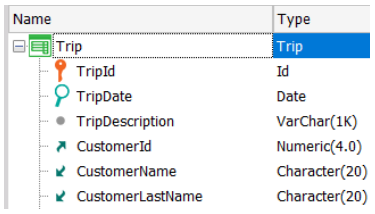
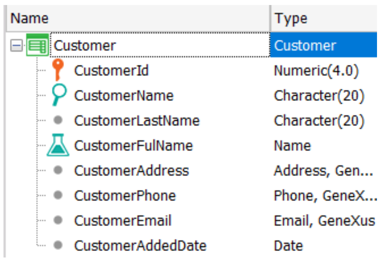
Redundant Attributes and their Maintenance

GeneXus™

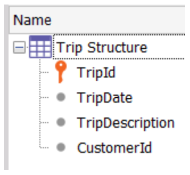
In this video, we will see how to define inferred attributes or formulas which by definition are not stored, as redundant and that they become attributes of a database table.



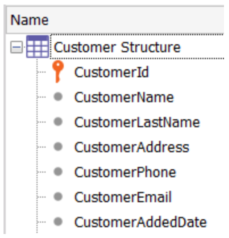
Name	Type
Trip	Trip
TripId	Id
TripDate	Date
TripDescription	VarChar(1K)
CustomerId	Numeric(4.0)
CustomerName	Character(20)
CustomerLastName	Character(20)



Name	Type
Customer	Customer
CustomerId	Numeric(4.0)
CustomerName	Character(20)
CustomerLastName	Character(20)
CustomerFulName	Name
CustomerAddress	Address, Gen...
CustomerPhone	Phone, GeneX...
CustomerEmail	Email, GeneXus
CustomerAddedDate	Date



Name
Trip Structure
TripId
TripDate
TripDescription
CustomerId



Name
Customer Structure
CustomerId
CustomerName
CustomerLastName
CustomerAddress
CustomerPhone
CustomerEmail
CustomerAddedDate

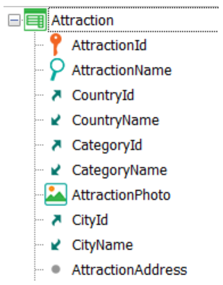
As we know, GeneXus automatically normalizes the database in Third Normal Form, which implies that the only attributes that can be in more than one table are the attributes that are primary keys, acting as foreign keys.

The rest of the attributes, which are called secondary attributes, are stored in a single table; if they are added to a transaction other than the one in which they were created, GeneXus will infer them, retrieving their value through the foreign key, their value from the table where they are stored. In addition, when we define an attribute as a formula in a transaction, it is no longer stored and becomes a virtual attribute.

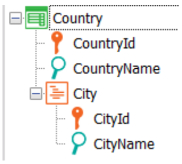
However, for performance reasons, in some cases we want to allow an inferred attribute to be stored in the table associated with the transaction where it is inferred; also, allow a formula attribute that must perform many calculations and takes a lot of time whenever its value is obtained, to be stored in its associated table in order to obtain the value more quickly.

GeneXus allows us to store an attribute that by default is not stored in a table, defining it as redundant.

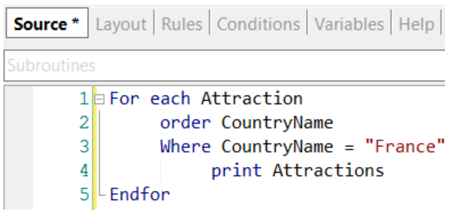
Referential redundancy



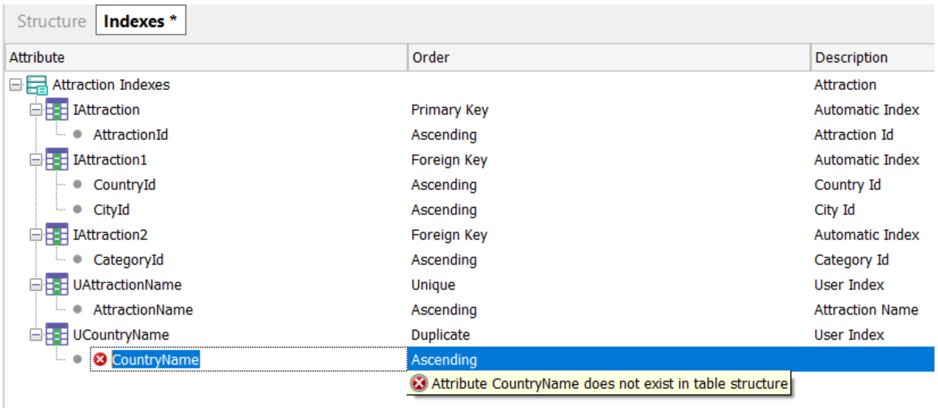
- Attraction
  - AttractionId
  - AttractionName
  - CountryId
  - CountryName
  - CategoryId
  - CategoryName
  - AttractionPhoto
  - CityId
  - CityName
  - AttractionAddress



- Country
  - CountryId
  - CountryName
  - City
    - CityId
    - CityName



```
For each Attraction
     order CountryName
     Where CountryName = "France"
          print Attractions
Endfor
```



