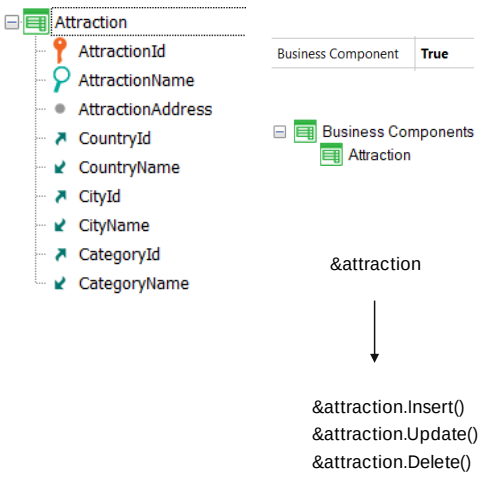
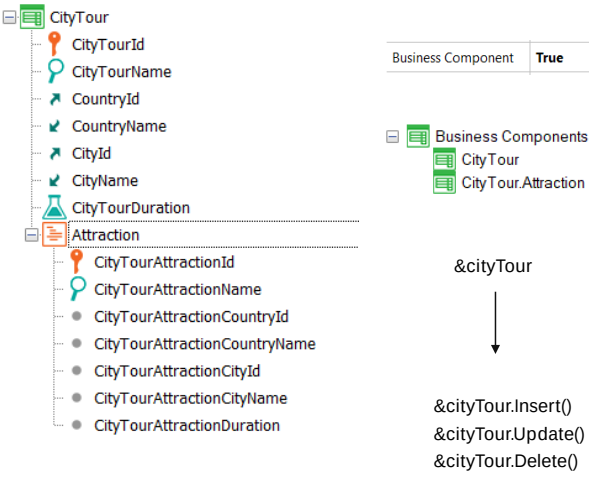
Atualização da base de dados

Comparação entre business component de um nível e de dois

GeneXus

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



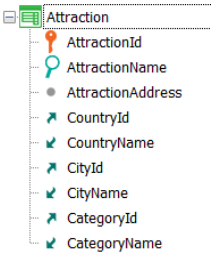


Havíamos visto como inserir, modificar e excluir informações da base de dados provenientes de transações de um e dois níveis através de business components.

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.

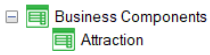
Single-level BC / Header of a Two-level BC



Insert

```
&attraction = new()
&attraction.AttractionName  = "Eiffel Tower"
&attraction.CountryId       = find( CountryId, CountryName = "France" )
&attraction.CityId          = find( CityId, CityName = "Paris" )
&attraction.CategoryId      = find( CategoryId, CategoryName = "Monument" )
&attraction.Insert()
   Commit
endif
```

Update



Delete

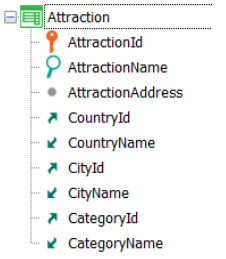
&attraction

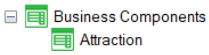
AttractionId	3
AttractionName	Eiffel Tower
CountryId	2
CountryName	France
CityId	1
CityName	Paris
CategoryId	2
CategoryName	Monument

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





Insert

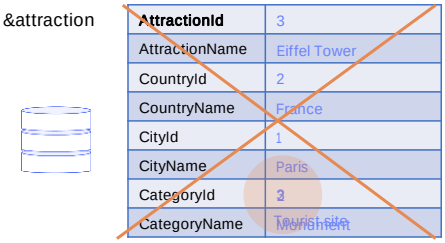
```
&attraction = new()
&attraction.AttractionName  = "Eiffel Tower"
&attraction.CountryId       = find( CountryId, CountryName = "France" )
&attraction.CityId          = find( CityId, CityName = "Paris" )
&attraction.CategoryId      = find( CategoryId, CategoryName = "Monument" )
&attraction.Insert()
   Commit
endif
```

Update

```
&attraction = Load(3)
&attraction.CategoryId  = find( CategoryId, CategoryName = "Tourist site" )

If &attraction.Update()
   Commit
endif
```

