


Relações entre entidades da realidade

GeneXus™

Attraction Information

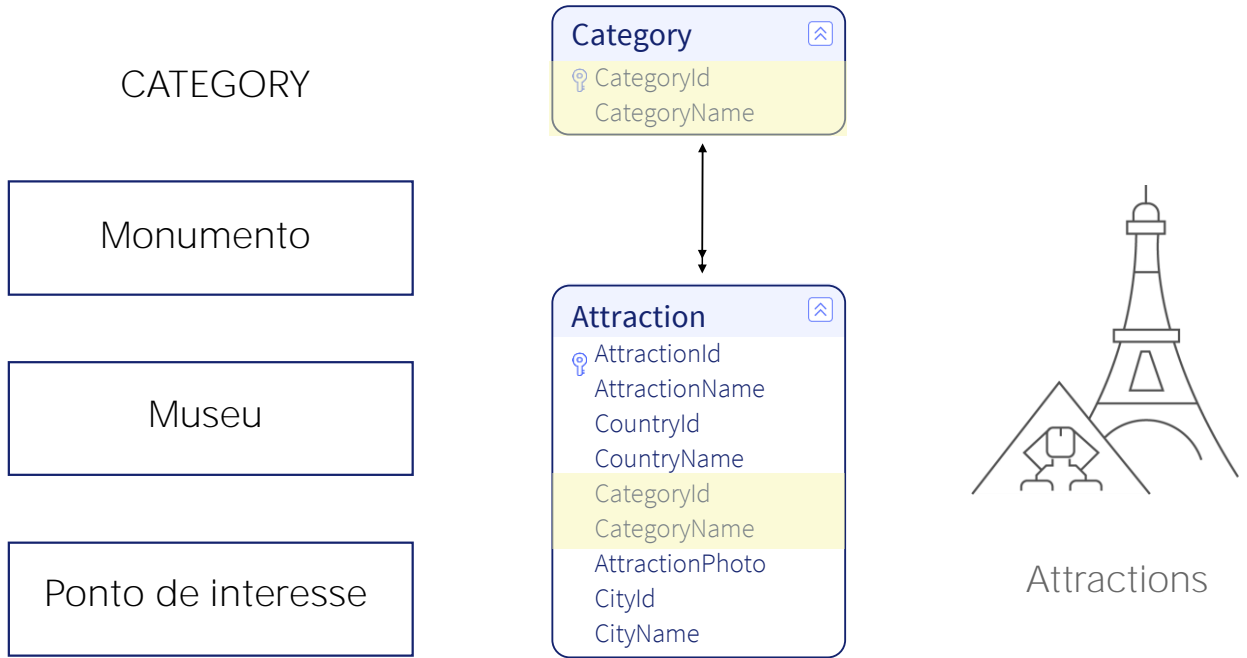
Name	Eiffel Tower
General	
Id	3
Name	Eiffel Tower
Country Id	2
Country Name	France
Category Id	1
Category Name	Museum



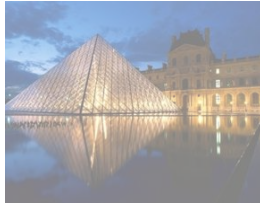
UPDATE DELETE



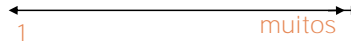
Em vários exemplos de nossa agência de viagens, encontramos que os atores da realidade se relacionam entre si de diferentes maneiras, por exemplo, quando uma atração pertence a uma categoria e que a sua vez essa categoria pode ser categoria de muitas atrações.



A forma que temos para representar essas relações é **quando projetamos (“desenhamos”) transações incluir os atributos de uma transação em outra.**



SUPPLIER



ATTRACTION

Em nossa agência foi informado agora que eles trabalham com fornecedores, que oferecem periodicamente visitas a atrações turísticas em diferentes partes do mundo.

Cada fornecedor fornece muitas atrações turísticas, mas cada atração é gerenciada por um único fornecedor.

