

Atributos redundantes e sua manutenção

GeneXus™

Neste vídeo, veremos como definir atributos inferidos ou fórmulas, que por definição não são armazenados, como redundantes e que se tornem atributos de uma tabela da base de dados.

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)
CustomerFullName	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

Como sabemos, GeneXus normaliza automaticamente a base de dados na Terceira Forma Normal, o que implica que os únicos atributos que podem estar em mais de uma tabela são os atributos que são chave primária, cumprindo funções de chave estrangeira.

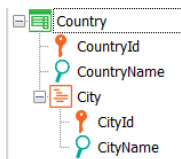
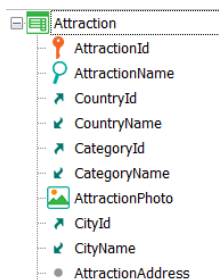
O restante dos atributos, aos quais chamamos de secundários, são armazenados em uma única tabela e no caso em que se agreguem a uma transação diferente daquela em que foram criados, GeneXus se encarrega de inferi-los, recuperando por meio da chave estrangeira, seu valor a partir da tabela onde estão armazenados.

Além disso, quando definimos um atributo como fórmula em uma transação, deixa de ser armazenado e se torna um atributo virtual.

No entanto, por motivos de desempenho, em alguns casos queremos permitir que um atributo inferido seja armazenado na tabela associada à transação onde é inferido, ou que um atributo fórmula que deve realizar muitos cálculos e consome muito tempo cada vez que é obtido seu valor, seja armazenado em sua tabela associada para obter o valor mais rapidamente .

GeneXus permite que possamos armazenar um atributo que por padrão não está armazenado em uma tabela, definindo-o como redundante.

Redundância referencial



Source * | Layout | Rules | Conditions | Variables | Help

Subroutines

```

1 For each Attraction
2   order CountryName
3   Where CountryName = "France"
4   print Attractions
5 Endfor

```

Structure | **Indexes ***

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

Quando redundamos um atributo inferido, se denomina redundância referencial.

O motivo é melhorar o desempenho, por exemplo, se queremos minimizar a busca do valor inferido a partir de outra tabela quando há muitos registros envolvidos, ou quando queremos incluir o atributo inferido em um índice de usuário, então o atributo deverá estar armazenado na tabela onde definimos o índice.

Redundância de fórmulas

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...	