Delete

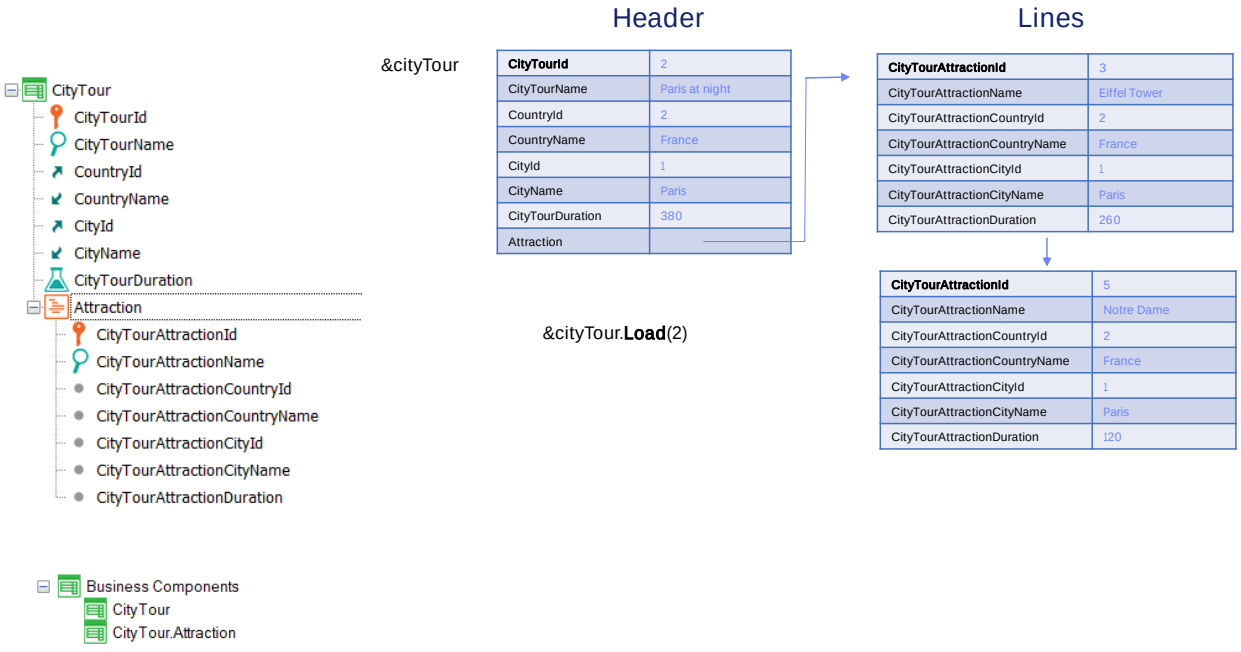
```
&attraction.Load(3)
&attraction.Delete()

If &attraction.Success()
  Commit
endif
```



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.

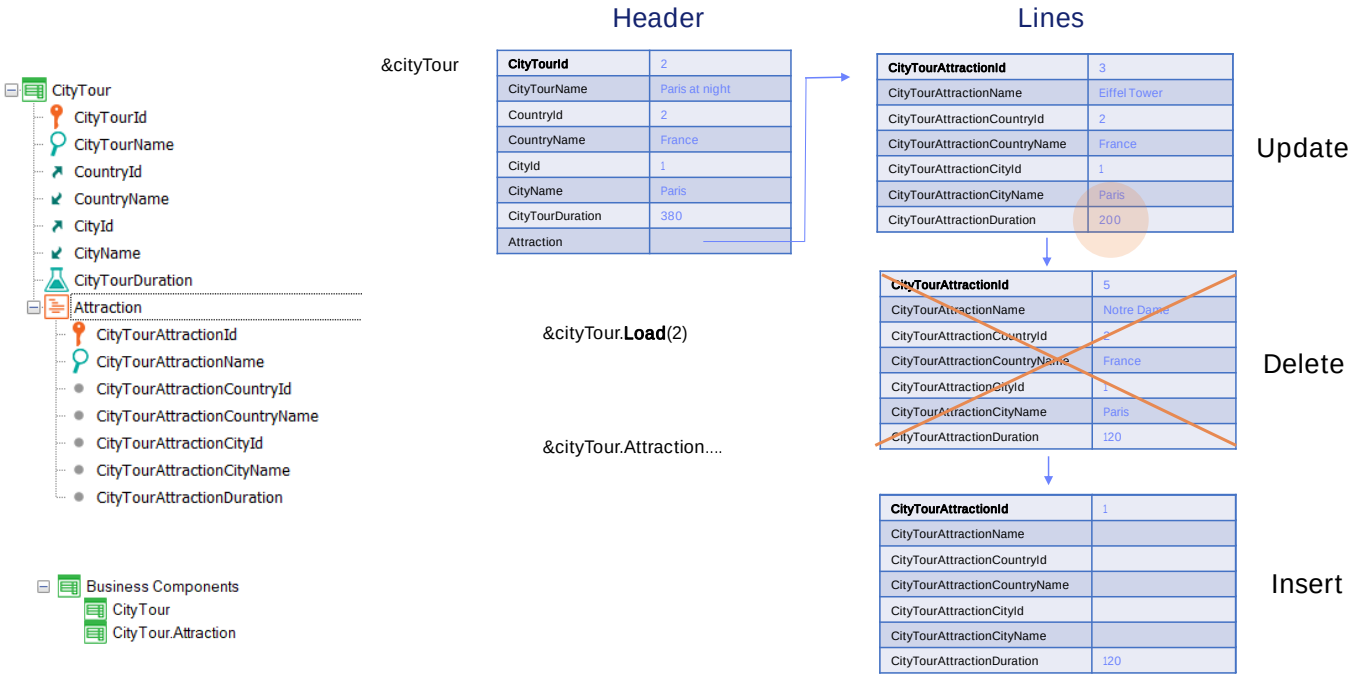
Por último, para excluir uma atração, a carregávamos na variável com o método Load e, então, invocávamos o método Delete, que era o que efetivamente a excluía da base de dados se tudo estivesse bem. A variável fica carregada com os valores que o registro tinha antes de ser excluído e em modo Delete.