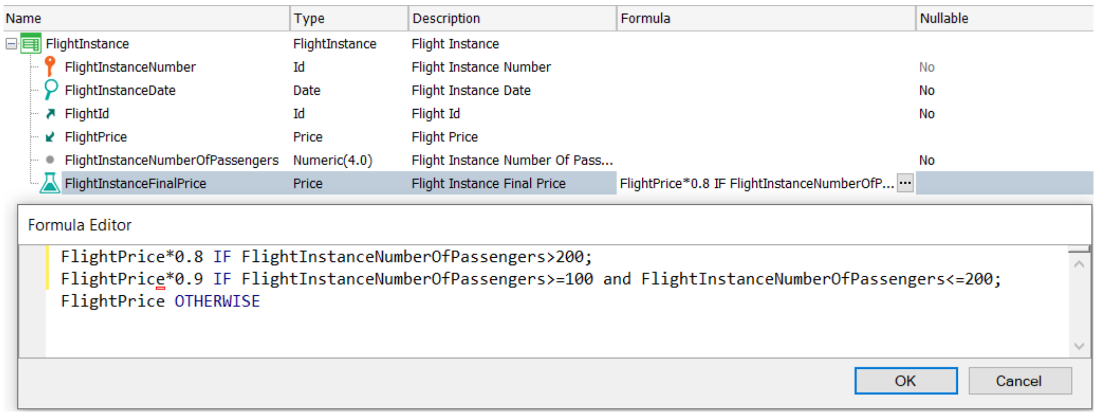
Attribute	Order	Description
Attraction Indexes		Attraction
IAttraction	Primary Key	Automatic Index
AttractionId	Ascending	Attraction Id
IAttraction1	Foreign Key	Automatic Index
CountryId	Ascending	Country Id
CityId	Ascending	City Id
IAttraction2	Foreign Key	Automatic Index
CategoryId	Ascending	Category Id
UAttractionName	Unique	User Index
AttractionName	Ascending	Attraction Name
UCountryName	Duplicate	User Index
CountryName	Ascending	

Attribute CountryName does not exist in table structure

When we make an inferred attribute redundant, it is called referential redundancy.

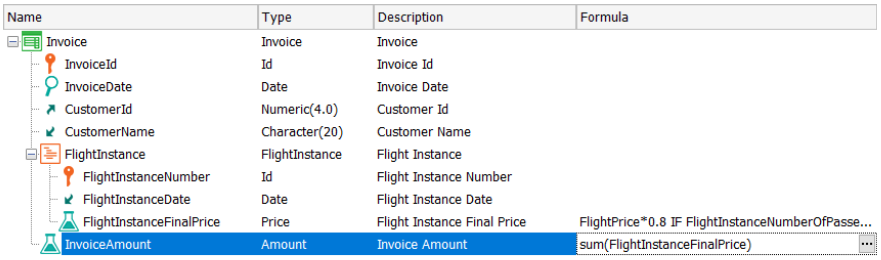
The purpose is to improve performance: for example, if we want to minimize the search for the inferred value from another table when there are many records involved, or when we want to include the inferred attribute in a user index, so the attribute must be stored in the table where we define the index.

Formula redundancy



Name	Type	Description	Formula	Nullable
FlightInstance	FlightInstance	Flight Instance		
FlightInstanceNumber	Id	Flight Instance Number		No
FlightInstanceDate	Date	Flight Instance Date		No
FlightId	Id	Flight Id		No
FlightPrice	Price	Flight Price		
FlightInstanceNumberOfPassengers	Numeric(4.0)	Flight Instance Number Of Pass...		No
FlightInstanceFinalPrice	Price	Flight Instance Final Price	FlightPrice*0.8 IF FlightInstanceNumberOfP...	

```
FlightPrice*0.8 IF FlightInstanceNumberOfPassengers>200;
FlightPrice*0.9 IF FlightInstanceNumberOfPassengers>=100 and FlightInstanceNumberOfPassengers<=200;
FlightPrice OTHERWISE
```



Name	Type	Description	Formula
Invoice	Invoice	Invoice	
InvoiceId	Id	Invoice Id	
InvoiceDate	Date	Invoice Date	
CustomerId	Numeric(4.0)	Customer Id	
CustomerName	Character(20)	Customer Name	
FlightInstance	FlightInstance	Flight Instance	
FlightInstanceNumber	Id	Flight Instance Number	
FlightInstanceDate	Date	Flight Instance Date	
FlightInstanceFinalPrice	Price	Flight Instance Final Price	FlightPrice*0.8 IF FlightInstanceNumberOfPasse...
InvoiceAmount	Amount	Invoice Amount	sum(FlightInstanceFinalPrice)

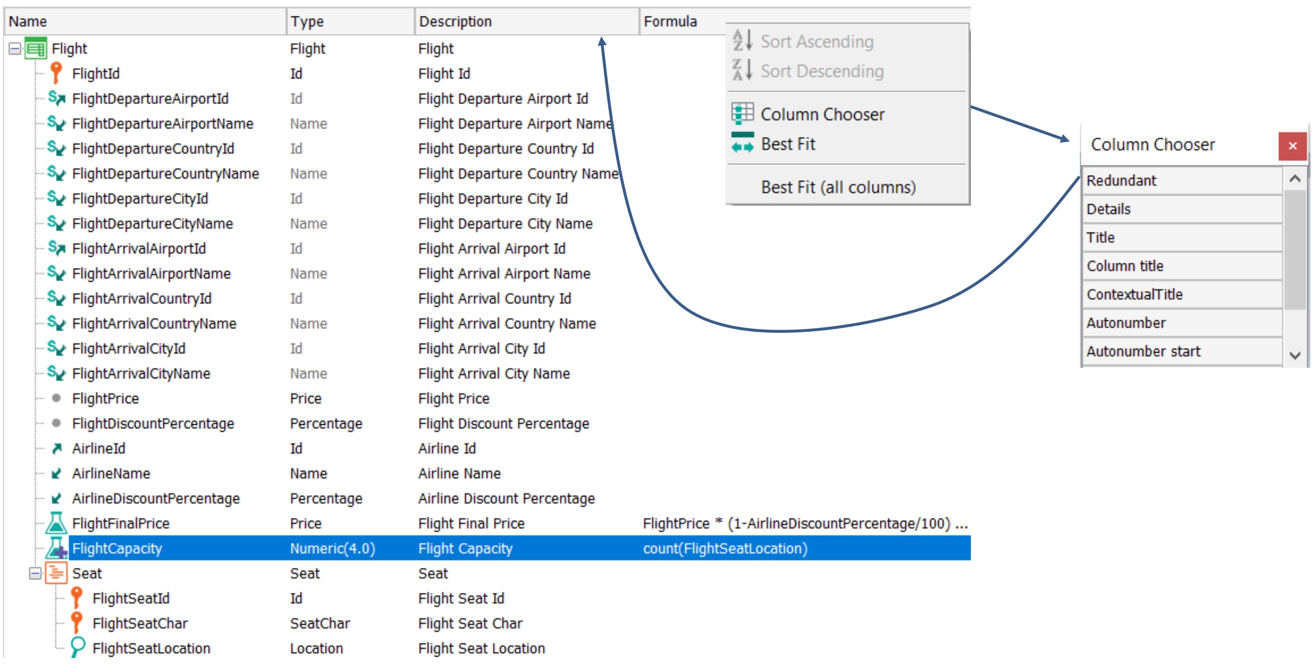
Formula redundancy is also intended to improve performance, especially when it is necessary to perform a large number of calculations to obtain the value of the formula.