Nova transação Supplier

The image shows two windows from the GeneXus IDE. The top window, titled 'Supplier', displays the structure of the object. It has a table with columns: Name, Type, Description, Formula, and Nullable. The bottom window, titled 'Diagram', shows two objects, 'Attraction' and 'Supplier', placed on a diagram canvas. The 'Attraction' object has attributes: AttractionId, AttractionName, CountryId, CountryName, CategoryId, CategoryName, AttractionPhoto, CityId, CityName, and AttractionAddress. The 'Supplier' object has attributes: SupplierId, SupplierName, and SupplierAddress.

Name	Type	Description	Formula	Nullable
Supplier	Supplier	Supplier		
SupplierId	Id	Supplier Id		No
SupplierName	Name	Supplier Name		No
SupplierAddress	Address, GeneXus	Supplier Address		No

Relação entre Attraction e Supplier?

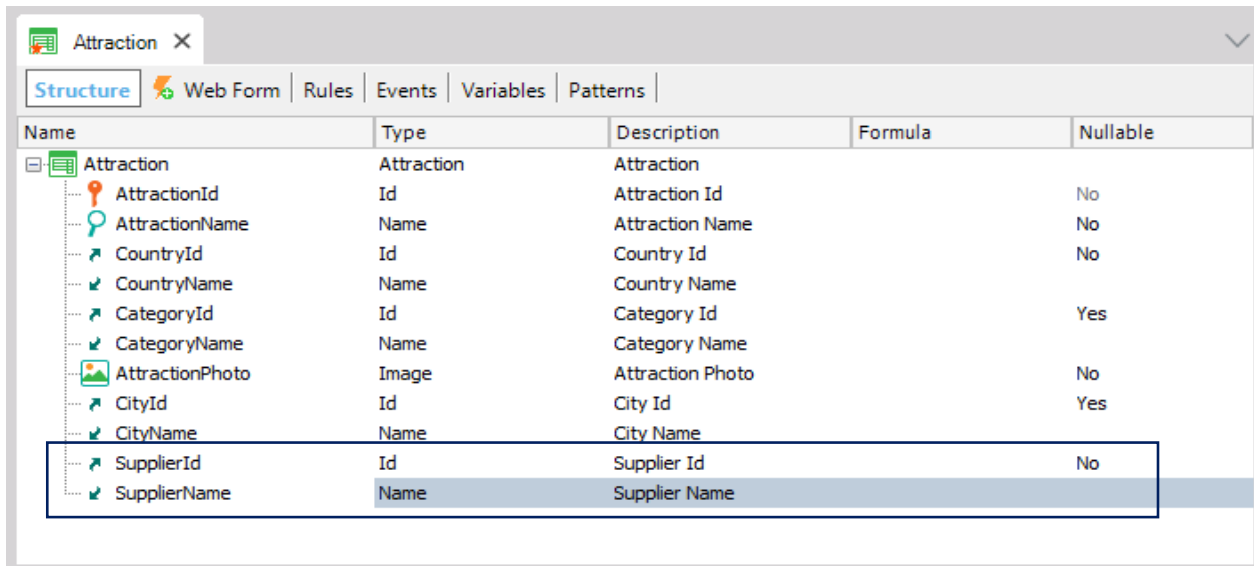
Para representar essa realidade, vamos criar a transação Supplier, na qual cadastraremos os fornecedores...

Acionamos File...New...Object...Colocamos o nome de Supplier... E colocamos os atributos:

SupplierId como chave-primária, SupplierName para armazenar o nome do fornecedor e SupplierAddress para armazenar seu endereço.

Observemos mediante o objeto diagrama de transações a relação entre os fornecedores e as atrações. Acionamos New Object, escolhemos o tipo Diagram e arrastamos as transações Attraction e Supplier até o diagrama. Visualizamos que não foi estabelecido ainda nenhuma relação entre esses dois atores. Salvamos.

Requisito: cada atração turística é oferecida por um único fornecedor.



Name	Type	Description	Formula	Nullable
Attraction	Attraction	Attraction		
AttractionId	Id	Attraction Id		No
AttractionName	Name	Attraction Name		No
CountryId	Id	Country Id		No
CountryName	Name	Country Name		
CategoryId	Id	Category Id		Yes
CategoryName	Name	Category Name		
AttractionPhoto	Image	Attraction Photo		No
CityId	Id	City Id		Yes
CityName	Name	City Name		
SupplierId	Id	Supplier Id		No
SupplierName	Name	Supplier Name		

Como uma atração turística tem um único fornecedor que a fornece, vamos incluir a chave-pimária do fornecedor na estrutura da transação Attraction.

