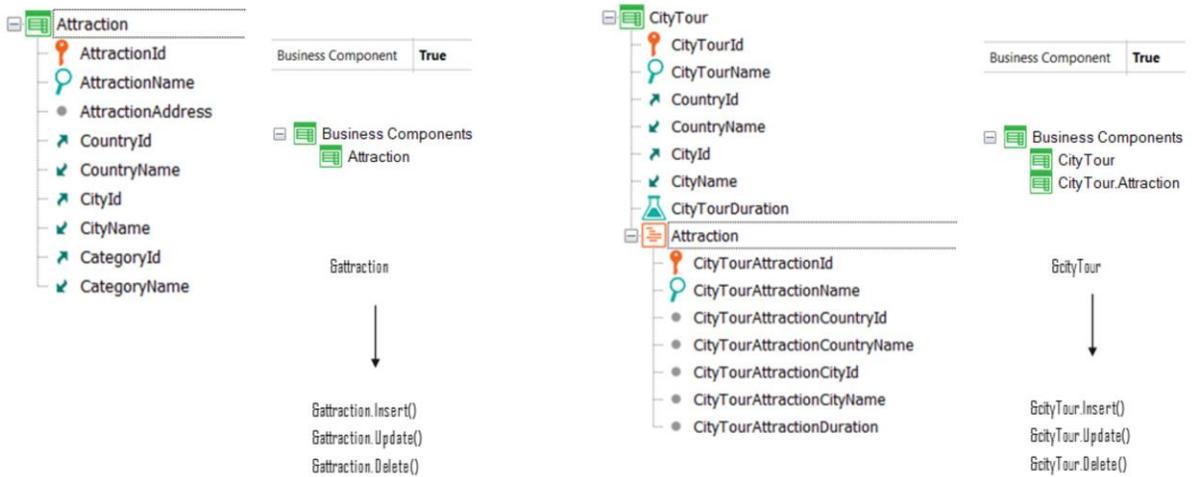


Atualização da base de dados

Comparação entre business component de um nível e de dois

GeneXus™



Havíamos visto como inserir, modificar e excluir informações da base de dados provenientes de transações de um e dois níveis através de business components.

Em todos os casos, devíamos carregar a estrutura de dados do Business Component em uma variável e realizar sobre ela a operação desejada, de maneira análoga a como fazíamos com a transação.

A diferença entre Business Component de um nível ou dois apresentou-se apenas em relação ao trabalho com as linhas.

Single-level BC / Header of a Two-level BC





Insert

```

Gattraction = new()
Gattraction.AttractionName = "Eiffel Tower"
Gattraction.CountryId = find( CountryId, CountryName = "France" )
Gattraction.CityId = find( CityId, CityName = "Paris" )
Gattraction.CategoryId = find( CategoryId, CategoryName = "Monument" )

Gattraction.Insert()
Commit
endif

```

Update

Delete

Attraction



AttractionId	3
AttractionName	Eiffel Tower
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CategoryId	2
CategoryName	Monument

Em relação ao primeiro nível, a forma de trabalho é idêntica.

Assim, por exemplo, para inserir uma atração pedíamos memória, atribuíamos valor aos elementos da estrutura e, então, invocávamos o método Insert, que inseria o novo registro na base de dados, se tudo estivesse bem. A variável ficava carregada com os valores atuais da base de dados e em modo Update.