This is especially valid for aggregate formulas that get their value from many records.

For this reason, in horizontal formulas it doesn’t seem to make much sense to define redundancies; however, in some cases, it is necessary to do so.

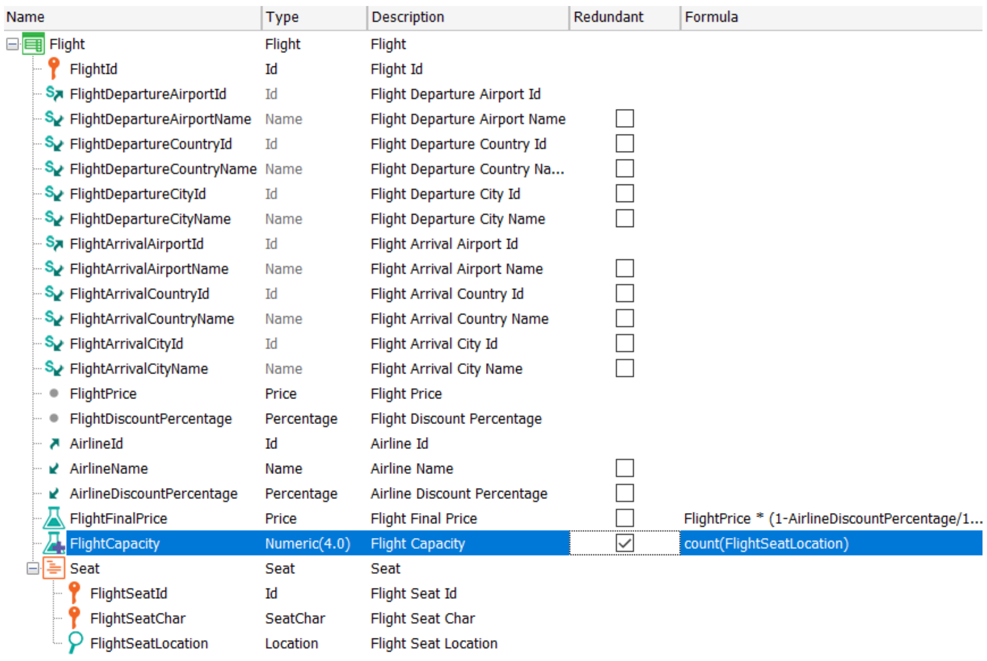
For example, if we want to define the formula attribute InvoiceAmount as redundant, we must first define as redundant the attribute FlightInstanceFinalPrice which is a horizontal formula.

How to define an attribute as redundant



Name	Type	Description	Formula
Flight	Flight	Flight	
FlightId	Id	Flight Id	
FlightDepartureAirportId	Id	Flight Departure Airport Id	
FlightDepartureAirportName	Name	Flight Departure Airport Name	
FlightDepartureCountryId	Id	Flight Departure Country Id	
FlightDepartureCountryName	Name	Flight Departure Country Name	
FlightDepartureCityId	Id	Flight Departure City Id	
FlightDepartureCityName	Name	Flight Departure City Name	
FlightArrivalAirportId	Id	Flight Arrival Airport Id	
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	
FlightDiscountPercentage	Percentage	Flight Discount Percentage	
AirlineId	Id	Airline Id	
AirlineName	Name	Airline Name	
AirlineDiscountPercentage	Percentage	Airline Discount Percentage	
FlightFinalPrice	Price	Flight Final Price	FlightPrice * (1-AirlineDiscountPercentage/100) ...
FlightCapacity	Numeric(4.0)	Flight Capacity	count(FlightSeatLocation)
Seat	Seat	Seat	
FlightSeatId	Id	Flight Seat Id	
FlightSeatChar	SeatChar	Flight Seat Char	
FlightSeatLocation	Location	Flight Seat Location	

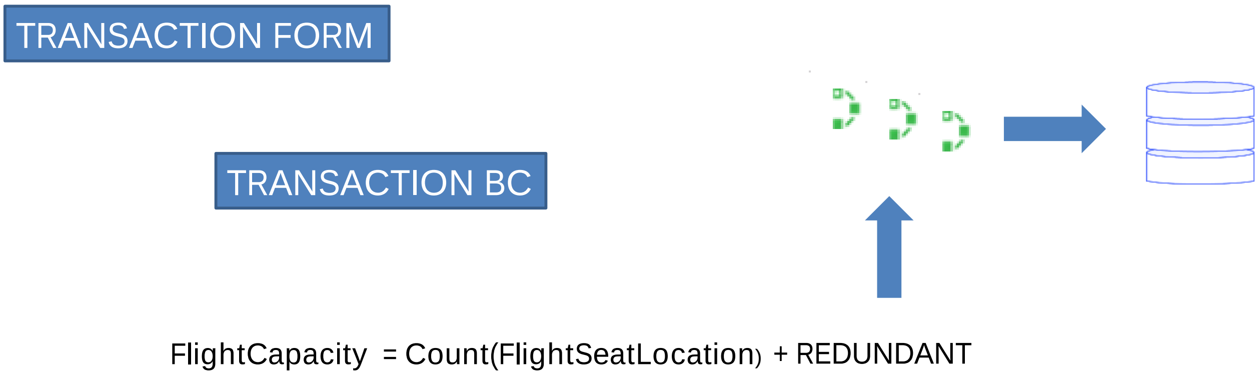
To define an attribute as redundant we do it from the transaction structure. We right-click on the column bar, click on Column Chooser and add the Redundant column by dragging it to the column bar.