Assim abrimos a transação e adicionamos o atributo SupplierId. Adicionamos também o atributo SupplierName, porque dessa forma podemos mostrar o nome do fornecedor na tela (Web Form) das atrações.

Outro olhar sobre a relação entre fornecedores e atrações

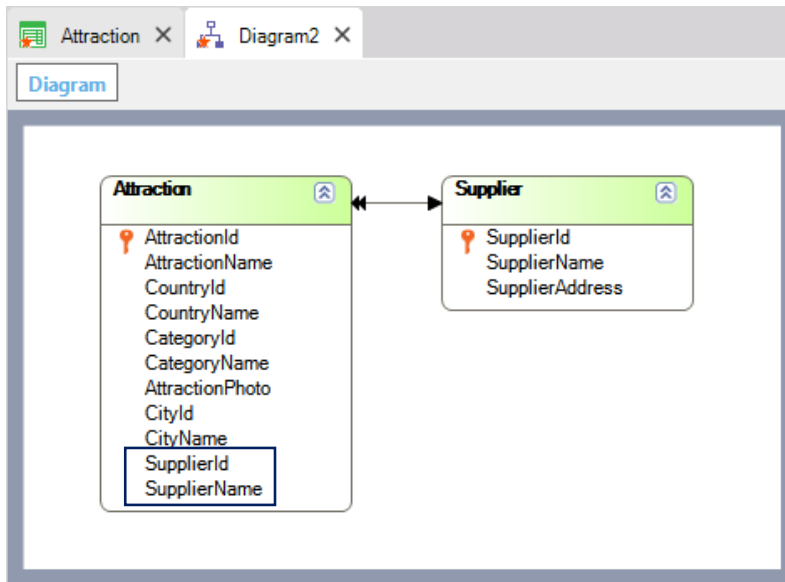


Diagrama de transação (não diagrama de tabela)

Voltamos a abrir o diagrama.

Visualizamos agora que existe uma flecha cuja ponta simples aponta para **Supplier** e cuja ponta dupla aponta para **Attraction**, indicando-nos que uma atração tem um único fornecedor e que um fornecedor pode oferecer muitas atrações.

Recapitulando, se adicionamos o atributo chave-primária de uma transação em outra transação, o qual como já vimos cumprirá o papel de chave-estrangeira, é estabelecido uma relação de 1 a muitos (também chamada de "1 a N").

Tabelas Criadas por GeneXus Baseado no Design Implementado

Name	Type	Description	Formula
Supplier Structure		Supplier	
SupplierId	Id	Supplier Id	
SupplierName	Name	Supplier Name	
SupplierAddress	Address, GeneXus	Supplier Address	

Name	Type	Description	Formula	Nullable
Supplier	Supplier	Supplier		
SupplierId	Id	Supplier Id		
SupplierName	Name	Supplier Name		
SupplierAddress	Address, GeneXus	Supplier Address		No

O lado “muitos” da relação é onde está a chave-estrangeira.

Se analisarmos agora quais seriam as tabelas que GeneXus gera a partir desse “desenho” de transações, visualizamos que a partir da transação Supplier será criada a tabela SUPPLIER com igual estrutura que a transação.

The image displays two screenshots from the GeneXus IDE. The top screenshot shows the 'Attraction Structure' window, which lists the fields and their types for the transaction. The bottom screenshot shows the 'Attraction' table structure, which lists the fields, their types, descriptions, formulas, and nullable status.

