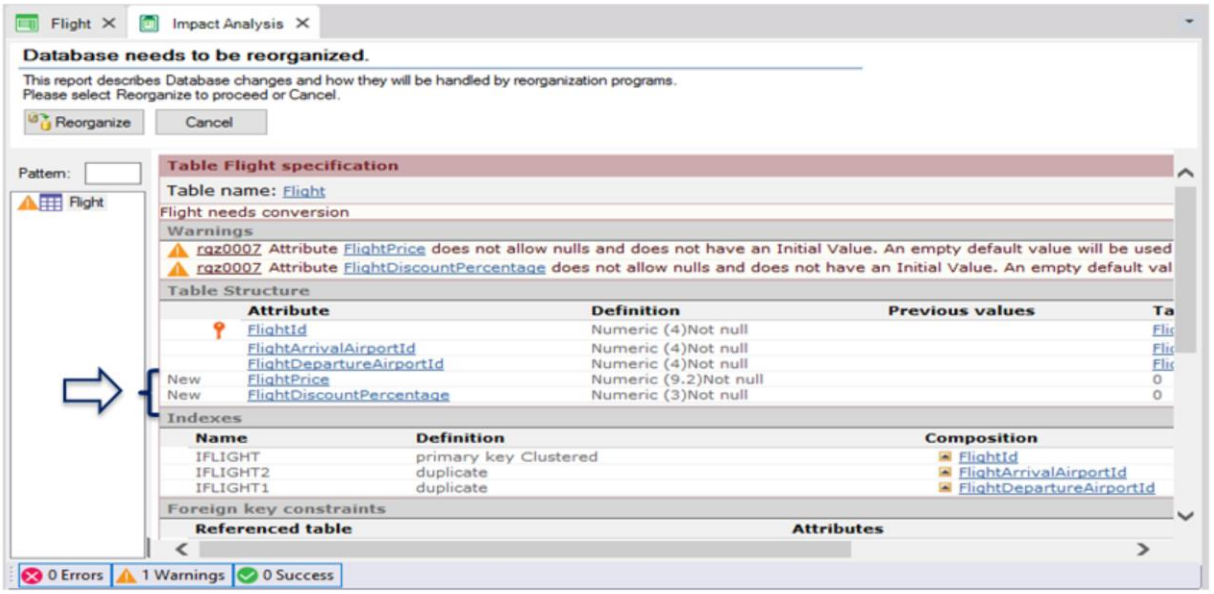


Lastly, we will add another attribute called FlightFinalPrice, based on the Price domain; **this time, the attribute will be defined as a global formula.**

To this end, **in this attribute's "Formula" column**, we will define the calculation necessary, **so that it is always run and this attribute provides "the flight's current price"; that is to say, the price after subtracting the discount percentage stored in** FlightDiscountPercentage **from FlightPrice.**
So, in this formula column we will type the corresponding calculation.

Note that in this window we only have to type the calculation, not the assignment.

Impact Analysis



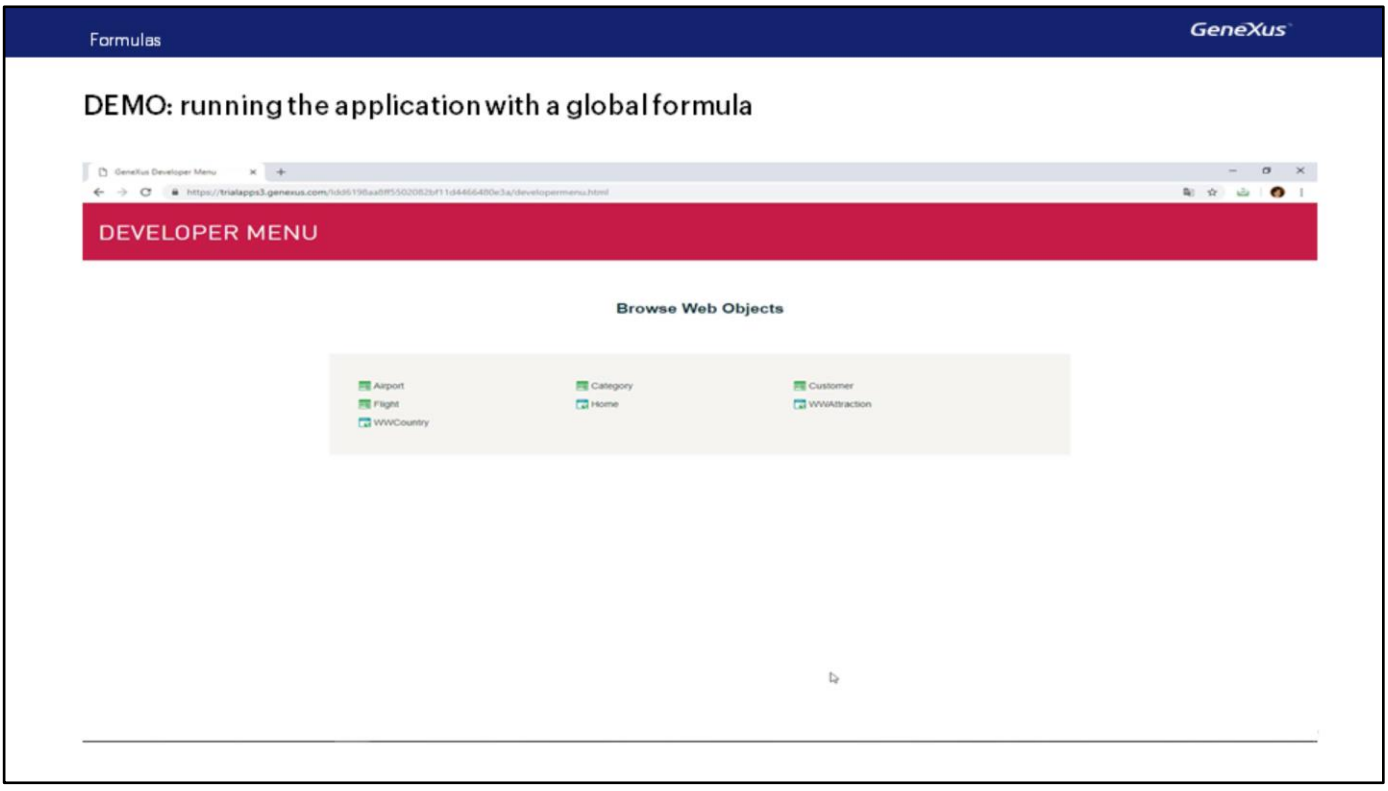
Now we press F5, which automatically saves all pending actions... and see what happens.

In the Flight physical table, only 2 fields are created, even though we have defined **three new attributes in the transaction structure.**

Due to the fact that the formula column contains a definition, **this attribute is not added in the physical table.**

Because the attribute is defined in the Knowledge Base with an associated formula, GeneXus can calculate its value. Also, in every object where this attribute is included, the calculation will be made and the result will be shown.

We reorganize... and see the application at runtime.



[DEMO: https://youtu.be/oi0rfa210xw]

We run the Flight transaction, query flight number 1, and in this form we see the three new attributes created:

- the flight price enabled for us to enter it,
- the discount percentage, also enabled for us to enter it,
- and the final price, disabled because it is the attribute defined as a formula, and its value is not entered; instead, it will be calculated and displayed.

Every attribute defined as a global formula will be read-only, and it will not be possible to enter a value for it. This happens because the attribute obtains its value from the associated calculation, which is run every time the attribute is used.

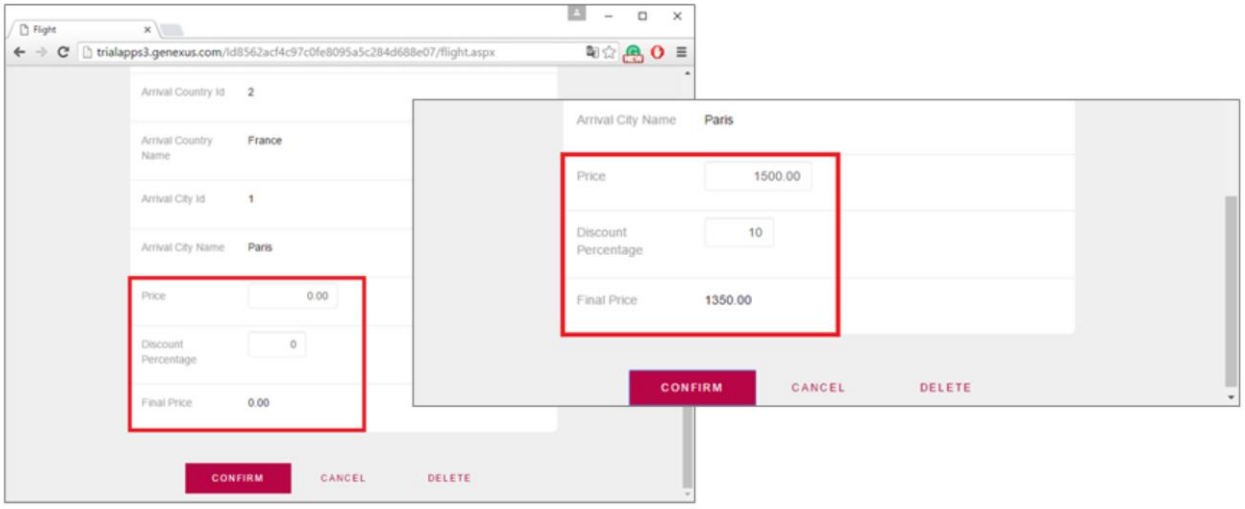
For this reason, there isn't a field in the physical table to store this attribute value. For this reason, there's no need for it to be editable.