Name	Type	Description	Redundant	Formula
Flight	Flight	Flight		
FlightId	Id	Flight Id		
FlightDepartureAirportId	Id	Flight Departure Airport Id		
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		
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		
FlightDiscountPercentage	Percentage	Flight Discount Percentage		
AirlineId	Id	Airline Id		
AirlineName	Name	Airline Name	☐	
AirlineDiscountPercentage	Percentage	Airline Discount Percentage	☐	
FlightFinalPrice	Price	Flight Final Price	☐	FlightPrice * (1-AirlineDiscountPercentage/1...
FlightCapacity	Numeric(4.0)	Flight Capacity	✓	count(FlightSeatLocation)
Seat	Seat	Seat		
FlightSeatId	Id	Flight Seat Id		
FlightSeatChar	SeatChar	Flight Seat Char		
FlightSeatLocation	Location	Flight Seat Location		

Note that the Redundant column is added and there we can select the checkbox to define the attribute as redundant. When we do this we see that a "+" sign has been added to the formula symbol to indicate that it is redundant.

Redundancy Maintenance



When we define an attribute as redundant, GeneXus creates procedures in charge of keeping the stored value updated when the record to which the formula belongs is edited.

When the value of a redundant formula attribute is changed by executing the transaction screen or a business component of the transaction, the formula is triggered, the calculation is performed, and the result is stored in the physical field of the database.

In the generated programs corresponding to transactions that involve redundant formula attributes, GeneXus incorporates routines to store the redundant data, whenever they are recalculated.

When querying the value of a redundant formula attribute, the formula is not triggered to obtain the calculation; instead, the stored value of the database field is taken.

When the value of some of the attributes that are part of the formula calculation changes, GeneXus will also trigger the update procedures for the redundant attribute. These procedures have the knowledge to calculate the new value and the result will be stored in the database.

Limitations when defining redundancies

- Subtypes **cannot** be defined as redundant.
- In order to define as redundant an attribute that is a formula, we must first define that formula as redundant.
- To change the definition of a formula that is redundant, we must first remove the redundancy.
- Formulas that add more than one level of non-redundant formulas will not be maintained correctly.

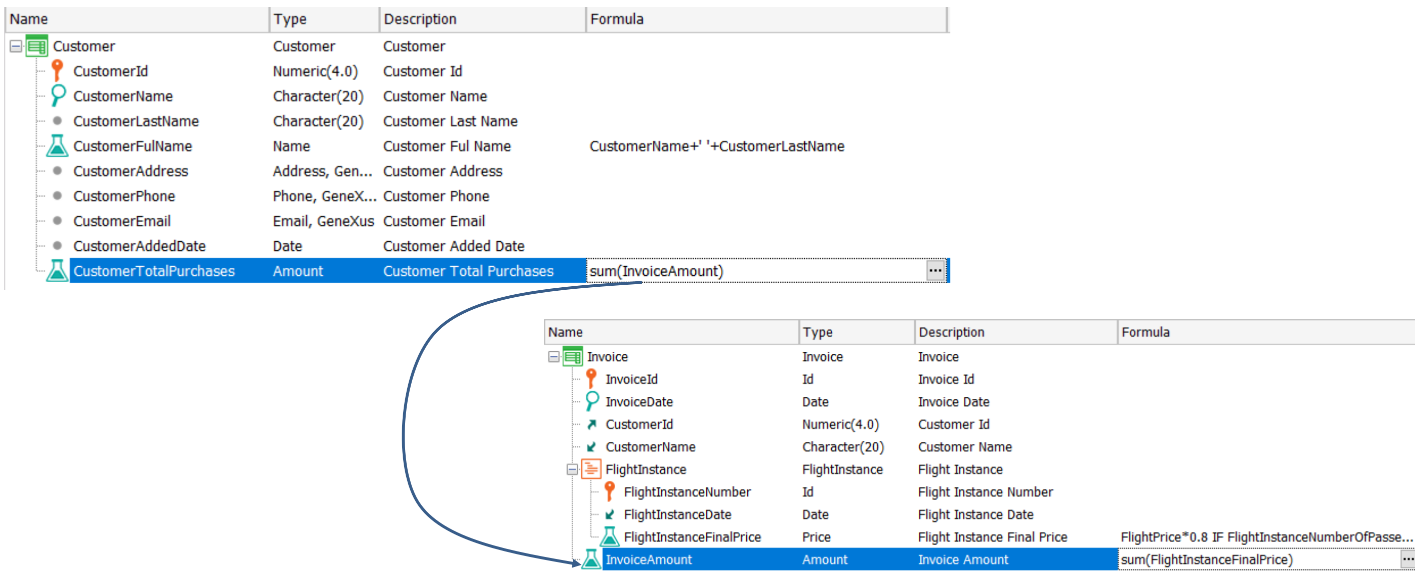
There are some limitations to define attributes as redundant, which we must consider.
First, we cannot define a subtype as redundant.

As we saw a few moments ago, in order to define an attribute that is a formula as redundant, we must first define that formula as redundant.

To change the definition of a formula that is redundant, we must first remove the redundancy, make the change and define the formula as redundant again.