Attraction Structure (Transaction):

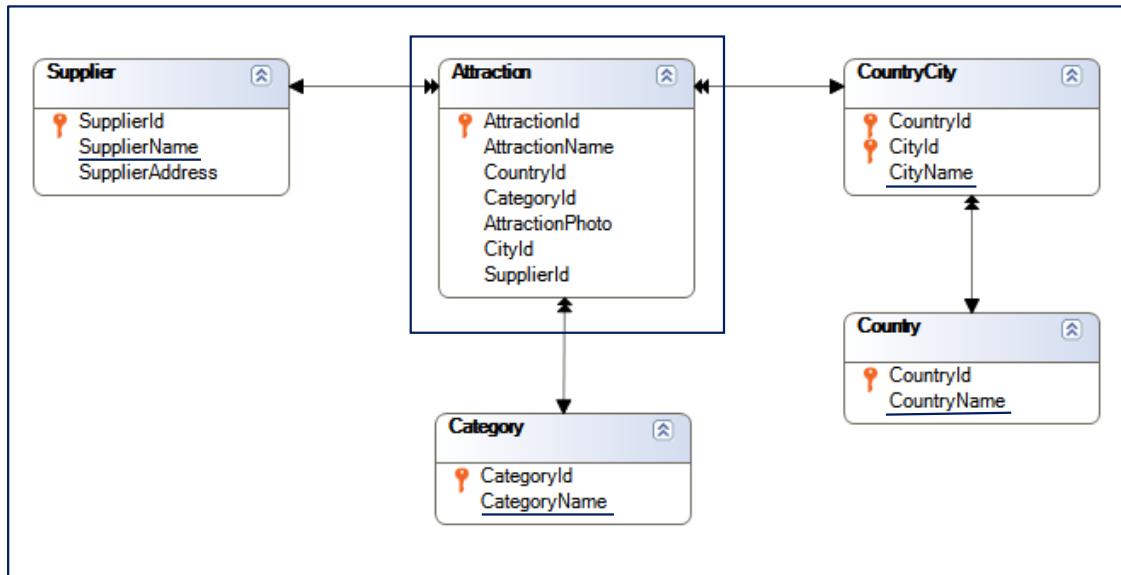
Name	Type	Description	Formula
Attraction Structure		Attraction	
AttractionId	Id	Attraction Id	
AttractionName	Name	Attraction Name	
CountryId	Id	Country Id	
CategoryId	Id	Category Id	
AttractionPhoto	Image	Attraction Photo	
CityId	Id	City Id	
SupplierId	Id	Supplier Id	

Attraction Table:

Name	Type	Description	Formula	Nullable
Attraction	Attraction	Attraction		
AttractionId	Id	Attraction Id		No
AttractionName	Name	Attraction Name		No
CountryId	Id	Country Id		No
CountryName	Name	Country Name		
CategoryId	Id	Category Id		
CategoryName	Name	Category Name		
AttractionPhoto	Image	Attraction Photo		No
CityId	Id	City Id		Yes
CityName	Name	City Name		
SupplierId	Id	Supplier Id		No
SupplierName	Name	Supplier Name		

E a partir da estrutura da transação Attraction, GeneXus cria a tabela ATTRACTION com a seguinte estrutura:

Se compararmos a estrutura da tabela ATTRACTION com a da transação Attraction, os atributos CountryName, CategoryName, CityName e SupplierName não são incluídos na tabela já que são atributos inferidos.



Como estudamos antes, como esses atributos estão na tabela estendida da tabela ATTRACTION, seu valor pode ser recuperado das tabelas onde estão fisicamente armazenados.

Essa é a forma mais comum de representar a relação de 1 a muitos entre dois atores da realidade, isto é, entre duas entidades do nosso sistema.

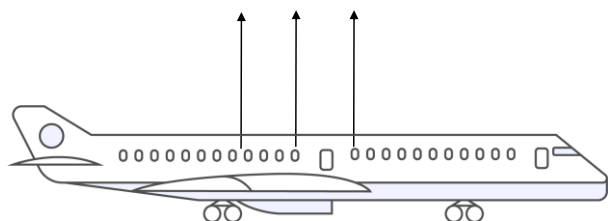


FLIGHT

1

muitos

SEAT



Flight

Porém, existem outros dois casos de relações 1 a muitos, nas quais usaremos outro tipo de representação.

Lembremos do caso dos voos, onde um voo tem muitos lugares e cada lugar está atribuído a um voo, significando uma relação de 1 a muitos.