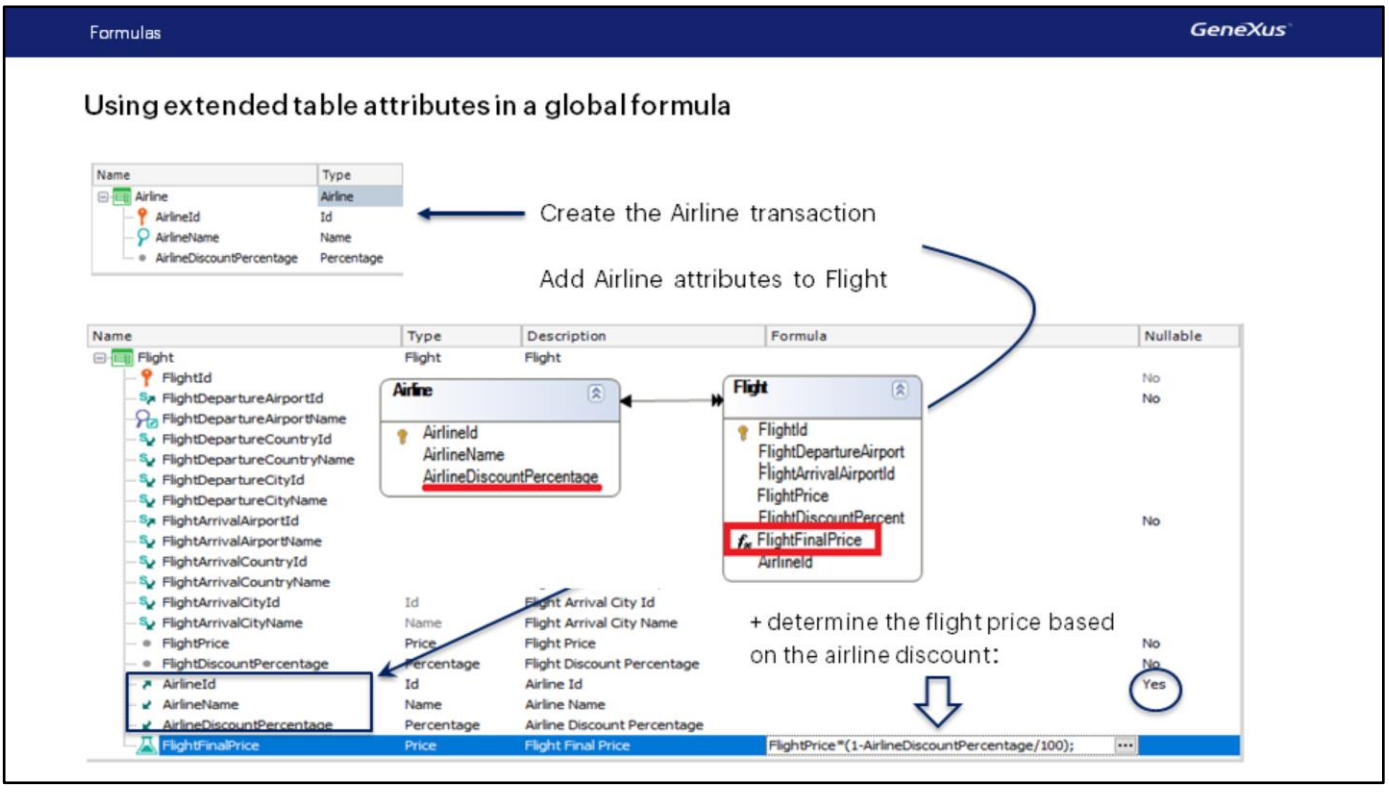
We will enter a price for this flight, and a discount percentage: 10%.

After leaving the field, we see that the formula is immediately run, and the final price of the flight is displayed with the discount applied.

Running the application with a global formula

FlightId 1





Let's return to GeneXus.

In this way, we have defined a **global formula** attribute.

Only attributes can be defined as global formulas in the way we've seen, using the Formula column in the transaction.

Something important to remember is that, even though in the example we have only used attributes from the transaction's own associated table -that is to say, its base table-, **attributes from the extended table can also be used.**

Let's see it.
We will create a new transaction called Airline to record the airlines.

We type:
- AirlineId
- AirlineName and...
- AirlineDiscountPercentage, to record the discount made by the airline for all its flights.

We save. Now we open the Flight transaction to assign an airline to every flight.

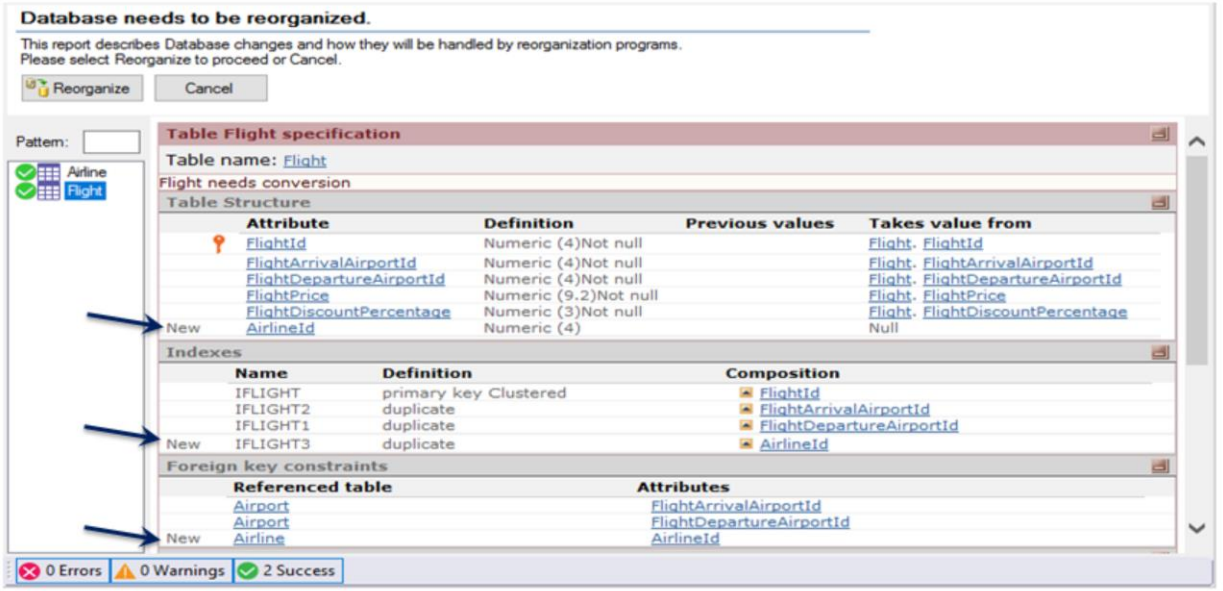
So, we add the AirlineId attribute, which here will be a foreign key... and change the value of its Nullable property to Yes... In this way, we can avoid indicating the flight's airline at this stage, because we still don't have any airlines recorded.

Later on we can change again the value of this Nullable property to No, so that it is mandatory to indicate the airline when entering or changing the details of a flight.
In addition, we add the AirlineName and AirlineDiscountPercentage attributes to also view this data in the form.

Now we will change the definition of this formula to have it calculate the final price of the flight by

applying it the generic discount of the airline, instead of applying the discount of the flight itself.