Formula Editor

```
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)

A redundância de fórmulas também tem como objetivo melhorar o desempenho, especialmente quando é necessário realizar uma grande quantidade de cálculos para obter o valor da fórmula.

Isto é especialmente válido para fórmulas aggregate que obtêm seu valor a partir de muitos registros.

Por este motivo, nas fórmulas horizontais não parece fazer muito sentido definir redundâncias, todavia, há momentos em que devemos fazer.

Por exemplo, se quisermos definir o atributo fórmula InvoiceAmount como redundante, devemos primeiro definir como redundante o atributo FlightInstanceFinalPrice, que é uma fórmula horizontal.

Como definimos um atributo como redundante

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	$FlightPrice * (1 - AirlineDiscountPercentage / 100) \dots$
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	

The 'Column Chooser' dialog box shows the following options:

- Redundant
- Details
- Title
- Column title
- ContextualTitle
- Autonumber
- Autonumber start

Para definir um atributo como redundante, fazemos isso a partir da estrutura da transação, clicando com o botão direito do mouse na barra de colunas, clicamos em Column Chooser e adicionamos a coluna Redundant arrastando-a para a barra de colunas.

Name	Type	Description	Redundant	Formula
Flight	Flight	Flight		
FlightId	Id	Flight Id		
FlightDepartureAirportId	Id	Flight Departure Airport Id		
FlightDepartureAirportName	Name	Flight Departure Airport Name	<input type="checkbox"/>	
FlightDepartureCountryId	Id	Flight Departure Country Id	<input type="checkbox"/>	
FlightDepartureCountryName	Name	Flight Departure Country Na...	<input type="checkbox"/>	
FlightDepartureCityId	Id	Flight Departure City Id	<input type="checkbox"/>	
FlightDepartureCityName	Name	Flight Departure City Name	<input type="checkbox"/>	
FlightArrivalAirportId	Id	Flight Arrival Airport Id		
FlightArrivalAirportName	Name	Flight Arrival Airport Name	<input type="checkbox"/>	
FlightArrivalCountryId	Id	Flight Arrival Country Id	<input type="checkbox"/>	
FlightArrivalCountryName	Name	Flight Arrival Country Name	<input type="checkbox"/>	
FlightArrivalCityId	Id	Flight Arrival City Id	<input type="checkbox"/>	
FlightArrivalCityName	Name	Flight Arrival City Name	<input type="checkbox"/>	
FlightPrice	Price	Flight Price		
FlightDiscountPercentage	Percentage	Flight Discount Percentage		
AirlineId	Id	Airline Id		
AirlineName	Name	Airline Name	<input type="checkbox"/>	
AirlineDiscountPercentage	Percentage	Airline Discount Percentage	<input type="checkbox"/>	
FlightFinalPrice	Price	Flight Final Price	<input type="checkbox"/>	FlightPrice * (1-AirlineDiscountPercentage/1...
FlightCapacity	Numeric(4,0)	Flight Capacity	<input checked="" type="checkbox"/>	count(FlightSeatLocation)
Seat	Seat	Seat		
FlightSeatId	Id	Flight Seat Id		
FlightSeatChar	SeatChar	Flight Seat Char		
FlightSeatLocation	Location	Flight Seat Location		

Observamos que é adicionada a coluna Redundant e ali podemos marcar o checkbox para definir o atributo como redundante. Quando fazemos isso, vemos que ao símbolo da fórmula foi adicionado um sinal de “+” para indicar que é redundante.

Manutenção de redundâncias

TRANSACTION FORM

TRANSACTION BC

FlightCapacity = Count(FlightSeatLocation) + REDUNDANT



Quando definimos um atributo como redundante, GeneXus cria procedimentos encarregados de manter atualizado o valor armazenado, quando se edita o registro ao qual pertence a fórmula.

Ao alterar o valor de um atributo fórmula redundante executando a tela da transação ou um business component da mesma, a fórmula é disparada realizando o cálculo e o resultado é armazenado no campo físico da base de dados.

Nos programas gerados correspondentes a transações que têm envolvidos atributos fórmulas redundantes, GeneXus incorpora rotinas que se encarregam de armazenar os dados redundantes, sempre que sejam recalculados.

Ao consultar o valor de um atributo fórmula redundante, não é disparada a fórmula para obter o cálculo, mas é obtido o valor armazenado do campo da base de dados.

Quando muda o valor de algum dos atributos que fazem parte do cálculo da fórmula, GeneXus também disparará os procedimentos de atualização do atributo redundante. Estes procedimentos têm o conhecimento de como calcular o novo valor e o resultado será armazenado na base de dados.

Limitações na definição de redundâncias

- Os subtipos não podem ser definidos como redundantes
- Para definir como redundante um atributo que é fórmula, devemos primeiro definir essa fórmula como redundante
- Para alterar a definição de uma fórmula que é redundante, deve-se primeiro remover a redundância
- As fórmulas que incluem mais de um nível de fórmulas não redundantes não serão mantidas corretamente.

Existem algumas limitações para definir atributos como redundantes, que devemos considerar.

Em primeiro lugar, não podemos definir um subtipo como redundante.

Como vimos há alguns momentos, para ser possível definir como redundante um atributo que é fórmula, devemos primeiro definir essa fórmula como redundante.

Para alterar a definição de uma fórmula que é redundante, deve primeiro remover a redundância, fazer a alteração e definir a fórmula como redundante novamente.

E nas fórmulas de agregação que se tornam redundantes, definidas com base em atributos que também são fórmulas redundantes, suas redundâncias não serão mantidas corretamente se mais de um nível for aninhado.

Exemplo de Limitações em redundância de fórmulas

Name	Type	Description	Formula
Customer	Customer	Customer	
CustomerId	Numeric(4,0)	Customer Id	
CustomerName	Character(20)	Customer Name	
CustomerLastName	Character(20)	Customer Last Name	
CustomerFullName	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)

Vejamos um exemplo em que aparecem as limitações quando há aninhamento de fórmulas aggregate.