Single-level BC / Header of a Two-level BC

Attraction

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

Business Components

- Attraction

```

Insert
    Gattraction = new()
    Gattraction.AttractionName = "Eiffel Tower"
    Gattraction.CountryId = find( CountryId, CountryName = "France" )
    Gattraction.CityId = find( CityId, CityName = "Paris" )
    Gattraction.CategoryId = find( CategoryId, CategoryName = "Monument" )

    Gattraction.Insert()
    Commit
endif

Update
    Gattraction = load(3)
    Gattraction.CategoryId = find( CategoryId, CategoryName = "Tourist site" )

    if Gattraction.Update()
        Commit
    endif

Delete
    Gattraction.load(3)
    Gattraction.Delete()

    if Gattraction.Success()
        Commit
    endif

```

Gattraction

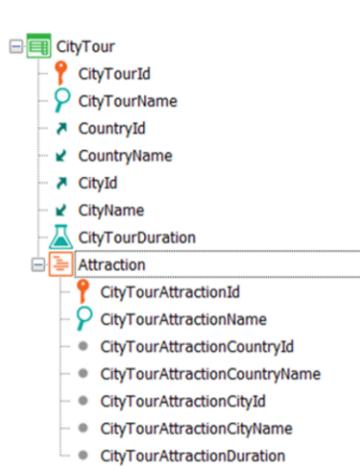


AttractionId	3
AttractionName	Eiffel Tower
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CategoryId	2
CategoryName	Monument

Para atualizar uma atração, carregávamos primeiro a variável com o método Load, atribuíamos o novo valor ao elemento ou elementos que queríamos modificar e, então, invocávamos o método Update, para realizar a atualização do registro na base de dados. Se esta for bem-sucedida, a variável fica carregada com os valores atuais e em modo Update.

Por último, para excluir uma atração, a carregávamos na variável com o método Load e, então, invocávamos o método Delete, que era o que efetivamente a excluía da base de dados se tudo estivesse bem. A variável fica carregada com os valores que o registro tinha antes de ser excluído e em modo Delete.

Two-level BC



CityTour

Header	
CityTourId	2
CityTourName	Paris at night
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CityTourDuration	300
Attraction	

CityTour.Load(2)

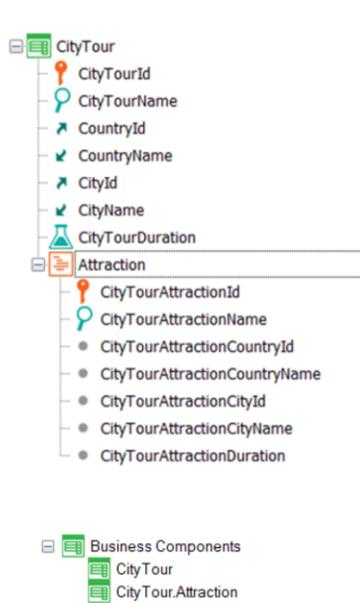
Lines

CityTourAttractionId	3
CityTourAttractionName	Eifel Tower
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	250

CityTourAttractionId	5
CityTourAttractionName	Notre Dame
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

É para inserir, atualizar ou excluir linhas, se o cabeçalho existia, então, primeiro carregávamos na variável BC, cabeçalho e linhas e, logo, manipulávamos a coleção de linhas.

Two-level BC



CityTour

Header	
CityTourId	2
CityTourName	Paris at night
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CityTourDuration	300
Attraction	

CityTour.Load(2)

CityTour.Attraction...

Lines

CityTourAttractionId	3
CityTourAttractionName	Eifel Tower
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	200

Update

CityTourAttractionId	5
CityTourAttractionName	Notre Dame
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

Delete

CityTourAttractionId	1
CityTourAttractionName	
CityTourAttractionCountryId	
CityTourAttractionCountryName	
CityTourAttractionCityId	
CityTourAttractionCityName	
CityTourAttractionDuration	120

Insert

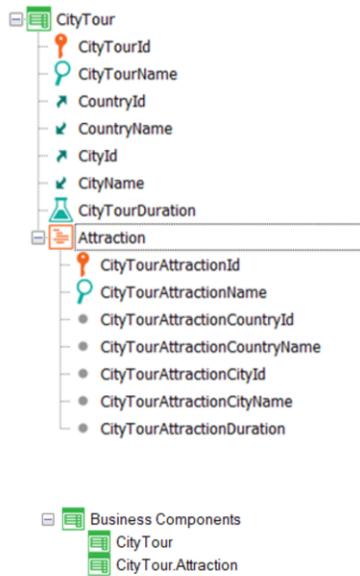
Por exemplo, se o city tour 2 tem duas linhas, uma para a Torre Eiffel e outra para a catedral de Notre Dame e quiséssemos adicionar uma nova, para o museu do Louvre, modificar a duração da visita à Torre Eiffel e excluir do tour a visita à Notre Dame... ou seja, inserir, modificar e excluir uma linha... teríamos que trabalhar com a coleção de linhas...