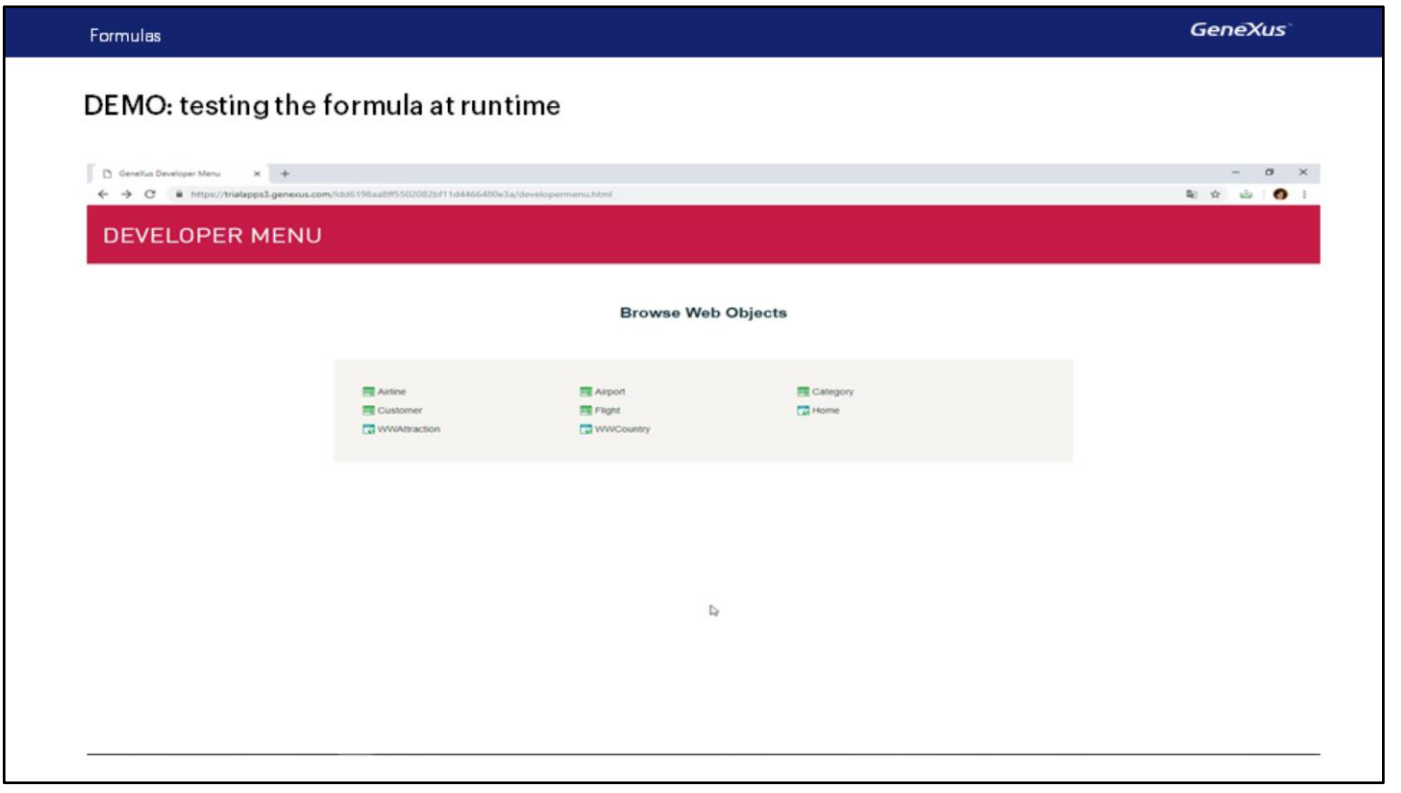
Impact Analysis



We press F5...

As we can see, the Airline physical table will be created with the three attributes defined, and in the Flight table the AirlineId foreign key will be created.

So, we reorganize and run...



[DEMO: https://youtu.be/VK8BR3faA_g]

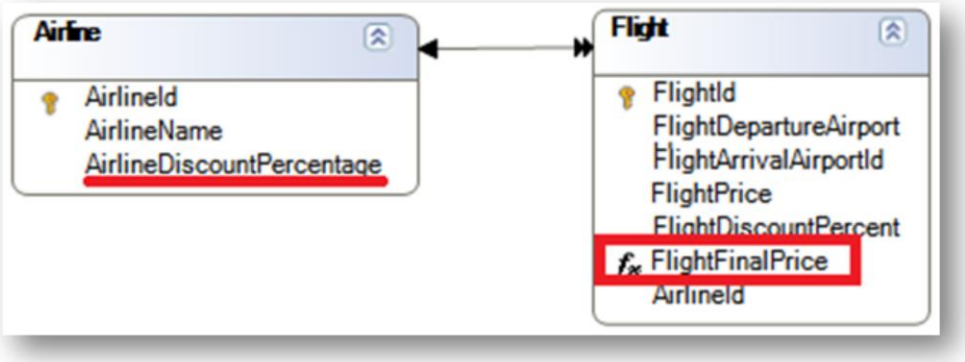
We run the Airline transaction and enter an airline, called TAM, with a 30% discount.

Now we will assign this airline to a flight.

So, we open the Flight transaction, flight number 1, and associate it with airline number 1...

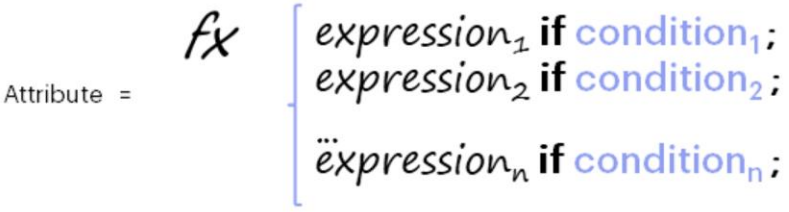
The new final price of the flight, which is a global formula attribute, is calculated.

Testing the formula at runtime



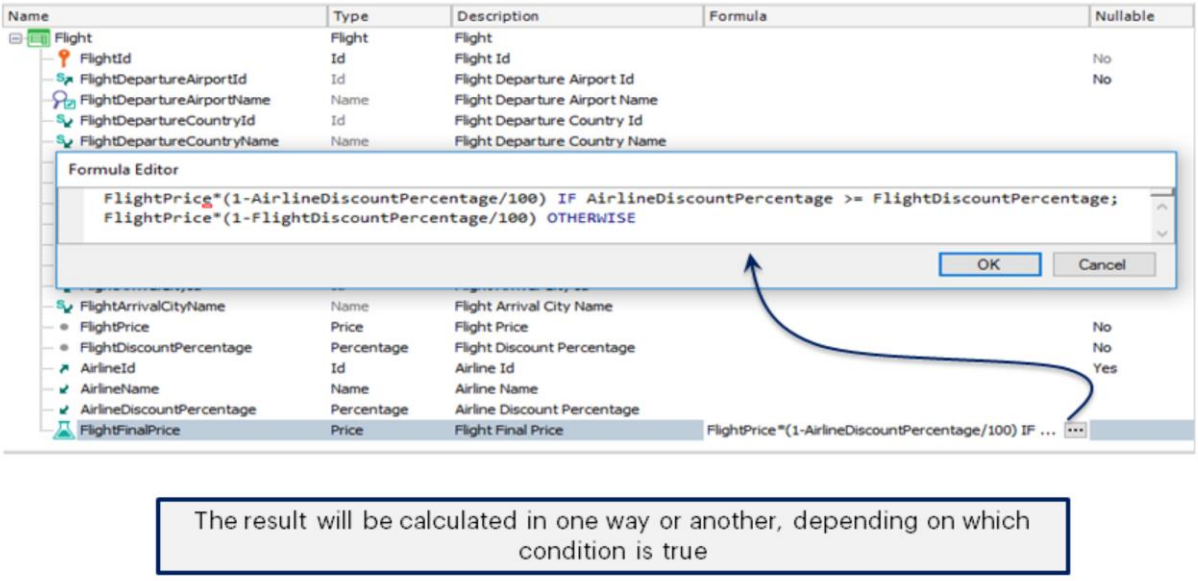
Now it involves the discount percentage of the **airline**, which is an attribute of the extended table of the Flight base table.

Adding conditions to the formula



We haven't mentioned yet that formulas can evaluate conditions, and the result can be calculated in different ways depending on whether these conditions are true or false.

Adding conditions to the formula



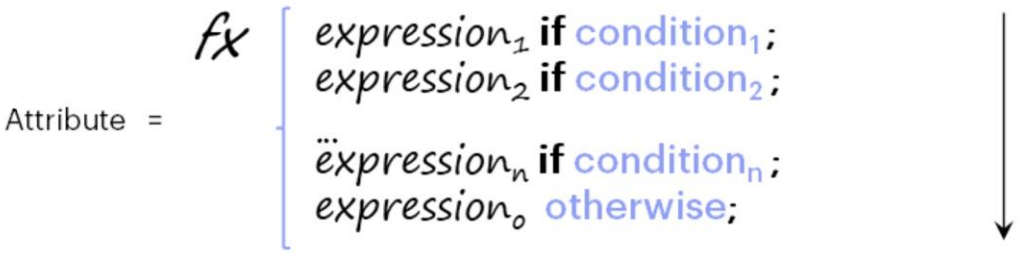
The result will be calculated in one way or another, depending on which condition is true

Let's see this.
To do so, we click on this button to edit the formula.

And we define that the highest discount percentage must be taken into account to calculate the final price of the flight, in order to make the best discount possible.

With this definition, if the airline has a higher discount for all its flights than the discount percentage of the flight itself, the airline discount will be considered to make the calculation.