Two-level BC



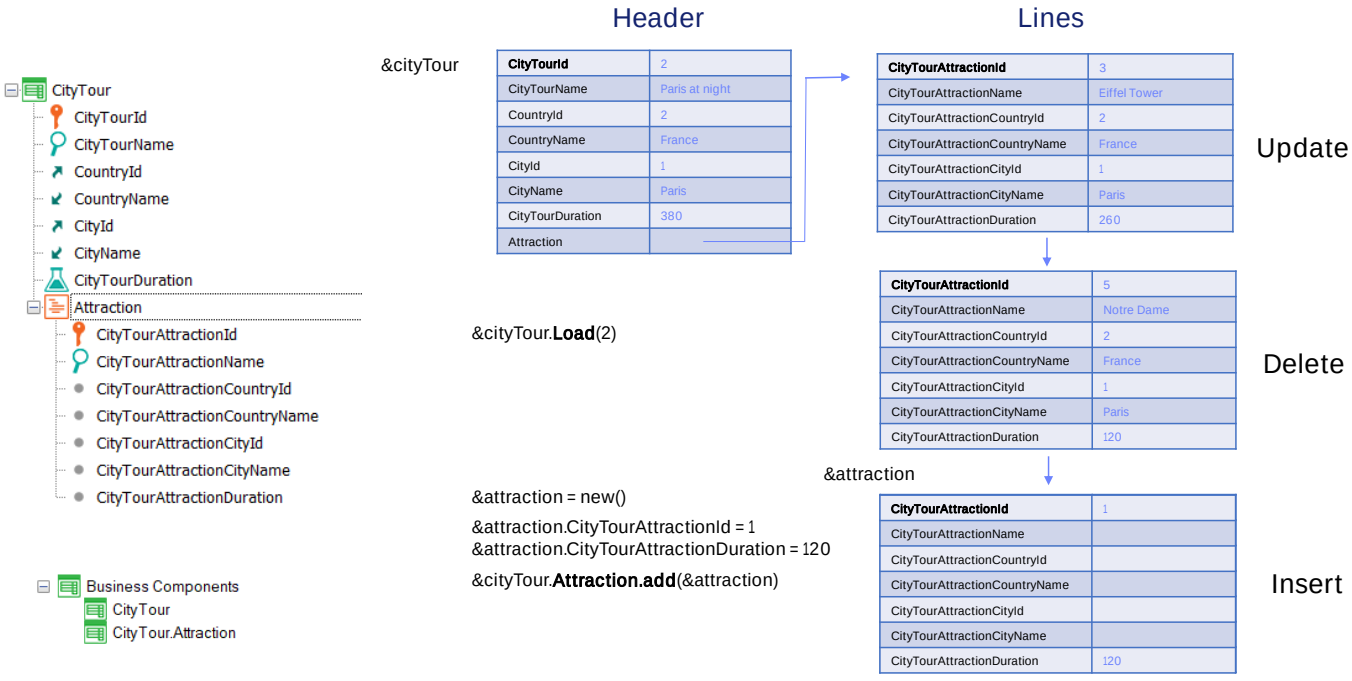
E para inserir, atualizar ou excluir linhas, se o cabeçalho existia, então, primeiro carregávamos na variável BC, cabeçalho e linhas e, logo, manipulávamos a coleção de linhas.