Two-level BC



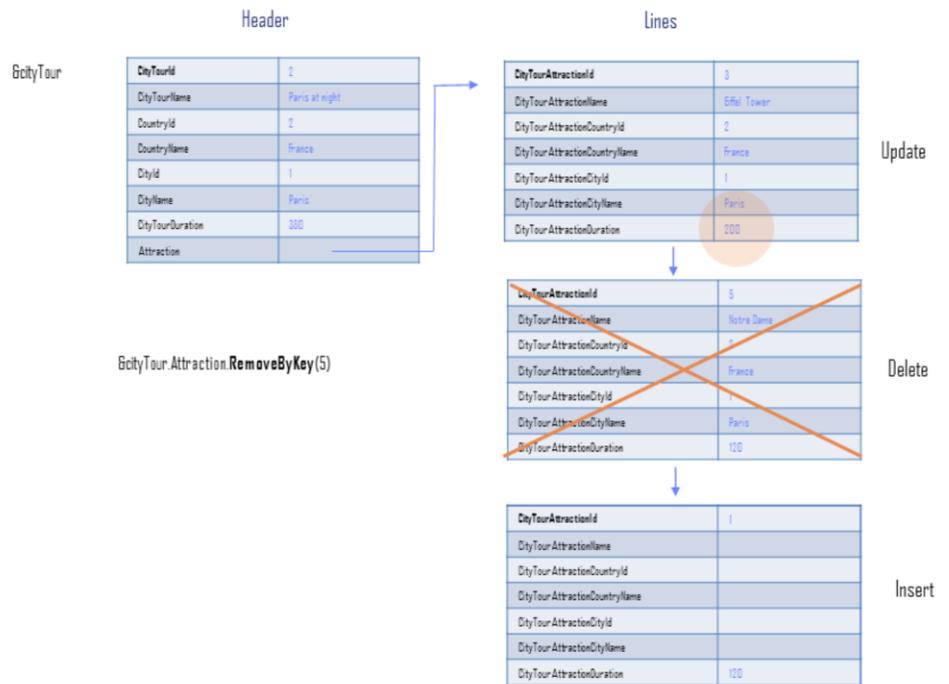
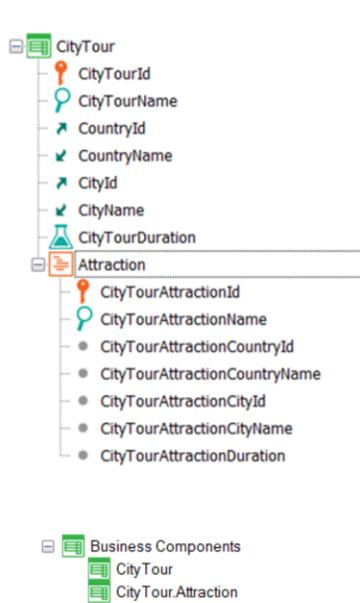
Para inserir uma, era necessário solicitar espaço de memória para uma variável do tipo BC das linhas, que terá seus elementos vazios. Atribuir valor aos seus elementos e, em seguida, utilizar o método add de coleções, para inserir a variável Gattraction.