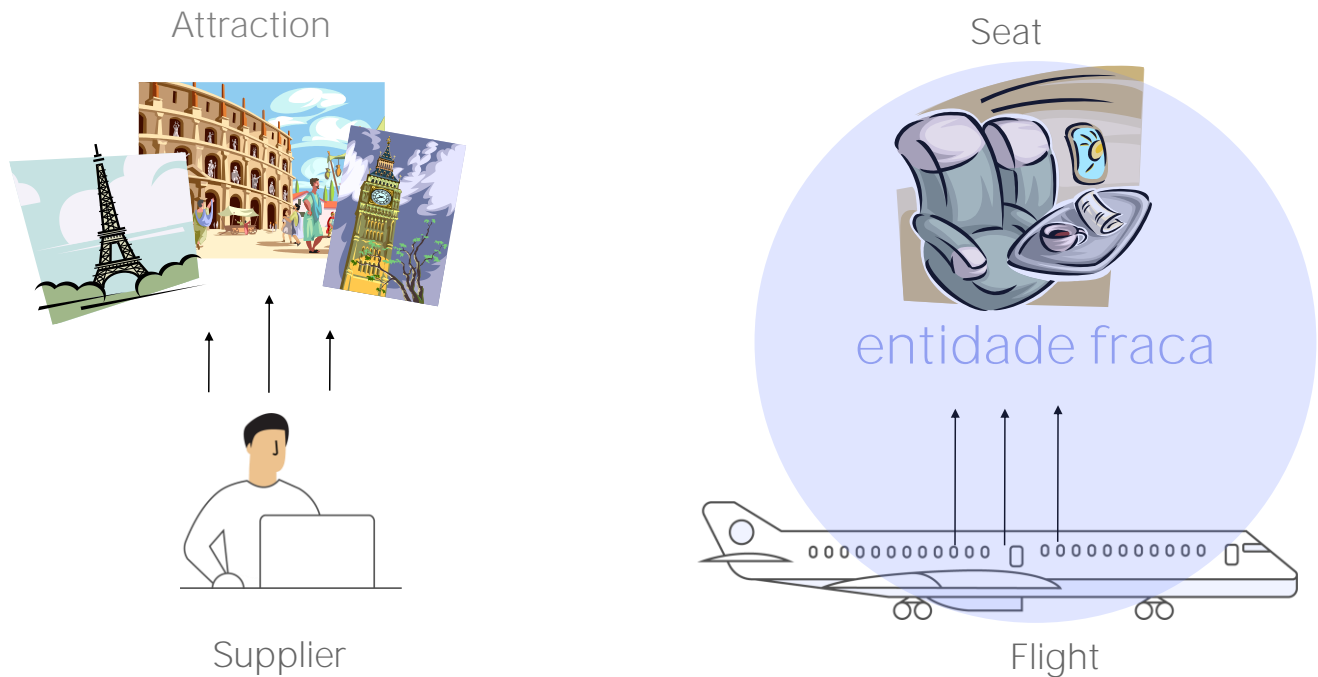
Outra maneira de modelar o relação 1 a N

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 Name		
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
AirlineId	Id	Airline Id		No
AirlineName	Name	Airline Name		
AirlineDiscountPercentage	Percentage	Airline Discount Percentage		
FlightFinalPrice	Price	Flight Final Price	FlightPrice*(1-AirlineDiscountPer...	
FlightCapacity	Numeric(4,0)	Flight Capacity	count(FlightSeat.Location)	
Seat	Seat	Seat		
FlightSeatId	Id	Flight Seat Id		No
FlightSeatChar	SeatChar	Flight Seat Char		No
FlightSeatLocation	Location	Flight Seat Location		No

Vamos abrir a estrutura da transação Flight para ver como representamos essa relação...

Nesse caso o lugar (Seat) está como um segundo nível na transação Flight.

Então, em que se diferencia essa relação de 1 a muitos com a relação de 1 a muitos entre as Atrações e seus Fornecedores?

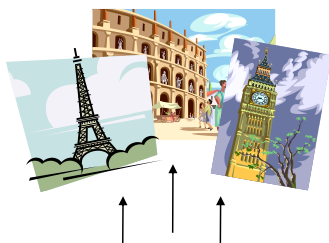
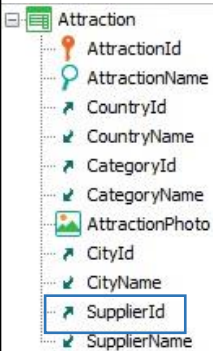
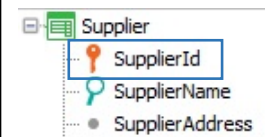


Por que não representamos da mesma maneira (com o mesmo “desenho” de transações) em ambos os casos?