In aggregation formulas that are made redundant, defined according to attributes that are also redundant formulas, their redundancies will not be properly maintained if more than one level is nested.

Example of formula redundancy constraints



Name	Type	Description	Formula
Customer	Customer	Customer	
CustomerId	Numeric(4.0)	Customer Id	
CustomerName	Character(20)	Customer Name	
CustomerLastName	Character(20)	Customer Last Name	
CustomerFulName	Name	Customer Ful Name	CustomerName+' '+CustomerLastName
CustomerAddress	Address, Gen...	Customer Address	
CustomerPhone	Phone, GeneX...	Customer Phone	
CustomerEmail	Email, GeneXus	Customer Email	
CustomerAddedDate	Date	Customer Added Date	
CustomerTotalPurchases	Amount	Customer Total Purchases	sum(InvoiceAmount)

Name	Type	Description	Formula
Invoice	Invoice	Invoice	
InvoiceId	Id	Invoice Id	
InvoiceDate	Date	Invoice Date	
CustomerId	Numeric(4.0)	Customer Id	
CustomerName	Character(20)	Customer Name	
FlightInstance	FlightInstance	Flight Instance	
FlightInstanceNumber	Id	Flight Instance Number	
FlightInstanceDate	Date	Flight Instance Date	
FlightInstanceFinalPrice	Price	Flight Instance Final Price	FlightPrice*0.8 IF FlightInstanceNumberOfPasse...
InvoiceAmount	Amount	Invoice Amount	sum(FlightInstanceFinalPrice)

Let's see an example where constraints appear when there is nesting of aggregate formulas.

Here, the CustomerTotalPurchases attribute is a Sum formula attribute of the InvoiceAmount attribute which is also a Sum formula attribute of the FlightInstanceFinalPrice attribute, which in turn is a horizontal formula.

If the CustomerTotalPurchases, InvoiceAmount and FlightInstanceFinalPrice attributes are defined as a redundant formula, only the CustomerTotalPurchases and InvoiceAmount attributes will be initialized and updated correctly, but not the FlightInstanceFinalPrice attribute.

Procedures created to maintain redundancy

GeneXus automatically creates the following procedures:

- **TableNameUpdateRedundancy** : Maintains redundancies of inferred attributes in tables where they are redundant

- **TableNameLoadRedundancy** : Recalculates and updates redundancies in TableName, both from formula and inferred attributes.

The programs used by GeneXus to maintain redundancies are created in the reorganizations that persist the attributes defined as redundant and are named:

- TableNameUpdateRedundancy
- TableNameLoadRedundancy

The program TableNameUpdateRedundancy is used to update inferred attribute redundancies. TableName is the name of the table where secondary attributes are stored.

Any transaction that defines a table whose secondary attributes are defined as redundant in other tables will call the TableNameUpdateRedundancy program, passing it as a parameter the primary key of that table.

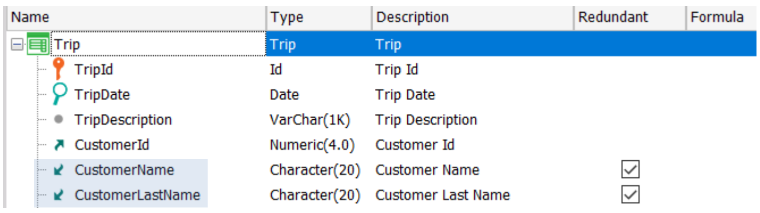
The program TableNameLoadRedundancy recalculates the redundancies of the TableName table, with TableName being the table where the redundant attributes are stored.

In the case of redundant inferred attributes, this procedure will access the table where the secondary attributes are stored and with their values will update the table where the redundant attributes are inferred.

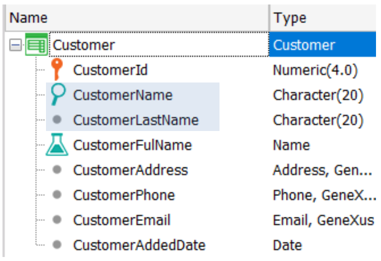
For redundant formula attributes, the procedure accesses the tables containing the attributes participating in the formula, performs the calculation and updates the redundant formula attribute in the table where it was defined.

Let's see some examples.

Secondary attribute redundancy



Name	Type	Description	Redundant	Formula
Trip	Trip	Trip		
TripId	Id	Trip Id		
TripDate	Date	Trip Date		
TripDescription	VarChar(1K)	Trip Description		
CustomerId	Numeric(4.0)	Customer Id		
CustomerName	Character(20)	Customer Name	✓	
CustomerLastName	Character(20)	Customer Last Name	✓	