Aqui, o atributo CustomerTotalPurchases é um atributo fórmula Sum do atributo InvoiceAmount que também é um atributo fórmula Sum do atributo FlightInstanceFinalPrice, o qual por sua vez é uma fórmula horizontal.

Se os atributos CustomerTotalPurchases, InvoiceAmount e FlightInstanceFinalPrice são definidos como fórmula redundante, apenas serão inicializados e atualizados corretamente os atributos CustomerTotalPurchases e InvoiceAmount, mas não o atributo FlightInstanceFinalPrice.

Procedimentos criados para manter a redundância

GeneXus cria automaticamente os procedimentos:

- **TableNameUpdateRedundancy** : Mantém redundâncias de atributos inferidos nas tabelas onde são redundantes
- **TableNameLoadRedundancy** : Recalcula e atualiza redundâncias em TableName, tanto de atributos fórmulas quanto inferidos.

Os programas utilizados por GeneXus para manter as redundâncias são criados nas reorganizações que persistem aos atributos definidos como redundantes e são denominados:

- TableNameUpdateRedundancy
- TableNameLoadRedundancy

O programa TableNameUpdateRedundancy é utilizado para atualizar as redundâncias de atributo inferidos. TableName é o nome da tabela onde estão armazenados os atributos secundários.

Toda transação que defina uma tabela cujos atributos secundários estão definidos redundantes em outras tabelas, chamará o programa TableNameUpdateRedundancy, passando-lhe como parâmetro a chave primária dessa tabela.

O programa TableNameLoadRedundancy recalcula as redundâncias da tabela TableName, sendo TableName a tabela onde estão armazenados os atributos redundantes.

No caso de atributos inferidos redundantes, este procedimento acessará a tabela onde estão armazenados os atributos secundários e com seus valores atualizará a tabela onde estão inferidos redundantes.

No caso de atributos fórmula redundantes, o procedimento acessa as tabelas onde estão os atributos que participam da fórmula, realiza o cálculo e atualiza o atributo fórmula redundante na tabela onde foi definido.

Vejamos alguns exemplos.

Redundância de atributo secundário

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	<input checked="" type="checkbox"/>	
CustomerLastName	Character(20)	Customer Last Name	<input checked="" type="checkbox"/>	

Name	Type
Customer	Customer
CustomerId	Numeric(4,0)
CustomerName	Character(20)
CustomerLastName	Character(20)
CustomerFullName	Name
CustomerAddress	Address, GeneXus
CustomerPhone	Phone, GeneXus
CustomerEmail	Email, GeneXus
CustomerAddedDate	Date

Suponhamos que na transação Trip queremos que os atributos inferidos CustomerName e CustomerLastName sejam redundantes.


Cada vez que na transação Customer seja modificado o atributo CustomerName ou CustomerLastName, será invocado automaticamente o procedimento CustomerUpdateRedundancy para manter seus valores atualizados.

Vejamos isto na análise de impacto.

Redundância de atributo secundário (cont.)

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.

 Reorganize

Pattern:





-  Trip
-  Customer
-  Trip

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

Em primeiro lugar, vemos que a tabela Trip precisa ser reorganizada, uma vez que serão criados os atributos armazenados CustomerName e CustomerLastName que eram inferidos.

Vemos também que aparecem dois procedimentos, um com o nome Customer e outro Trip.

Redundância de atributo secundário (cont.)

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 \(CustomerLastName, CustomerName\)](#)

Se clicamos sobre procedimento que diz Customer, vemos que seu nome é na verdade CustomerUpdateRedundancy.

Ele acessa a tabela Customer filtrando por CustomerId (vemos que diz For First Customer) e então acessa a tabela Trip filtrando pelo valor da chave estrangeira CustomerId, para fazer um UPDATE nela.

Este procedimento é disparado quando mudam os atributos CustomerName e CustomerLastName da tabela Customer e se encarrega de atualizar os valores CustomerName e CustomerLastName que são redundantes em Trip.

Se selecionamos o procedimento que diz Trip, vemos que seu nome é TripLoadRedundancy.

Este procedimento se encarrega de disparar a atualização na tabela Trip dos atributos CustomerName e CustomerLastName.

Redundância de atributo fórmula

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	<input type="checkbox"/>	
FlightInstanceNumberOfPa...	Numeric(4,0)	Flight Instance Number Of P...		
FlightInstanceFinalPrice	Price	Flight Instance Final Price	<input checked="" type="checkbox"/>	FlightPrice*0.8 IF FlightInstanceNumberOf...

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](#))

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](#))

Neste exemplo, definimos o atributo fórmula FlightInstanceFinalPrice como redundante.

Vemos que além da reorganização da tabela FlightInstance onde será criado o atributo FlightInstanceFinalPrice como armazenado, aparecem os procedimentos chamados Flight e FlightInstance.

Se observamos a análise de impacto do procedimento Flight, vemos que se chama FlightUpdateRedundancy e que atualizará o atributo redundante FlightInstanceFinalPrice, a partir do valor do atributo FlightPrice que integra a fórmula.

Para fazer isso, acessa a tabela Flight filtrada por sua chave primária FlightId e depois faz um For Each na tabela FlightInstance filtrada pela chave estrangeira FlightId, onde realizará um UPDATE, atualizando o atributo FlightInstanceFinalPrice.

Se clicamos sobre FlightInstance, vemos que percorrerá a tabela FlightInstance, acessando a tabela Flight para recalcular a fórmula e atualizar o valor de FlightInstanceFinalPrice armazenado na tabela FlightInstance.

Rebuild redundancy

Programa: 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		
CustomerFullName	Name	Customer Ful Name	<input checked="" type="checkbox"/>	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		