Two-level BC



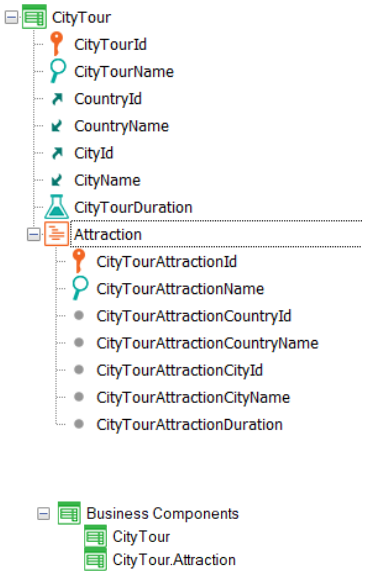
Por exemplo, se o city tour 2 tem duas linhas, uma para a Torre Eiffel e outra para a catedral de Notre Dame e quiséssemos adicionar uma nova, para o museu do Louvre, modificar a duração da visita à Torre Eiffel e excluir do tour a visita à Notre Dame... ou seja, inserir, modificar e excluir uma linha...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

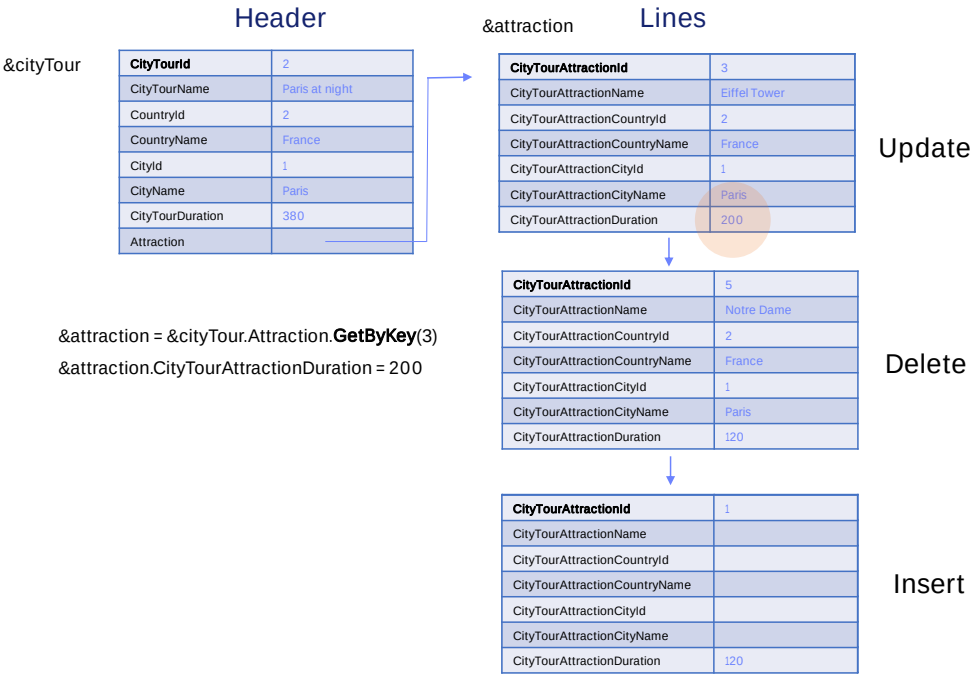
Two-level BC



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 &attraction.