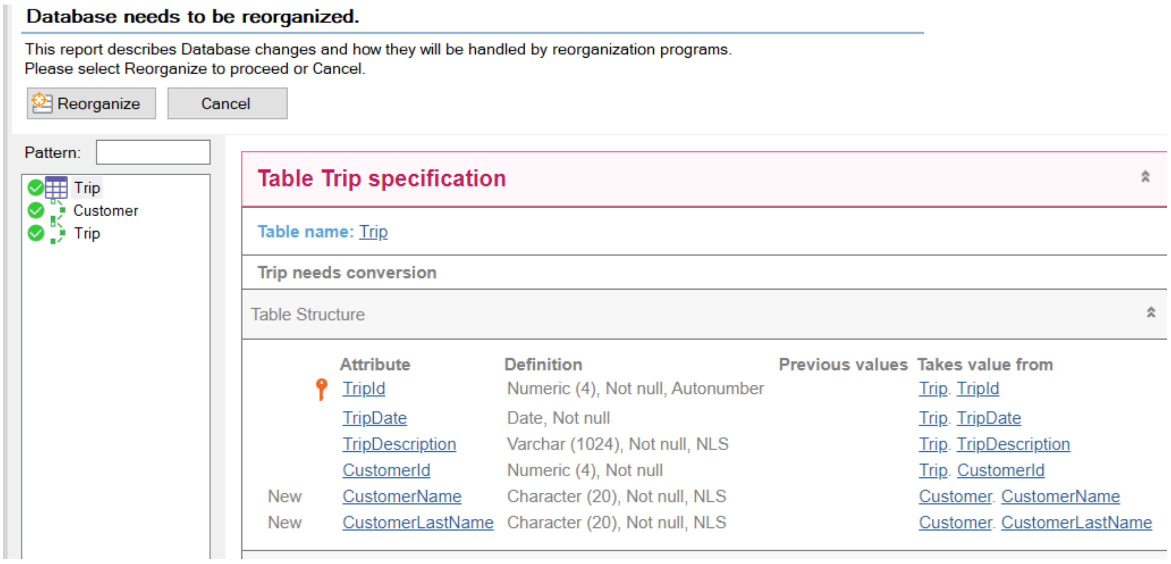
Name	Type
Customer	Customer
CustomerId	Numeric(4.0)
CustomerName	Character(20)
CustomerLastName	Character(20)
CustomerFulName	Name
CustomerAddress	Address, Gen...
CustomerPhone	Phone, GeneX...
CustomerEmail	Email, GeneXus
CustomerAddedDate	Date

Suppose that in the Trip transaction we want the inferred attributes CustomerName and CustomerLastName to be redundant.

Whenever the attributes CustomerName or CustomerLastName are modified in the Customer transaction, the CustomerUpdateRedundancy procedure will be automatically invoked to keep their values up to date.

Let's look at this in the impact analysis.

Secondary attribute redundancy (continued)



Database needs to be reorganized.

This report describes Database changes and how they will be handled by reorganization programs.
Please select Reorganize to proceed or Cancel.

Table Trip specification

Table name: Trip

Trip needs conversion

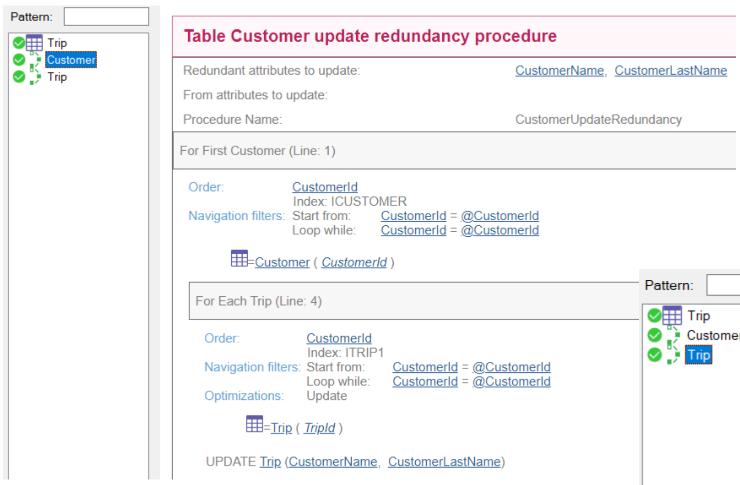
Table Structure

	Attribute	Definition	Previous values	Takes value from
	TripId	Numeric (4), Not null, Autonumber		Trip. TripId
	TripDate	Date, Not null		Trip. TripDate
	TripDescription	Varchar (1024), Not null, NLS		Trip. TripDescription
	CustomerId	Numeric (4), Not null		Trip. CustomerId
New	CustomerName	Character (20), Not null, NLS		Customer. CustomerName
New	CustomerLastName	Character (20), Not null, NLS		Customer. CustomerLastName

First, we see that the Trip table needs to be reorganized, since the stored attributes CustomerName and CustomerLastName that were inferred will be created.

In addition, we see that two procedures appear, one named Customer and another one named Trip.

Secondary attribute redundancy (continued)


Pattern:

- Trip
  - Customer
  - Trip

Table Customer update redundancy procedure

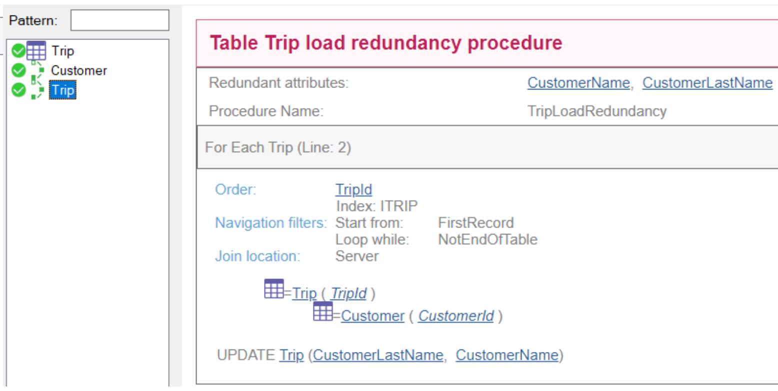
Redundant attributes to update: CustomerName, CustomerLastName
From attributes to update:
Procedure Name: CustomerUpdateRedundancy

For First Customer (Line: 1)

```
Order:              CustomerId
                    Index: ICUSTOMER
Navigation filters: Start from:   CustomerId = @CustomerId
                    Loop while:   CustomerId = @CustomerId

        =Customer ( CustomerId )
```

For Each Trip (Line: 4)

```
Order:              CustomerId
                    Index: ITRIP1
Navigation filters: Start from:   CustomerId = @CustomerId
                    Loop while:   CustomerId = @CustomerId
Optimizations:      Update

        =Trip ( TripId )

UPDATE Trip (CustomerName, CustomerLastName)
```


Pattern:

- Trip
  - Customer
  - Trip

Table Trip load redundancy procedure

Redundant attributes: CustomerName, CustomerLastName
Procedure Name: TripLoadRedundancy

For Each Trip (Line: 2)

```
Order:              TripId
                    Index: ITRIP
Navigation filters: Start from:   FirstRecord
                    Loop while:   NotEndOfTable
Join location:      Server

        =Trip ( TripId )
            =Customer ( CustomerId )

UPDATE Trip (CustomerLastName, CustomerName)
```

If we click on the procedure that reads Customer, we see that its name is actually CustomerUpdateRedundancy.