Two-level BC



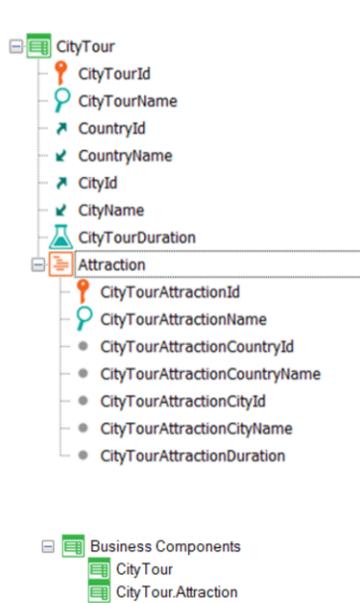
Para modificar uma linha, primeiro a obtemos na variável `Gattraction` BC das linhas, com o método `GetByKey`. Em seguida, simplesmente atualizamos o elemento que nos interessa.

Two-level BC



E, por último, para excluir una línea, utilizamos el método `RemoveByKey` de la colección de líneas de la variable `&cityTour`.

Two-level BC



GetCityTour

Header	
CityTourId	2
CityTourName	Paris at night
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CityTourDuration	300
Attraction	

GetCityTour.Load(2)

GetCityTour.Attraction...

```

if GetCityTour.Update()
  Commit
else
  --
endif
    
```

Lines

CityTourAttractionId	3
CityTourAttractionName	Eiffel Tower
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	200

Update

CityTourAttractionId	5
CityTourAttractionName	Notre Dame
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

Delete

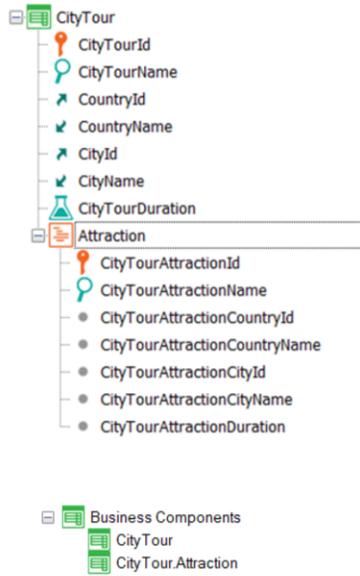
CityTourAttractionId	1
CityTourAttractionName	Louvre Museum
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

Insert

Depois de ter realizado as três operações sobre as linhas, invocamos o método Update, que é o que efetivamente impacta o realizado na variável na base de dados, executando as regras da transação, bem como os controles de integridade referencial.

Se for bem-sucedida, a variável fica carregada com os valores atuais, incluindo o caso de atributos inferidos e fórmulas. E fica, logicamente, em modo Update.

Two-level BC



GoCityTour

Header	
CityTourId	2
CityTourName	Paris at night
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CityTourDuration	200
Attraction	

GoCityTour **Load**(2)

```

Gattraction = new()
Gattraction.CityTourAttractionId = 1
Gattraction.CityTourAttractionDuration = 120
GoCityTour Attraction.add(Gattraction)
    
```

```

Gattraction = GoCityTour Attraction GetByKey(3)
Gattraction.CityTourAttractionDuration = 200
    
```

GoCityTour **Attraction RemoveByKey**(5)

```

if GoCityTour Update()
    Commit
endif
    
```

Lines

CityTourAttractionId	3
CityTourAttractionName	Eifel Tower
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	200

Update

CityTourAttractionId	5
CityTourAttractionName	Notre Dame
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

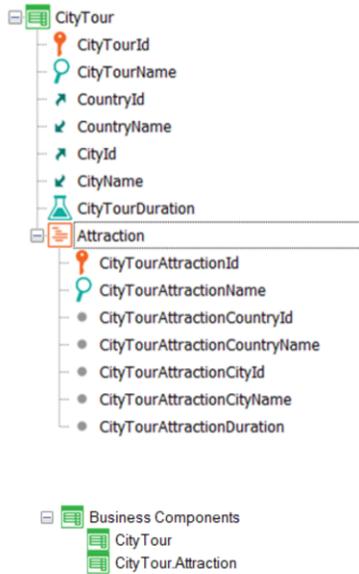
Delete

CityTourAttractionId	1
CityTourAttractionName	Louvre Museum
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

Insert

Aqui vemos o código completo do programa, na ordem em que fomos operando com as linhas.