```
Source * | Layout | Rules | Conditions | Variables | Help
Subroutines
1 | New
2 |   CustomerName = 'Anna'
3 |   CustomerLastName = 'Morgan'
4 | EndNew
5 |
6 | New
7 |   CustomerName = 'Peter'
8 |   CustomerLastName = 'Smidth'
9 | EndNew
10|
```

O atributo CustomerFullName não será atualizado automaticamente, para fazer isso invocamos GXLRED:

```
1 | Event 'Update redundant attributes'
2 |   call("gxlred")
3 |   Commit
4 | Endevent
```

Embora GeneXus mantenha as redundâncias de forma automática, em alguns casos é necessário atualizá-las explicitamente.

Um exemplo é quando temos um atributo redundante que queremos atualizar a partir de um procedimento com um New.

No exemplo, o atributo fórmula CustomerFullName obtém seu valor da concatenação dos atributos CustomerName e CustomerLastName e foi definido como redundante.

No source do procedimento, são criados dois registros na tabela Customer, mediante cláusulas News paralelas.

Neste caso, não será atualizado o valor do atributo CustomerFullName definido como redundante na tabela Customer

Para tais casos existe o utilitário 'Rebuild Redundancy', que permite atualizar algumas das redundâncias definidas em uma KB.

Para executar este utilitário, devemos invocar o programa GXLRED, que por sua vez invocará todos os programas de redundância chamados <tablename>loadredundancy, (como, por exemplo: CustomerLoadRedundancy)

Podemos invocar este programa a partir de um evento utilizando o método call e então fazemos Commit.

Atributos redundantes e NULL

GeneXus gerencia automaticamente se um atributo redundante é NULL:

- Atributos inferidos redundantes:
 - Se o atributo original é NULL, o redundante é NULL
 - Se o atributo original tiver Nullable = No, o redundante é NULL apenas se a FK for NULL
- Atributos fórmula redundantes

Para mais informações sobre atributos redundantes: <https://wiki.genexus.com/commwiki/servlet/wiki?6661>

Outro tema a considerar é a atribuição de um atributo redundante como Nullable.

Os desenvolvedores não podem gerenciar a nulabilidade dos atributos redundantes. GeneXus o calcula automaticamente com os seguintes critérios:

Para atributos inferidos, se o atributo original da redundância for nullable, então o atributo redundante também será nullable.

Se o atributo original tem Nullable em No, a nulabilidade do atributo redundante dependerá da nulabilidade da chave estrangeira correspondente.

Se a chave estrangeira correspondente pode ser nula, então o atributo redundante também pode ser nulo. Caso contrário, o atributo redundante será não nulo.

As fórmulas globais definidas como redundantes não permitem valores nulos.

Se deseja saber mais sobre este tema e em geral sobre como definir atributos inferidos ou atributos fórmulas como redundantes, o convidamos a visitar o seguinte link da wiki: <https://wiki.genexus.com/commwiki/servlet/wiki?6661>

GeneXus™

training.genexus.com

wiki.genexus.com

training.genexus.com/certifications