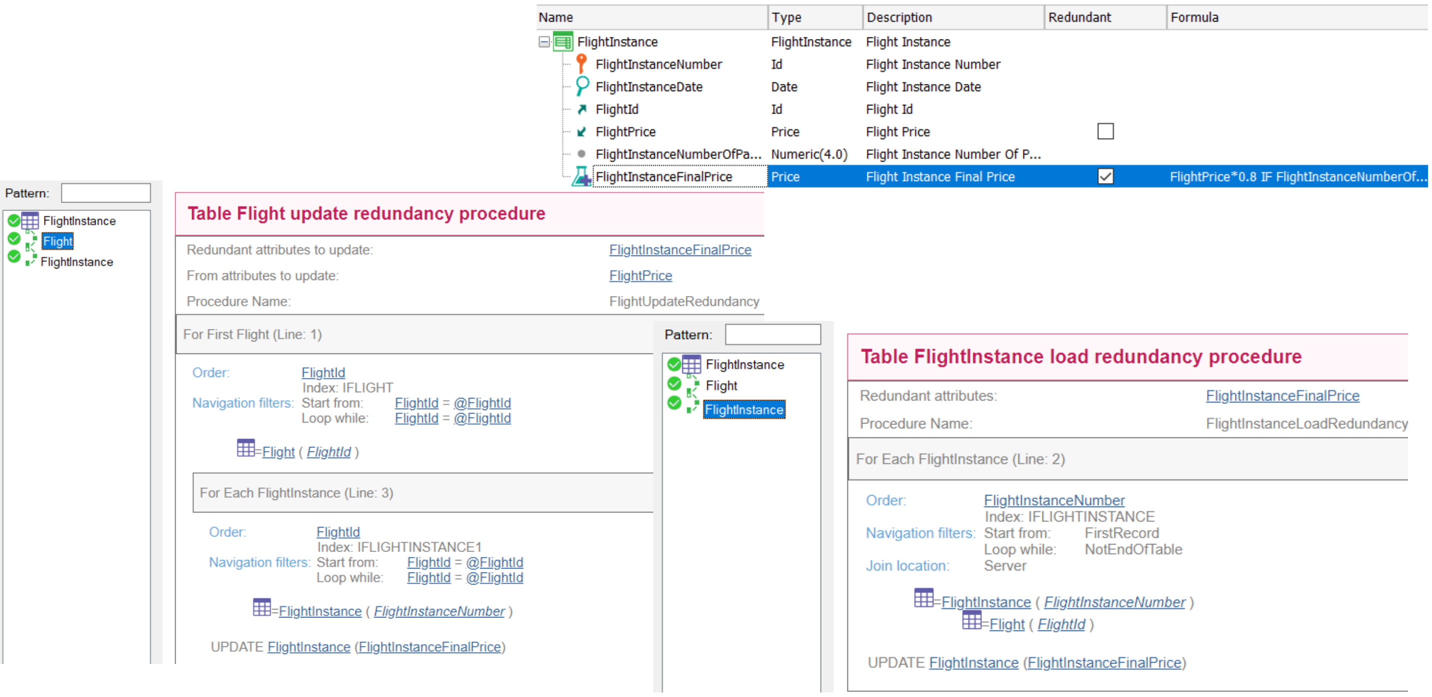
It accesses the Customer table filtering by CustomerId (we see that it says For First Customer) and then accesses the Trip table filtering by the value of the CustomerId foreign key, to do an UPDATE on it.

This procedure is triggered when the CustomerName and CustomerLastName attributes of the Customer table change and is responsible for updating the CustomerName and CustomerLastName values that are redundant in Trip.

If we select the procedure that reads Trip, we see that its name is TripLoadRedundancy.

This procedure is responsible for triggering the update in the Trip table of the CustomerName and CustomerLastName attributes.

Formula attribute redundancy



Name	Type	Description	Redundant	Formula
FlightInstance	FlightInstance	Flight Instance		
FlightInstanceNumber	Id	Flight Instance Number		
FlightInstanceDate	Date	Flight Instance Date		
FlightId	Id	Flight Id		
FlightPrice	Price	Flight Price	☐	
FlightInstanceNumberOfPa...	Numeric(4.0)	Flight Instance Number Of P...		
FlightInstanceFinalPrice	Price	Flight Instance Final Price	✓	FlightPrice*0.8 IF FlightInstanceNumberOf...

Pattern:

FlightInstance
Flight
FlightInstance

Table Flight update redundancy procedure

Redundant attributes to update:	FlightInstanceFinalPrice
From attributes to update:	FlightPrice
Procedure Name:	FlightUpdateRedundancy

For First Flight (Line: 1)

Order: FlightId
Index: IFLIGHT
Navigation filters: Start from: FlightId = @FlightId
Loop while: FlightId = @FlightId

=Flight (FlightId)

For Each FlightInstance (Line: 3)

Order: FlightId
Index: IFLIGHTINSTANCE1
Navigation filters: Start from: FlightId = @FlightId
Loop while: FlightId = @FlightId

=FlightInstance (FlightInstanceNumber)

UPDATE FlightInstance (FlightInstanceFinalPrice)

Pattern:

FlightInstance
Flight
FlightInstance

Table FlightInstance load redundancy procedure

Redundant attributes:	FlightInstanceFinalPrice
Procedure Name:	FlightInstanceLoadRedundancy

For Each FlightInstance (Line: 2)

Order: FlightInstanceNumber
Index: IFLIGHTINSTANCE
Navigation filters: Start from: FirstRecord
Loop while: NotEndOfTable
Join location: Server

=FlightInstance (FlightInstanceNumber)
=Flight (FlightId)

UPDATE FlightInstance (FlightInstanceFinalPrice)

In this example, we define the formula attribute FlightInstanceFinalPrice as redundant.
We see that, in addition to the reorganization of the FlightInstance table where the FlightInstanceFinalPrice attribute will be created as stored, the procedures named Flight and FlightInstance appear.