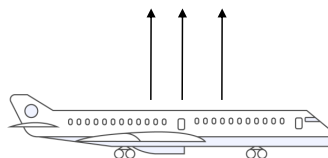
Observemos que os lugares não tem sentido de existir se não estiverem em um voo, ou seja, não tem sentido considerar um lugar sem relacionar **sempre** com o voo ao qual pertence... Por outro lado, uma atração poderia não ter um fornecedor que a forneça e, porém, existir por si mesma como tal...

A outra diferença é que quando cadastramos os dados de um voo já cadastramos também os dados de seus lugares (da mesma forma quando lançamos um pedido com seus itens, lançamos toda a informação junta, de uma só vez). Em contra-partida, os dados de Fornecedores e Atrações não faz sentido cadastrar todos no mesmo momento.

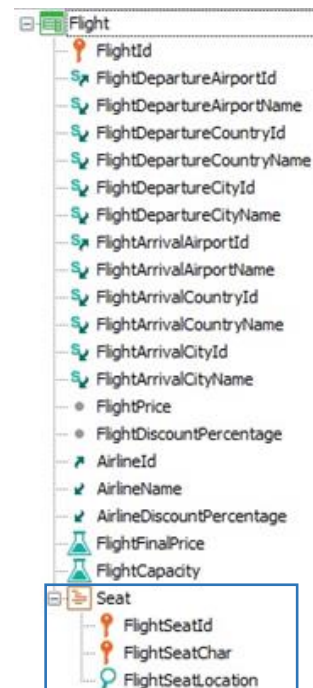
Uma entidade como os lugares, que somente tem sentido de existir se representam-se em função de outra entidade (neste caso os voos), dizemos que é um **entidade fraca**.



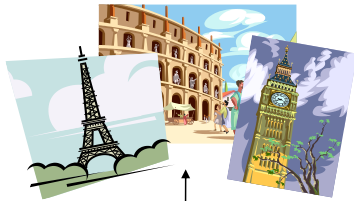
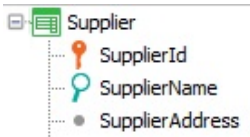
1 a muitos



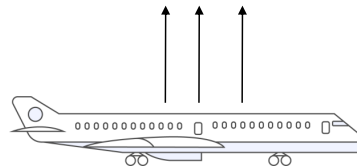
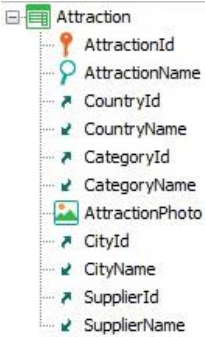
1 a muitos (fraca)



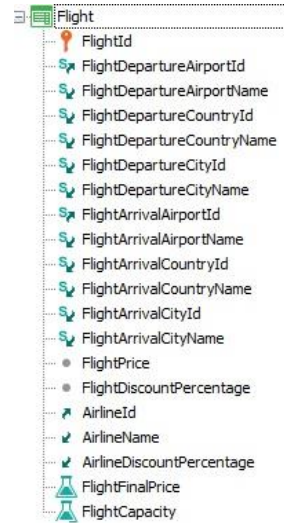
Esse tipo de relação **1 a N fraca** podemos representar com uma única transação de dois níveis, onde a entidade fraca está no segundo nível. A diferença da relação 1 a N de Fornecedores e Atrações, é que criamos **duas** transações, e em uma colocamos como chave-estrangeira a chave-primária da outra.