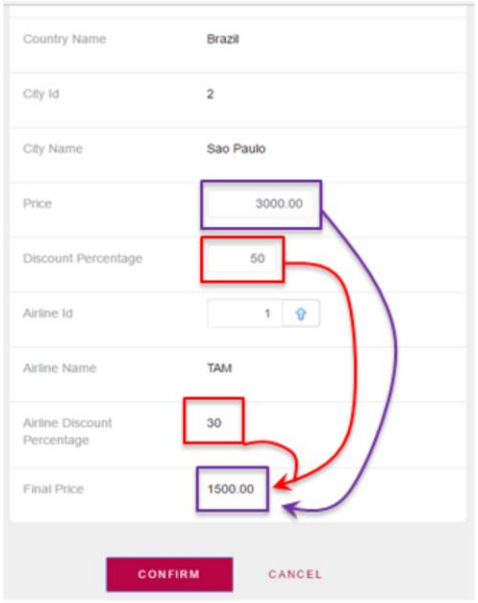
We also add a default condition



Otherwise:

the discount percentage of the flight itself is used to make the calculation.

Testing the new definition of the formula



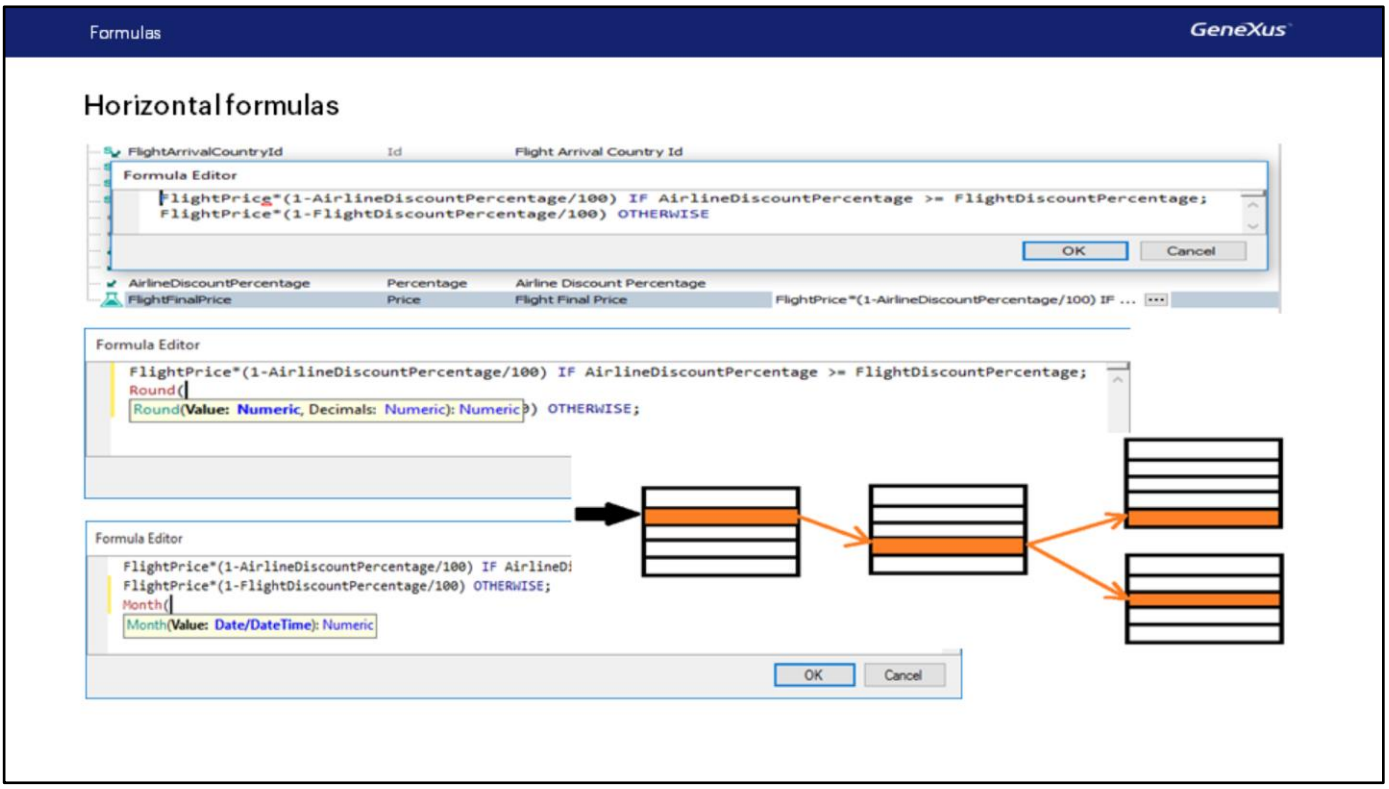
Note that formulas are written as expressions, so they end with a semicolon. To calculate the formula, GeneXus keeps the first expression that meets the condition. If no condition is met, and an otherwise clause has been added, it uses this one.

We move the Final Price attribute to the end of the list, so that the information is more clearly organized.

We press F5, and run the Flight transaction. In the first flight we set its discount percentage to be higher than the overall discount percentage of the airline; for example, 50%.

We exit this field and go to the airline field, so that it has everything necessary to calculate the formula.

The final price of the flight was calculated using the highest discount.



Let's return to GeneXus.

As we've seen, formulas can contain several lines followed by IF, and may contain a last line with OTHERWISE in case none of the above conditions are met.

In turn, even though in this example each result is obtained through a calculation, functions applied to attributes or expressions can also be used, such as Round, to obtain a rounded result.

Also, Month can be used to obtain the month of a date, etc... even a user-defined procedure can be called to return a value.

We will study procedures later on.

Formulas of this type, which make a calculation obtained from the data of the record in which we're positioned (just one), and occasionally of the associated records (through the extended table), are usually called Horizontal formulas.

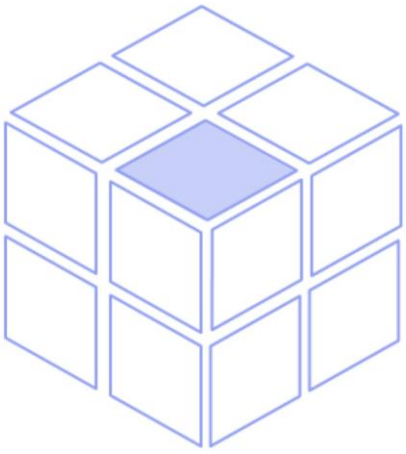
Aggregate formulas

Global Formulas

Attribute = fx

Local (or inline) Formulas

&Variable = fx



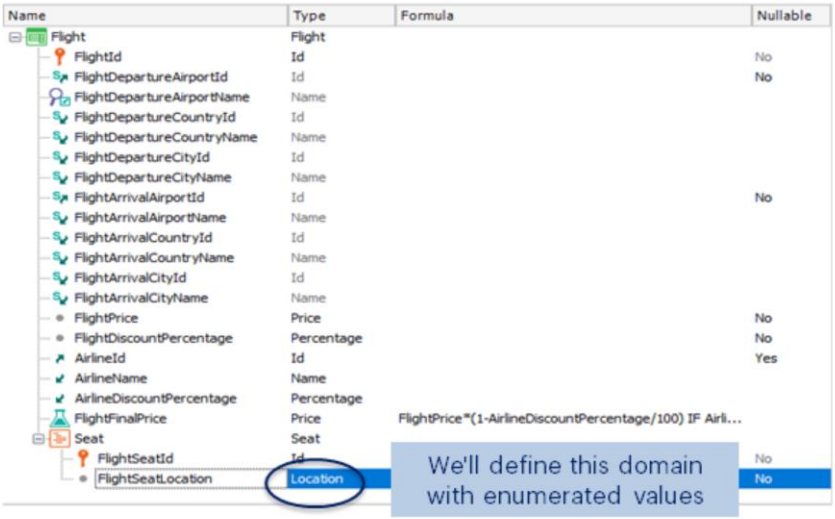
Now we will see another type of formulas called **Aggregate formulas.**

We will explain them by defining examples of global formulas -that is to say, the corresponding calculations will be defined in relation to attributes, and therefore they will not be created as physical fields. As a result, our examples can also be assigned in another context, such as a variable, for example.

They may also be assigned locally in a certain section of an object (we will see this in another video).

Aggregate formulas: Count, Sum, Average, etc.