Two-level BC



GoCityTour

Header	
CityTourId	2
CityTourName	Paris oh la la
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CityTourDuration	300
Attraction	

GoCityTour Load(2)

GoCityTour CityTourName = "Paris oh, la, la"

```

GAttraction = new()
GAttraction.CityTourAttractionId = 1
GAttraction.CityTourAttractionDuration = 120
GoCityTour.Attraction.add(GAttraction)

GAttraction = GoCityTour.Attraction.GetByKey(3)
GAttraction.CityTourAttractionDuration = 200
  
```

GoCityTour Attraction RemoveByKey(5)

```

if GoCityTour.Update()
  Commit
endif
  
```

Lines

CityTourAttractionId	3
CityTourAttractionName	Eifel Tower
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	200

Update

CityTourAttractionId	5
CityTourAttractionName	Notre Dame
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

Delete

CityTourAttractionId	1
CityTourAttractionName	Louvre Museum
CityTourAttractionCountryId	2
CityTourAttractionCountryName	France
CityTourAttractionCityId	1
CityTourAttractionCityName	Paris
CityTourAttractionDuration	120

Insert

É claro que nada nos impede de atualizar também informações do cabeçalho.

Neste caso, alteramos o nome do city tour antes de trabalhar com as linhas, depois trabalhamos com as linhas e, então, fazemos Update...

Observemos que aqui a ordem não importa. Poderíamos ter modificado o atributo do cabeçalho em qualquer outro momento antes do Update.

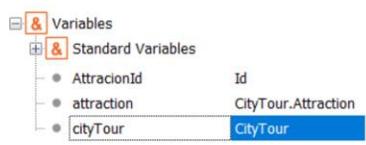
Methods to operate on the database

- Load(), Update(), Delete(), Insert()
- Save(), Asignation instead of Load, InsertOrUpdate()

```
&cityTour ...
...
&cityTour.Save()
```

```
&cityTour ...
...
&cityTour.Insert()
```

```
&cityTour ...
...
&cityTour.Update()
```

	TrnMode.Insert	TrnMode.Update	TrnMode.Delete
		<pre>&cityTour.Load(2) &cityTour.Insert() &cityTour.Update()</pre>	<pre>&cityTour.Delete()</pre>
	<pre>&cityTour = new()</pre>		

Temos utilizado repetidamente os métodos: Load, para carregar informações existentes em um BC, em combinação com os métodos Update() ou Delete(), para atualizar ou excluir, e o método Insert(), para inserir.

Mas podemos utilizar outras formas. Basta termos claras as diferenças e casos de uso.

Por exemplo, o método Save nos poupa de especificar a operação de que se trata. Se parece mais com o botão Confirm da transação. Vai depender do modo em que se encontra a variável, se tentará inserir ou atualizar: se encontra-se em modo Insert, tentará inserir e se estiver em Update, atualizar.

Por exemplo, toda variável BC definida em um objeto está em modo Insert e o mesmo ocorre quando lhe destina um novo espaço de memória com new().

Se a variável for carregada com Load() passa imediatamente a estar em modo Update.

Depois de realizada alguma operação sobre ela, se for bem-sucedida, estará em modo Update, se foi inserido ou modificado, e em modo Delete se foi aplicado o método Delete().

Então, para saber que operação tentou se realizar ao encontrar este Save, devemos conhecer o contexto desta variável. Em vez disso, com os métodos Insert e Update isso não é necessário. Independentemente do modo, vai querer inserir ou atualizar as informações na base de dados.

More about the differences between methods and...

Para conhecer um pouco mais as diferenças entre os diferentes métodos, incluindo como trabalhar com Business Components carregados a partir de Data Providers, te convidamos a ver o vídeo correspondente.

GeneXus™

training.genexus.com

wiki.genexus.com

training.genexus.com/certifications