1 a muitos



1 a muitos (fraca)

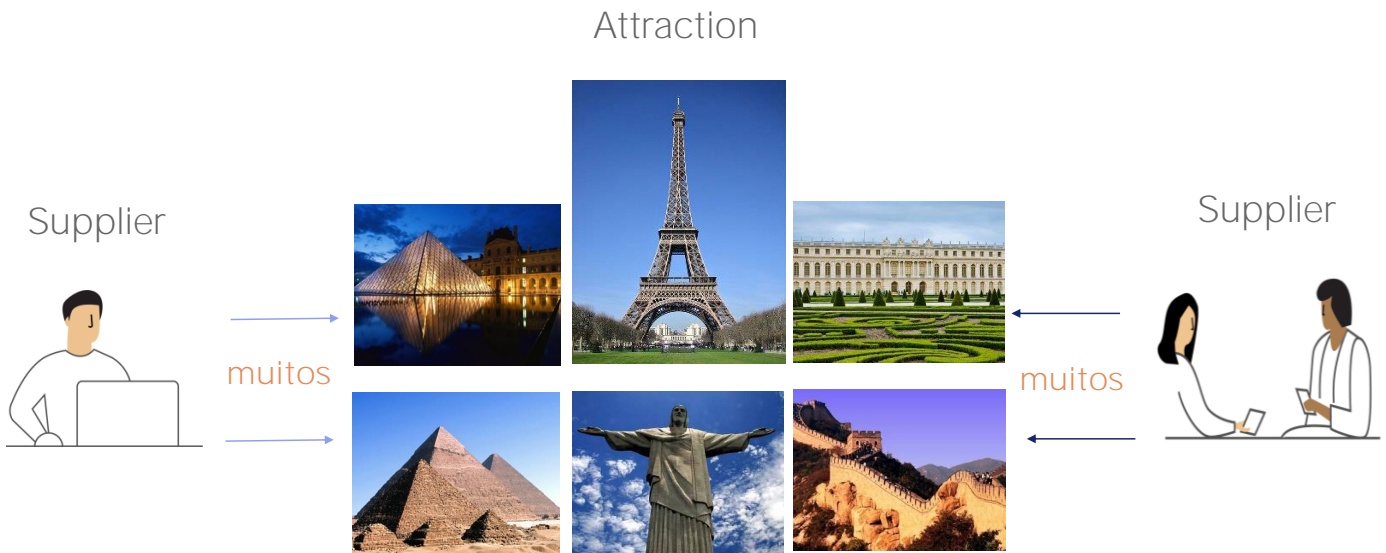


Esse tipo de relação **1 a N fraca** podemos representar com uma única transação de dois níveis, onde a entidade fraca está no segundo nível. A diferença da relação 1 a N de Fornecedores e Atrações, é que criamos **duas** transações, e em uma colocamos como chave-estrangeira a chave-primária da outra.

Aí fica estabelecido a diferença entre entidade forte e fraca. Observemos que por ser FlightId parte da chave-primária, é impossível definir um lugar de voo, por exemplo o 2 A Janela, sem inserir um valor para o voo, FlightId. Em contra-partida, é perfeitamente possível cadastrar aqui uma atração e não especificar seu fornecedor, se esse atributo estiver com o valor Yes na sua propriedade Nullable.

Muito bem, até agora aprendemos relações de 1 a muitos, mas nem sempre este é o caso da realidade que devemos representar.

Relação “muitos” a “muitos” (N a N)

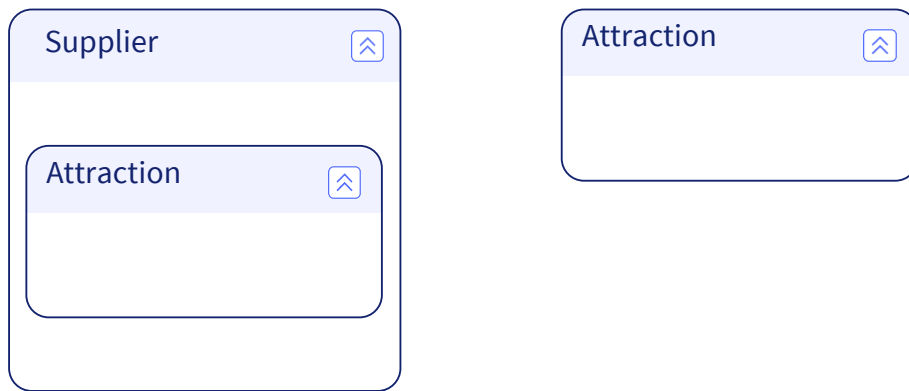


Suponhamos, por exemplo, que a agência de viagens diga que **sua realidade foi alterada**.