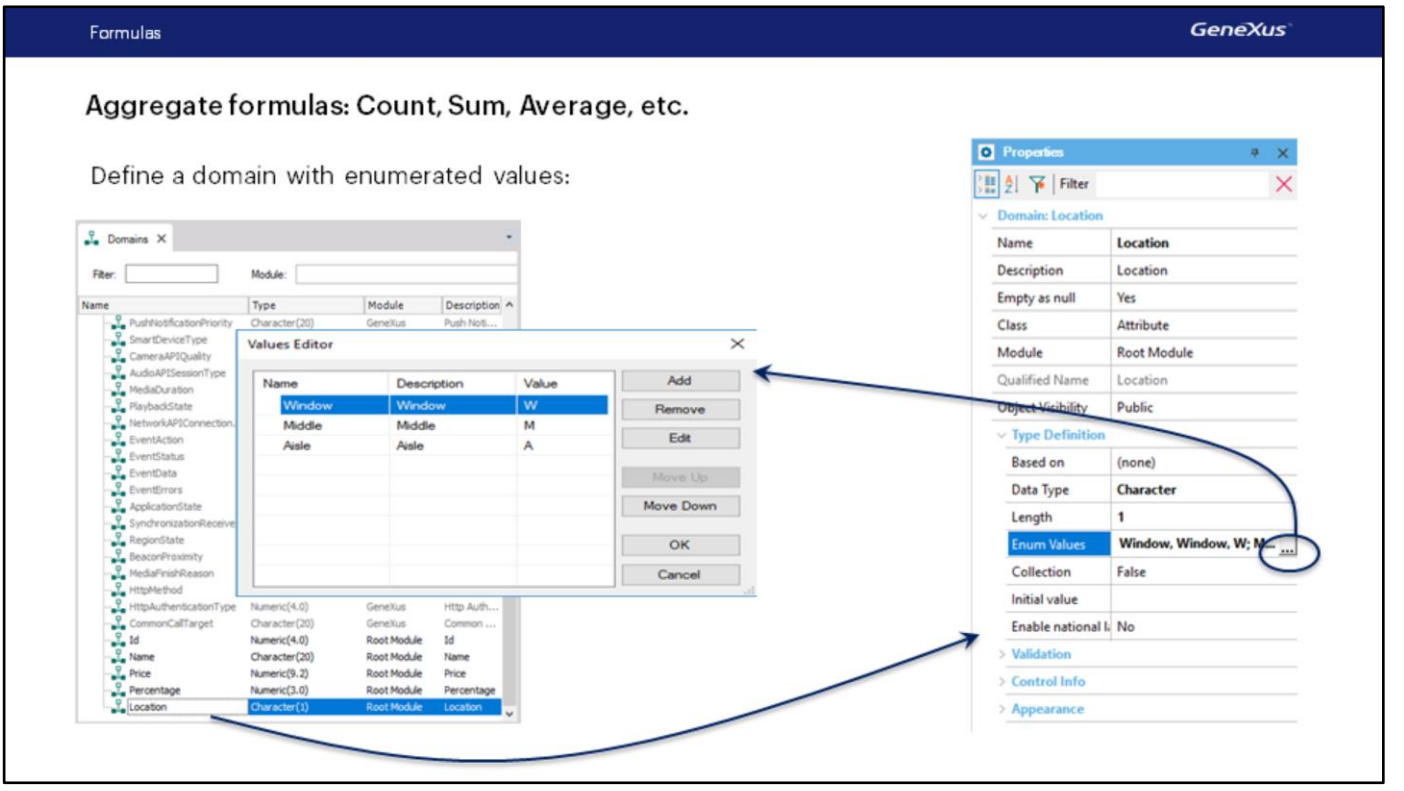
Example: Create a second level in the Flight transaction



Now we will create a second level in the Flight transaction... and call it: Seat

As described by this level name, we will use it to record the seats available in the flight; for each seat, we will indicate if it is next to a window, an aisle, or in the middle. Next, we will need to know the total number of seats available in the flight.

We type a period and complete the attribute name: FlightSeatId
Now we create another attribute called FlightSeat Location... it will be a character of 1.



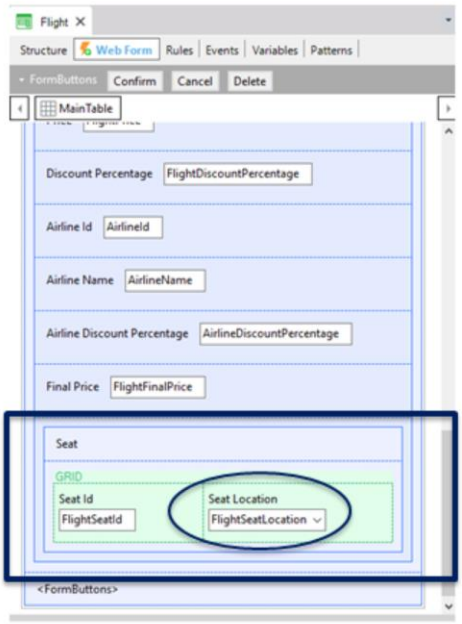
Now we edit the domains, to change a property of the Location domain that we've just created:

We locate the **Enum Values** property, and define the three values that this domain can take:

- Window... the character stored in this case will be “W”
- Middle... the character stored in this case will be “M”
- or Aisle... the character stored in this case will be “A”.

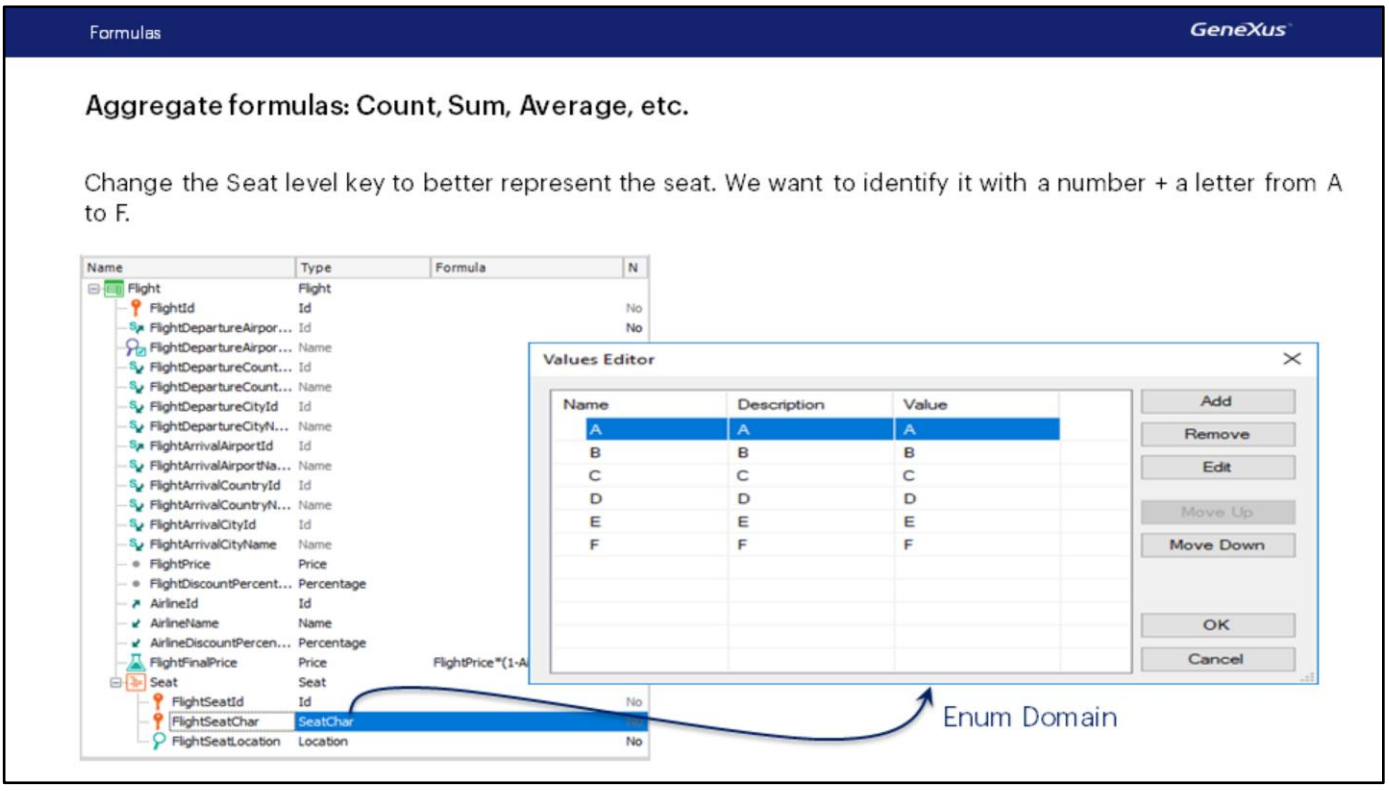
We click on OK.

Aggregate formulas: Count, Sum, Average, etc.



Look at the form of the Flight transaction. A grid has been added to enter the flight's seats, and for every seat we can indicate its position through a combo control.

This combo offers the values “window” “middle” or “aisle”, because they are the possible values defined for the domain of the FlightSeatLocation attribute.



Before pressing F5, let's look at the definition of the second level.

If the key is made up of FlightId plus FlightSeatId, **for each flight, the seat numbers cannot be repeated.** However, we need the seat number to be repeated, because it is identified by this number and a letter. In this way, we will have seats 1A, 1B, 1C, 2A, 2B, and so on.

So, we add a new FlightSeatChar attribute, and its domain will be SeatChar, a character of 1.

We will make this attribute part of the key to record the same seat numbers with different letters.