Two-level BC



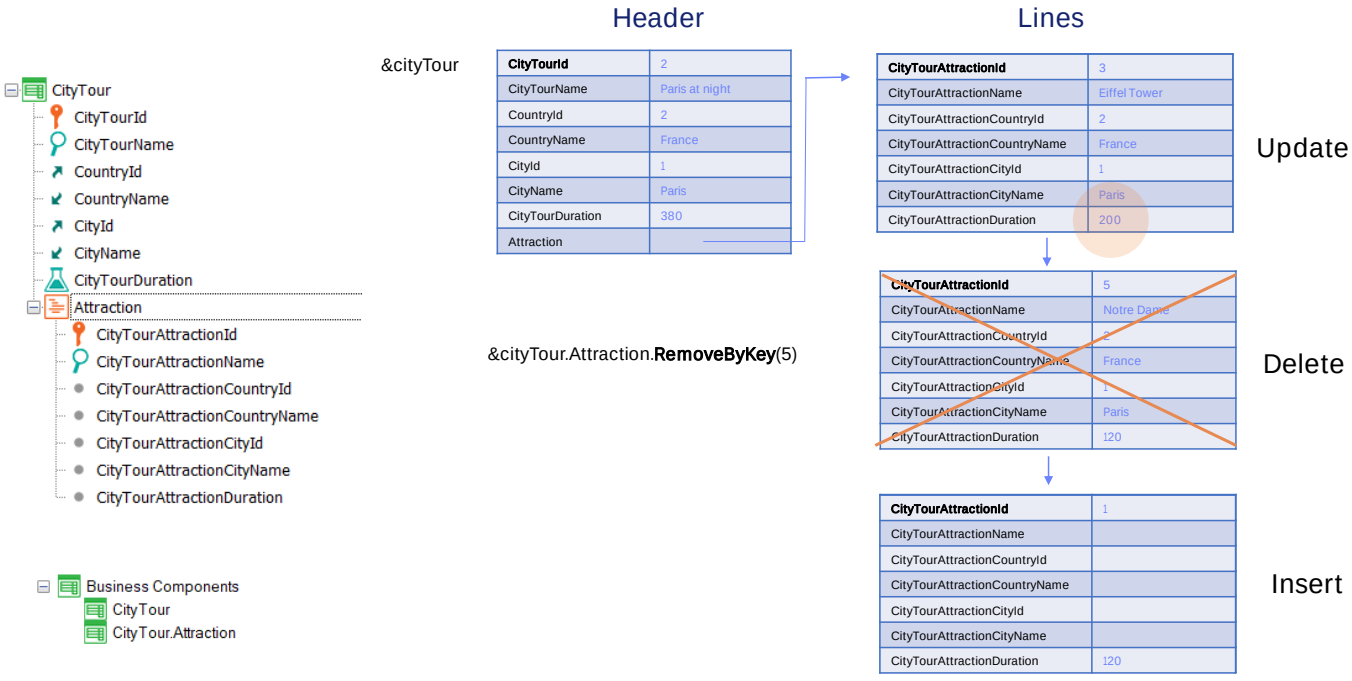


&attraction = &cityTour.Attraction.**GetByKey**(3)

&attraction.CityTourAttractionDuration = 200

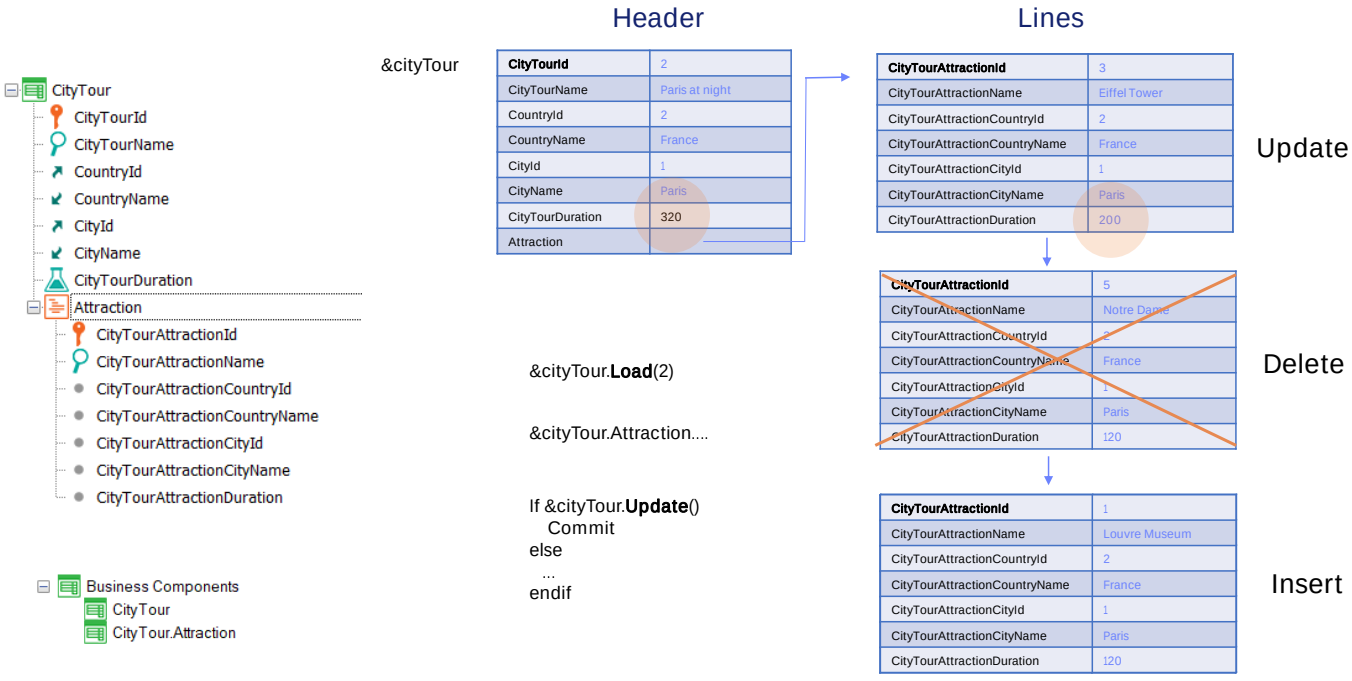
Para modificar uma linha, primeiro a obtemos na variável &attraction BC das linhas, com o método GetByKey. Em seguida, simplesmente atualizamos o elemento que nos interessa.