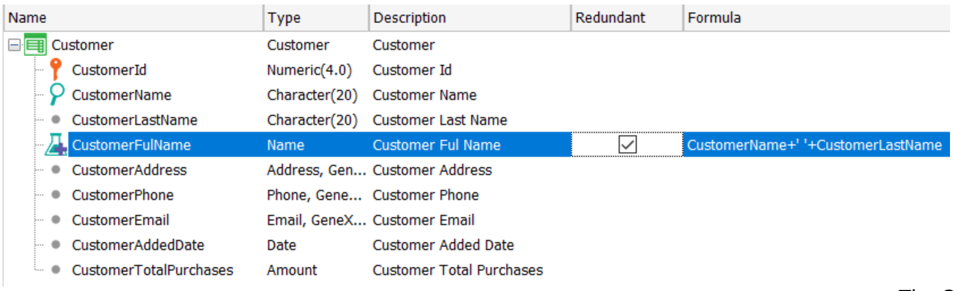
If we look at the impact analysis of the Flight procedure, we see that it is called FlightUpdateRedundancy and that it will update the redundant attribute FlightInstanceFinalPrice, from the value of the FlightPrice attribute that integrates the formula.

To do so, it accesses the Flight table filtered by its primary key FlightId and then runs a For each on the FlightInstance table filtered by foreign key FlightId, where it will perform an UPDATE, updating the FlightInstanceFinalPrice attribute.

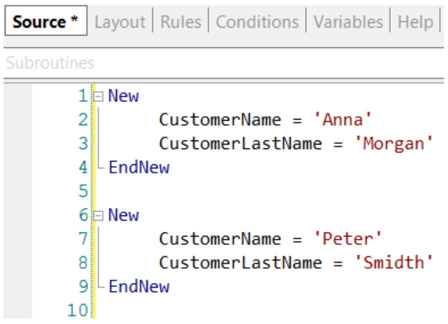
If we click on FlightInstance, we see that it will run through the FlightInstance table, accessing the Flight table to recalculate the formula and update the FlightInstanceFinalPrice value stored in the FlightInstance table.

Rebuild redundancy

Program: GXLRED

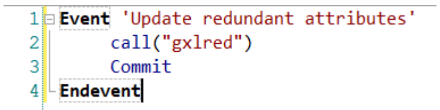


Name	Type	Description	Redundant	Formula
Customer	Customer	Customer		
CustomerId	Numeric(4.0)	Customer Id		
CustomerName	Character(20)	Customer Name		
CustomerLastName	Character(20)	Customer Last Name		
CustomerFulName	Name	Customer Ful Name	✓	CustomerName+' '+CustomerLastName
CustomerAddress	Address, Gen...	Customer Address		
CustomerPhone	Phone, Gene...	Customer Phone		
CustomerEmail	Email, GeneX...	Customer Email		
CustomerAddedDate	Date	Customer Added Date		
CustomerTotalPurchases	Amount	Customer Total Purchases		



```
New
      CustomerName = 'Anna'
      CustomerLastName = 'Morgan'
EndNew

New
      CustomerName = 'Peter'
      CustomerLastName = 'Smidth'
EndNew
```

The CustomerFullName attribute will not be updated automatically; to do so, we invoke GXLRED:



```
Event 'Update redundant attributes'
      call("gxlred")
      Commit
Endevent
```

Although GeneXus maintains redundancies automatically, in some cases it is necessary to update them explicitly.

An example is when we have a redundant attribute that we want to update from a procedure with a New.

In the example, the formula attribute CustomerFullName gets its value from the concatenation of the attributes CustomerName and CustomerLastName and has been defined as redundant.
In the source of the procedure, two records are created in the Customer table, by means of parallel New clauses.

In this case, the value of the CustomerFullName attribute defined as redundant in the Customer table will not be updated.

For such cases, there is the 'Rebuild Redundancy' utility, which allows updating some of the redundancies defined in a KB.

To run this utility, we must invoke the GXLRED program, which in turn will invoke all redundancy programs named <tablename> loadredundancy, (such as, for example: CustomerLoadRedundancy).

We can invoke this program from an event using the Call method and then make Commit.

Redundant attributes and NULL

GeneXus automatically manages whether a redundant attribute is NULL:

- Redundant inferred attributes:
  - If the original attribute is NULL, the redundant one is NULL
  - If the original attribute has Nullable = No, the redundant one is NULL only if the FK is NULL.

- Redundant formula attributes

More information about redundant attributes: https://wiki.genexus.com/commwiki/servlet/wiki?6661

Another aspect to consider is the assignment of a redundant attribute as Nullable.

Developers cannot manage the nullability of redundant attributes. GeneXus calculates it automatically according to the following criteria:

For inferred attributes, if the redundancy's original attribute is nullable, then the redundant attribute is also nullable.

If the original attribute has Nullable set to No, the nullability of the redundant attribute will depend on the nullability of the corresponding foreign key.
If the corresponding foreign key can be null, then the redundant attribute can also be null. Otherwise, the redundant attribute will not be null.

Global formulas defined as redundant do not accept null values.

To learn more about this topic and about defining inferred attributes or formula attributes as redundant in general, please visit the following wiki link: https://wiki.genexus.com/commwiki/servlet/wiki?6661

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