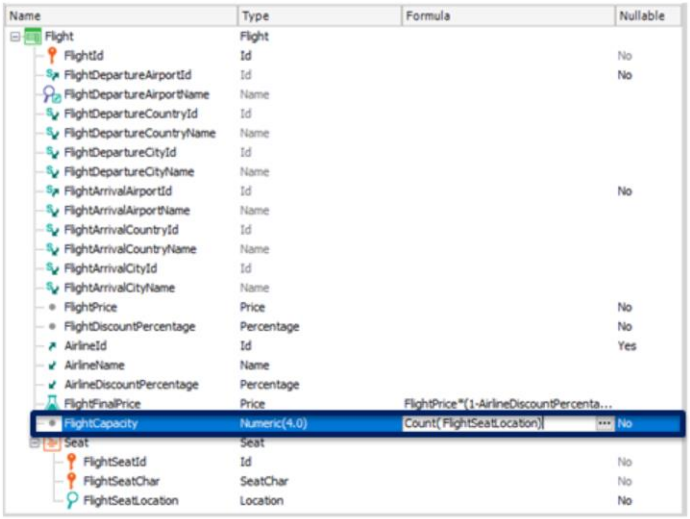
We will limit the possible letters from A to F, and to do so we will edit the SeatChar domain that we've just created...

We locate its Enum Values property and define the possible values:

In this case, the descriptions' values match the values stored. We click on OK.

Aggregate formulas: Count, Sum, Average, etc.

Add a new formula to count the number of seats:



Now, to find out the maximum number of passengers allowed on a flight, according to the number of seats, we will define a new attribute in the first level. In its Formula column we will indicate the calculation consisting in counting the number of seats offered in the flight...

So, we create the FlightCapacity attribute, and its data type will be numeric of 4.

In its Formula column we type: Count... and between brackets we add an attribute that lets GeneXus know that we want to count the seats. To this end, we choose the FlightSeatLocation attribute that belongs to the table containing flight seats, the FlightSeat table.

The attribute indicated in the formula's brackets tells GeneXus the table to navigate in order to make the calculation.

Aggregate formulas: Count, Sum, Average, etc.

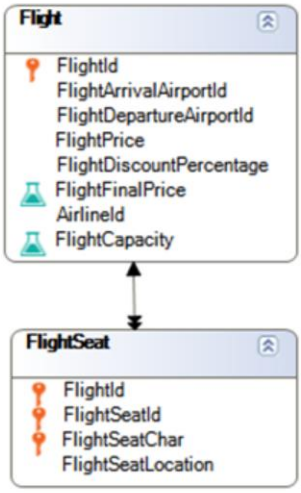
FlightCapacity = Count(FlightSeatLocation)

FLIGHT

FlightId	FlightDepartureAirportId	...	FlightPrice	AirlineId
1	1		1500	1
2	3	...	2500	2
3	1	...	1000	1
...	...	...	...	...

FLIGHTSEAT

FlightId	FlightSeatId	FlightSeatChar	FlightSeatLocation
1	1	A	Window
1	1	B	Aisle
1	2	A	Window
1	2	B	Aisle
1	3	C	Middle
2	1	A	Window
2	1	B	Middle
3	...	...	...
...	...	...	...



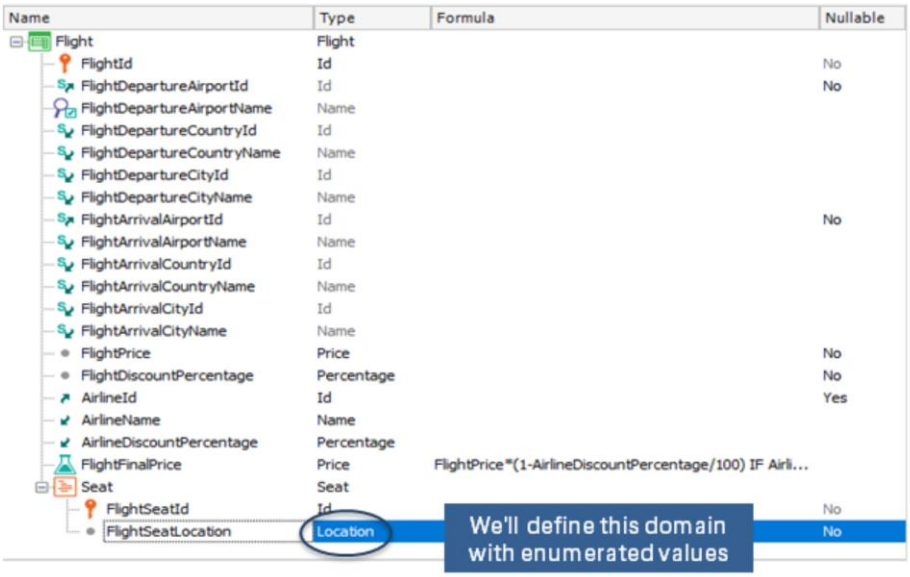
Given a flight, we want to count its seats (many); that is to say, the records of the FlightSeat table related to the flight.

When calculating the formula, we will be positioned on a record of the FLIGHT table that has a 1-to-Many relationship with the seats table, FLIGHTSEAT, which contains the seats of ALL flights. This relationship is determined by the FlightId attribute.
Note that when writing Count(FlightSeatLocation), we're not counting the windows or aisles, but the **records** where these seats are stored. Since we're positioned on a certain flight, GeneXus will only count the seats corresponding to **that** flight.

That is to say, if GeneXus finds a relationship between the table associated with the formula attribute and the table that will be run through by the formula to make its calculation, **it will only take the related records into account to make this calculation...**

If no relationship is found, GeneXus will make the calculation over **all the records of the table navigated.**

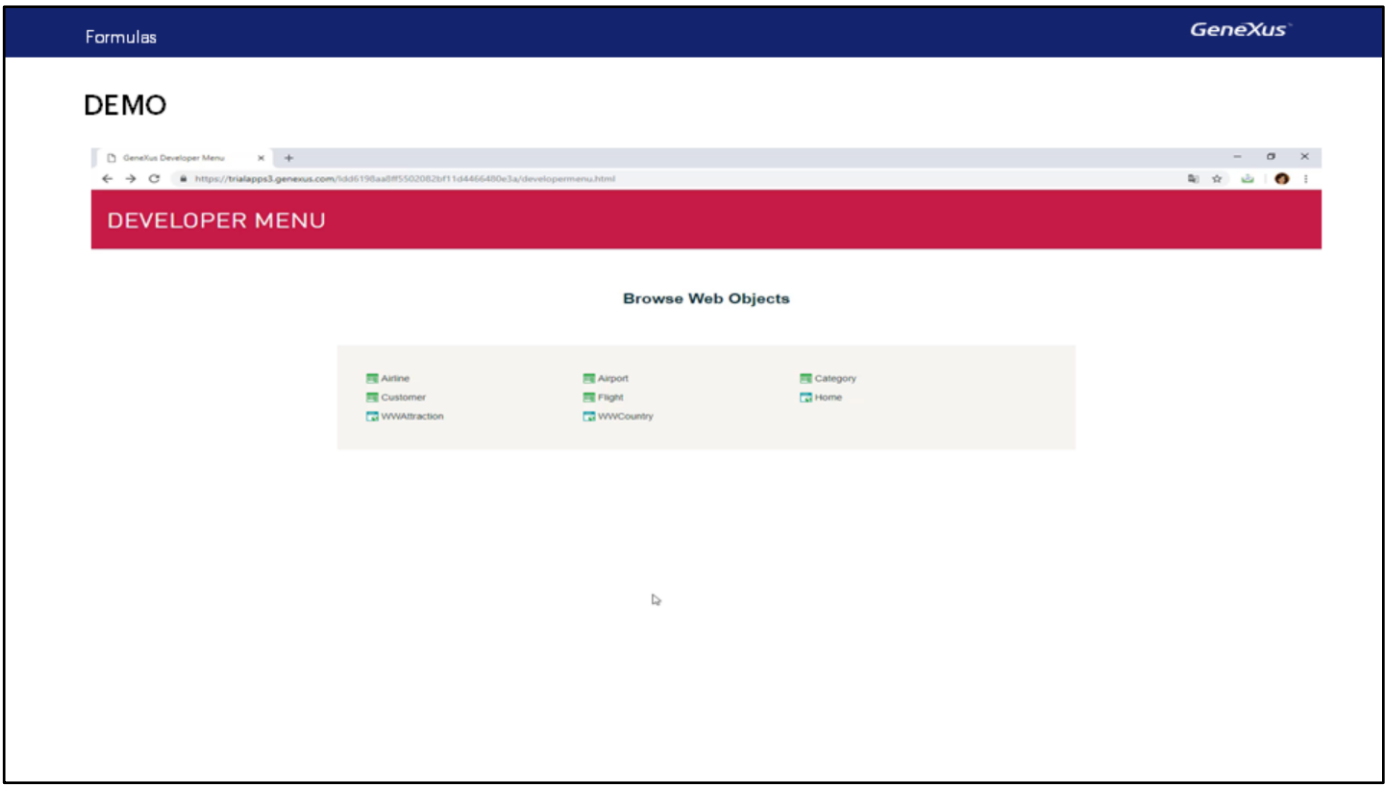
Aggregate formulas: Count, Sum, Average, etc.