Two-level BC



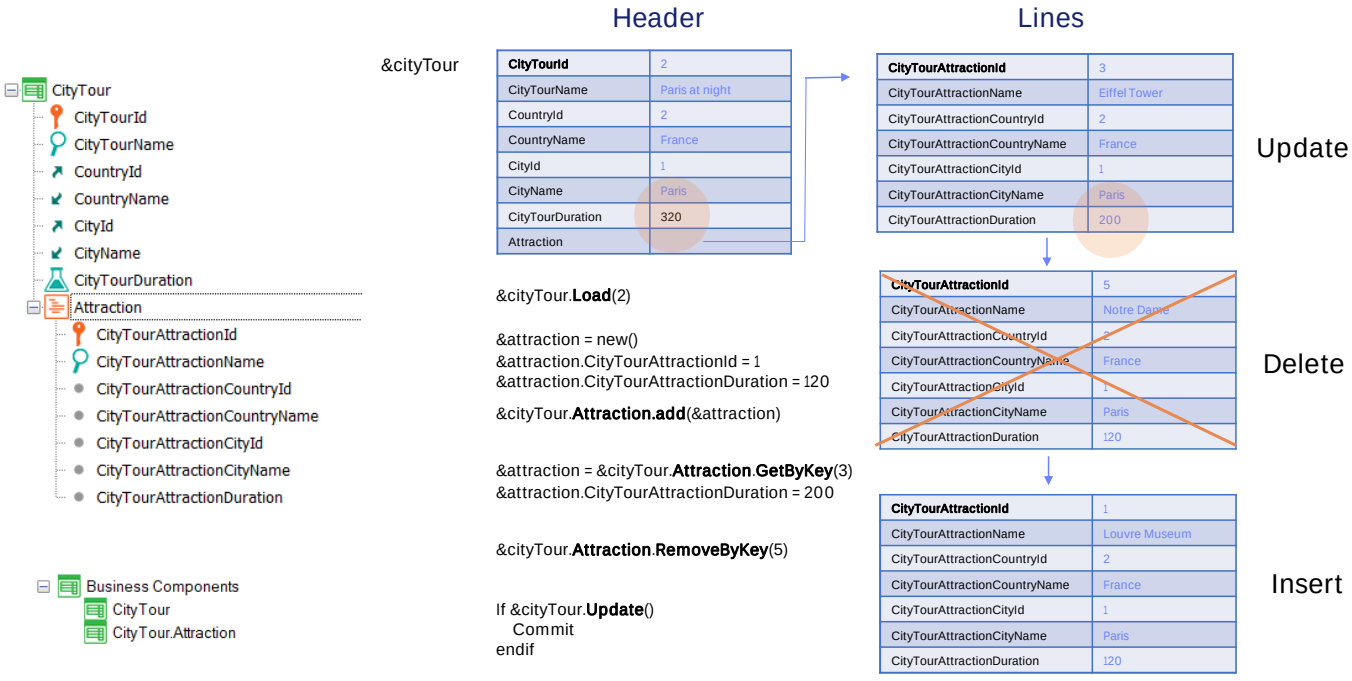
E, por último, para excluir uma linha, utilizamos o método RemoveByKey da coleção de linhas da variável &cityTour.