Let's try this at runtime by pressing F5...

As we can see, the FLIGHTSEAT physical table will be created, and it will be associated with the 2nd level of the Flight transaction; also, it will contain the attributes and they key that we've defined... **note that the structure of the FLIGHT transaction will not be modified** because the FlightCapacity attribute will not be physically created, as we expected.

We agree, so we click on reorganize...



[DEMO: https://youtu.be/Iv8t1CPDmkA]

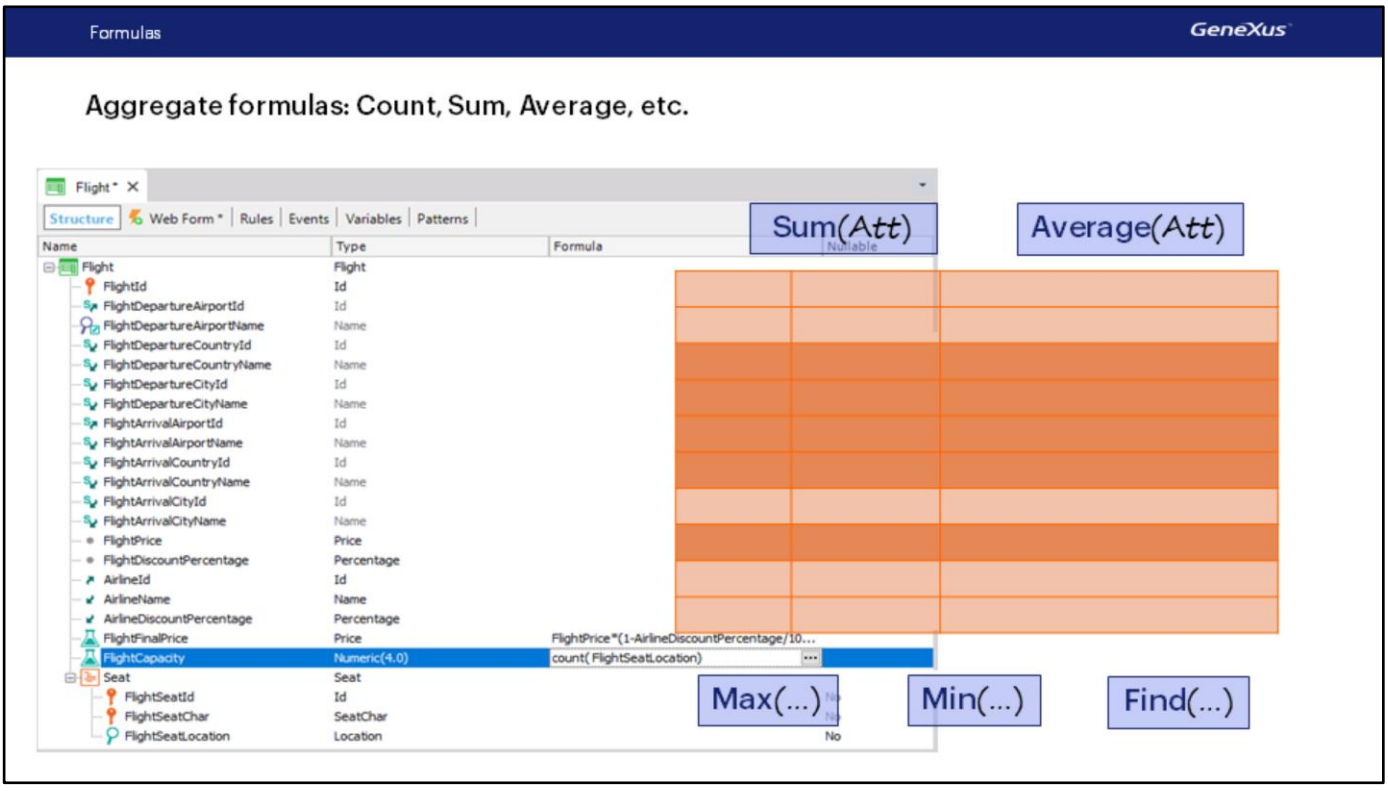
We run the Flight transaction... query flight number 1 and enter some seats:
- 1A - window
- 1B – middle
- 1C – aisle
- 1D – window
- 1E – middle
- 1F – aisle

As we enter the seats, note that the total number of seats is updated every time we add a new seat to the flight.

Lastly, we add:

- 2A – window... and stop here...

Let's return to GeneXus.

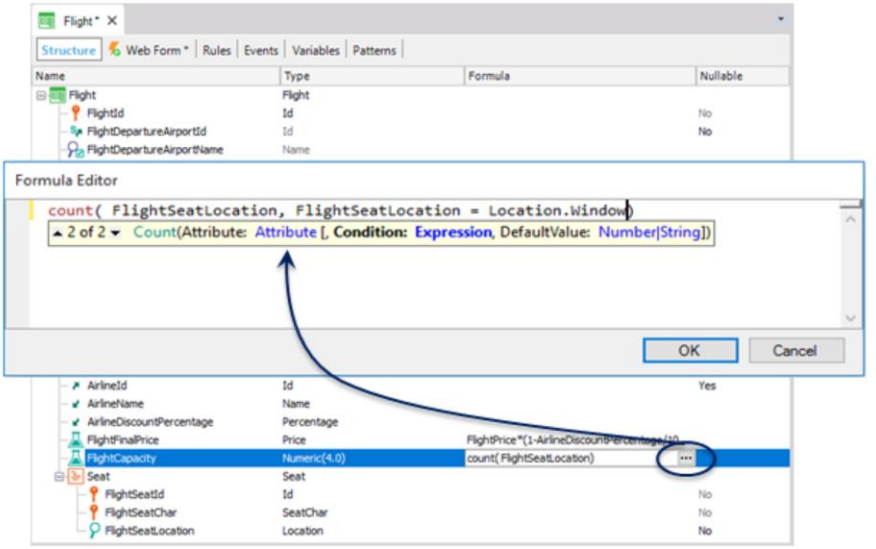


In addition, there are other Aggregate formulas **that make operations by taking several records into account.**

For example: **Sum**, to sum the values of the indicated attribute, **Average, to find the average of the indicated attribute values,** and others such as Max, Min and Find.

Aggregate formulas: Count, Sum, Average, etc.

Conditions can be added to count "certain" lines:



If we want to count not only the total seats in the flight we're positioned in, but also those that meet another condition, such as, for example, the number of seats next to the window, we can add this condition to the formula. In this way... since the FlightSeatLocation attribute belongs to the Location domain, and it has 3 enumerated values defined, the syntax to ask for the value taken by the attribute is as follows:

domain name, period, and the name associated with the value we want to filter by, which in this case is Window.

We click on OK.

We press F5.

We run the Flight transaction, record number 1; the flight's capacity is now 3, and it corresponds to the number of seats located next to a window, which matches the seats we entered in the seat grid.

In sum, we have seen that in addition to the implicit condition (when there are related records), it is also possible to count, sum, search for, maximize or average; that is to say, **add those records that comply with a certain explicit condition indicated by us. This condition is called "filtering condition" because it allows us to keep only those records that we're interested in.**

Lastly, we must remember that just like every other global formula, Aggregate formulas can have a "triggering condition." That is to say, the formula is only calculated when this condition is met.

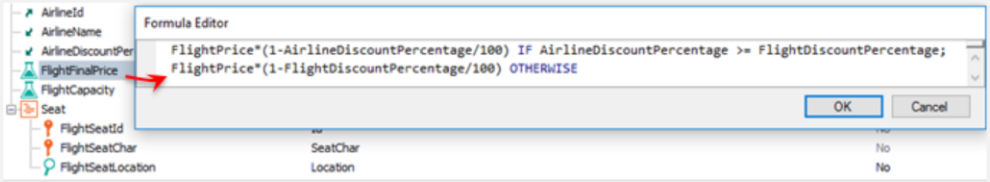
Summing up

Horizontal:

- To make a calculation, they access a record and, occasionally, those related through an extended table.

Attribute = $expression_1$ **if** $condition_1$;
$expression_2$ **if** $condition_2$;
...
$expression_n$ **if** $condition_n$;
$expression_o$ **otherwise**;

- Example: FlightFinalPrice



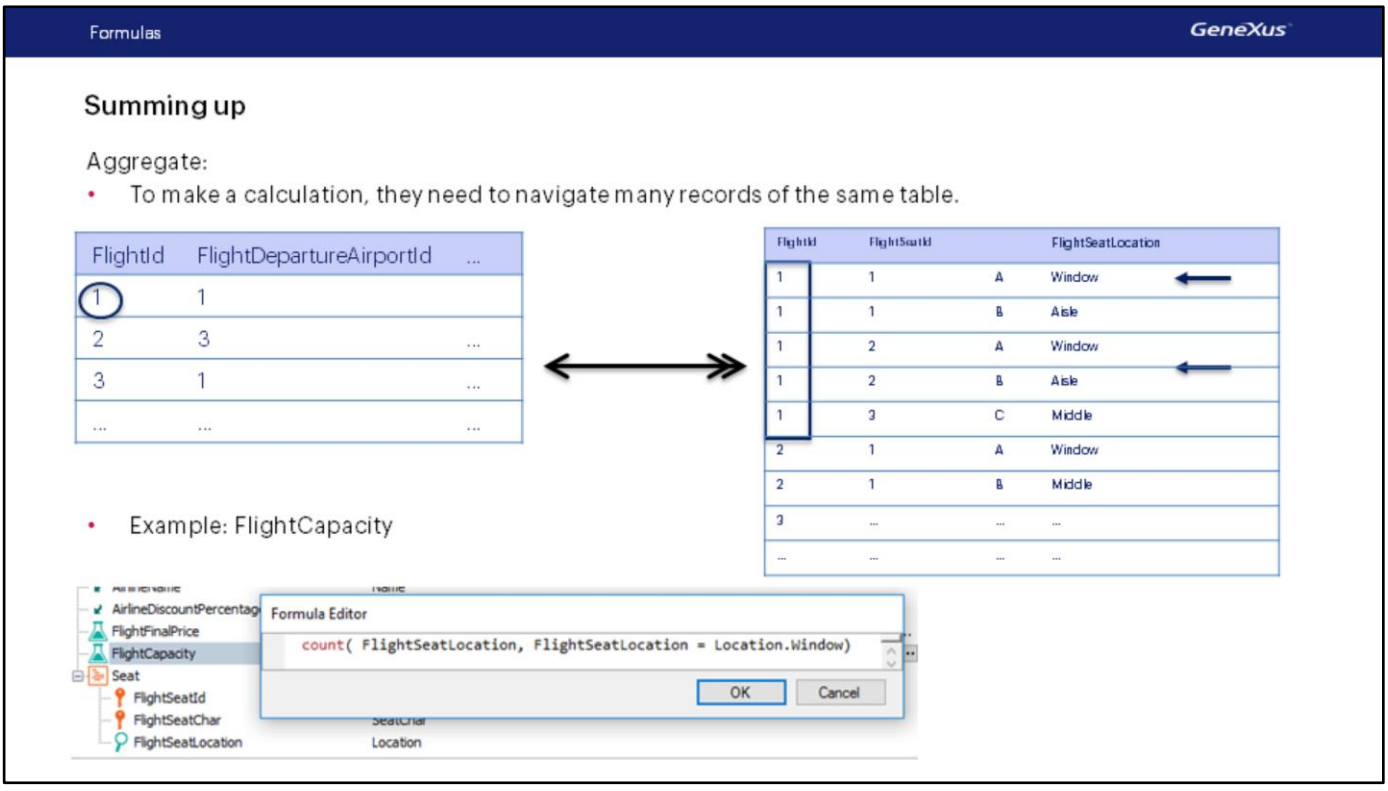
In sum: we've seen two types of formulas:

Horizontal formulas - they access a record to make a calculation. Also, these records may be related through the extended table.

That was the case of FlightFinalPrice.

These attributes belonged to the FLIGHT table, and the others to the Airline table.

As we saw in the example, we could set a formula attribute to be calculated in different ways depending on the value of a condition.

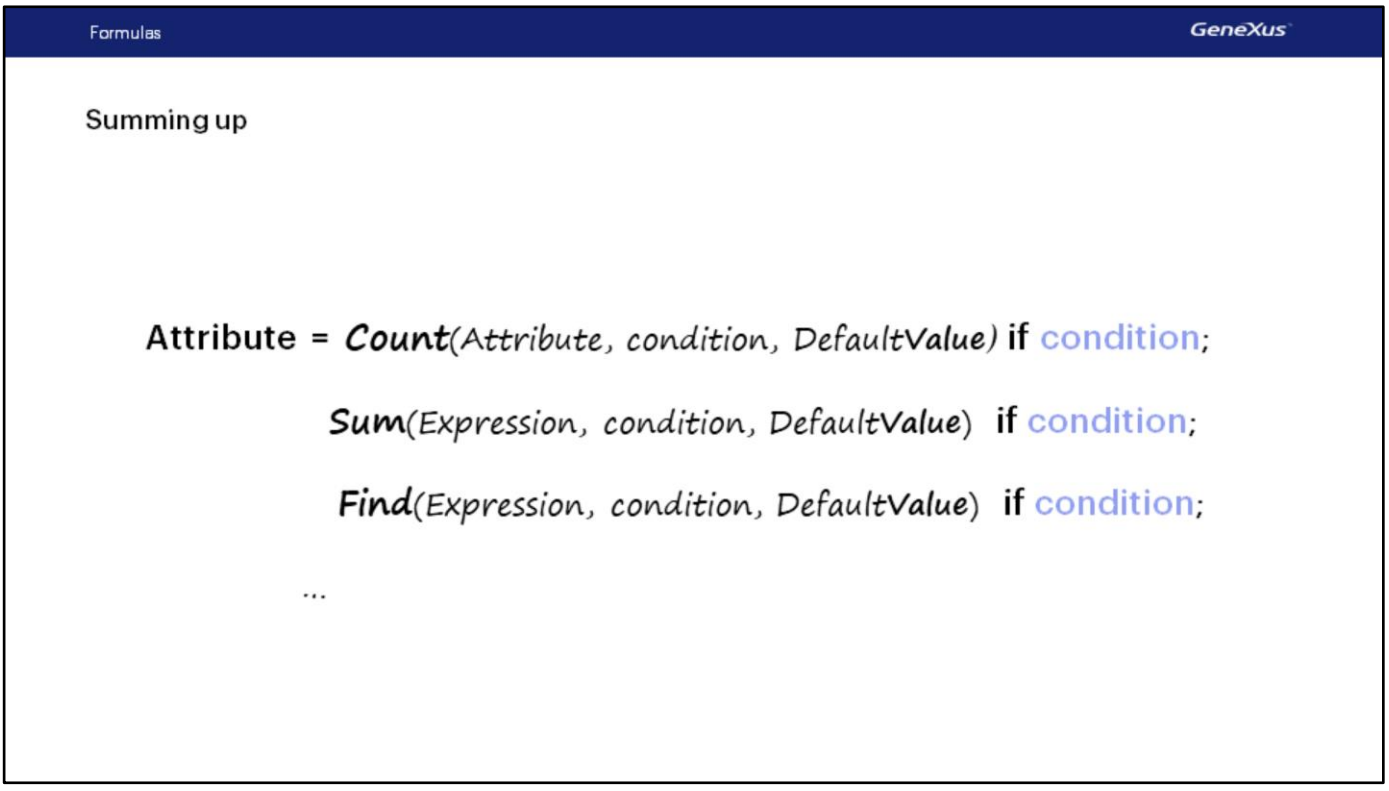


Aggregate formulas - to make their calculation, they need to navigate many records in the same table.

That was the case of FlightCapacity.

From the FLIGHT table associated with the formula attribute, it made a calculation over the FLIGHTSEAT table that contains the FlightSeatLocation attribute.

In this case, since the formula attribute is associated with a table, Flight, which has a 1-to-many relationship with the table over which the Count operation will be made, only the related records will be counted. If the relationship didn't exist, they would all be counted. In addition, because we indicate conditions for the records to be counted, the related records will only be counted as long as they meet that condition.



The filtering condition is the second parameter in the formula, and as a third parameter we can indicate a default value; that is to say, the value that will be returned by the formula if no record is found to count, sum, etc.

Just like horizontal formulas, aggregate formulas can be stated with conditions.

Summing up

Attribute = 2 + **Count**(Attribute, condition, DefaultValue) *

Sum(Expression, condition, DefaultValue) **if** condition;

$Att_1 + Att_2 * Att_3$ otherwise;

...

Also, horizontal formulas can be combined with aggregate formulas, providing a high degree of expressiveness in calculations, but we won't talk about it in this course.

Lastly, let's remove the filtering condition from the FlightCapacity attribute...

And send the new definitions to GeneXus Server.

Summing up

- **Aggregate:**
  - Sum
  - Average
  - Max
  - Min
  - Find
- The attribute referenced inside the formula's brackets provides information to GeneXus about the table to be navigated to perform the calculation.
- If GeneXus finds a relationship between the navigated table and the one where the formula attribute is stated, it will only consider the related records to make the calculation.
- If it doesn't find a relationship, GeneXus will perform the operation considering all the records in the navigated table.

GeneXus™

The power of doing.

Videos	training.genexus.com
Documentation	wiki.genexus.com
Certifications	training.genexus.com/certifications