Two-level BC



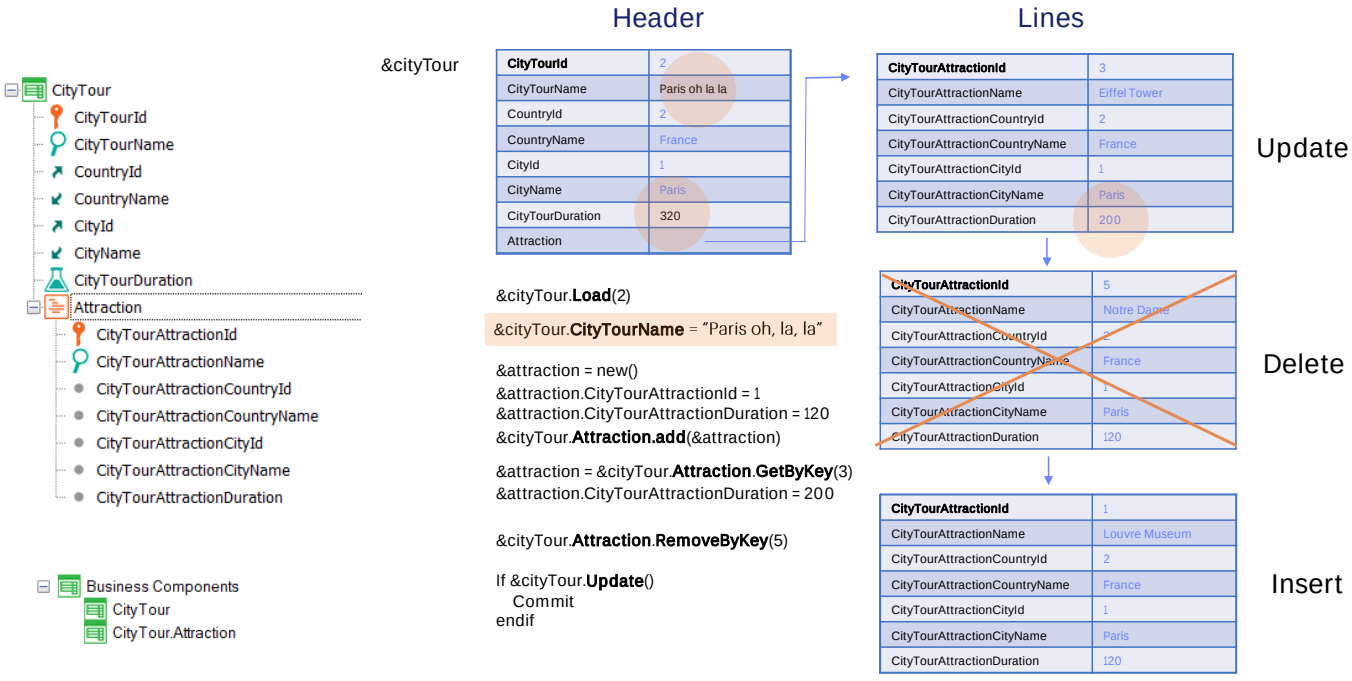
Depois de ter realizado as três operações sobre as linhas, invocamos o método Update, que é o que efetivamente impacta o realizado na variável na base de dados, executando as regras da transação, bem como os controles de integridade referencial.
Se for bem-sucedida, a variável fica carregada com os valores atuais, incluindo o caso de atributos inferidos e fórmulas. E fica, logicamente, em modo Update.

Two-level BC



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

Two-level BC



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

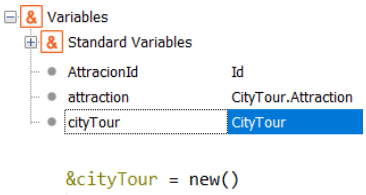
- Load(), Update(), Delete(), Insert()
- Save(), Asignation instead of Load, InsertOrUpdate()

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

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

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

TrnMode.Insert



TrnMode.Udpate

```
&cityTour.Load(2)
&cityTour.Insert()
&cityTour.Update()
```

TrnMode.Delete

```
&cityTour.Delete()
```

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.

GeneXus™

training.genexus.com
wiki.genexus.com