Cada fornecedor fornece muitas atrações turísticas (como está agora), *mas cada atração pode ser gerenciada por VÁRIOS fornecedores (e não por um apenas, como está agora)*.

Ou seja, que a relação entre Fornecedores e Atrações já não é de “1 a muitos” senão “muitos a muitos”.
Como representamos isso no GeneXus?

Como representar uma relação N a N



A solução é utilizando duas transações, uma para cada entidade. Também, a uma delas adicionamos como segundo nível da outra. Fazemos isso levando em conta como é que será cadastrado os dados, se para cada fornecedor com todas as suas atrações turísticas ...

Como representar uma relação N a N



...ou para cada atração todos os fornecedores que a fornece.

Modelando a relação N a N em GeneXus

The image displays two screenshots from the GeneXus IDE, showing the structure of two transactions: 'Attraction' and 'Supplier'.

Attraction Transaction Structure:

Name	Type	Description	Formula	Nullable
Attraction	Attraction	Attraction		
AttractionId	Id	Attraction Id		No
AttractionName	Name	Attraction Name		No
CountryId	Id	Country Id		No
CountryName	Name	Country Name		
CategoryId	Id	Category Id		Yes
CategoryName	Name	Category Name		
AttractionPhoto	Image	Attraction Photo		No
CityId	Id	City Id		Yes
CityName	Name	City Name		

Supplier Transaction Structure:

Name	Type	Description	Formula	Nullable
Supplier	Supplier	Supplier		
SupplierId	Id	Supplier Id		No
SupplierName	Name	Supplier Name		No
SupplierAddress	Address, GeneXus	Supplier Address		No
Attraction	Attraction	Attraction		
AttractionId	Id	Attraction Id		No
AttractionName	Name	Attraction Name		
AttractionPhoto	Image	Attraction Photo		

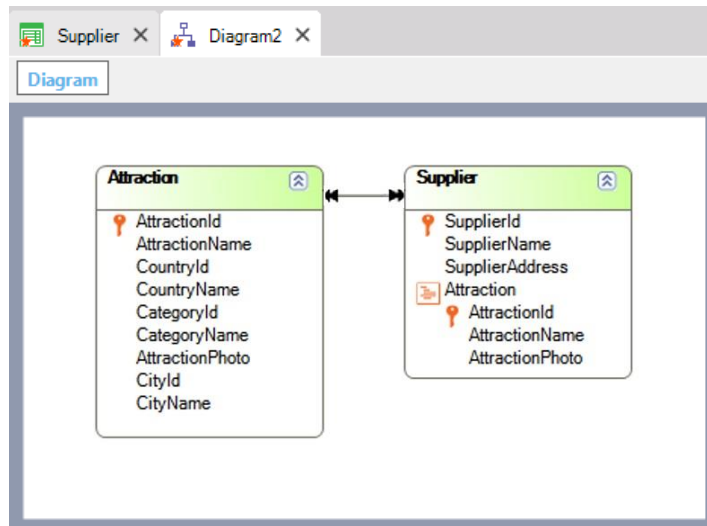
Em nosso caso a agência solicita que seja quando o usuário for cadastrar os dados de um fornecedor, cadastre também todas as atrações que ele fornece.

Vamos fazer isso no GeneXus...

Abrimos a transação Attraction e retiramos os atributos SupplierId e SupplierName e salvamos.

Agora abrimos a transação Supplier, onde adicionamos um segundo nível e adicionamos os atributos: AttractionId, (observemos que ao escrever como atributo chave um que começa com "Attraction", automaticamente foi alterado o nome do nível para Attraction), adicionamos também AttractionName e AttractionPhoto.

Um olhar sobre a relação estabelecida entre fornecedores e atrações



Vamos ver como ficou essa relação abrindo o diagrama que tínhamos entre a transação Attraction e Supplier.

Agora existe um dupla flecha em cada extremo da relação, o que indica que a relação é de “muitos” a “muitos”, ou seja, que uma atração é fornecida por muitos fornecedores e que um fornecedor fornece muitas atrações.

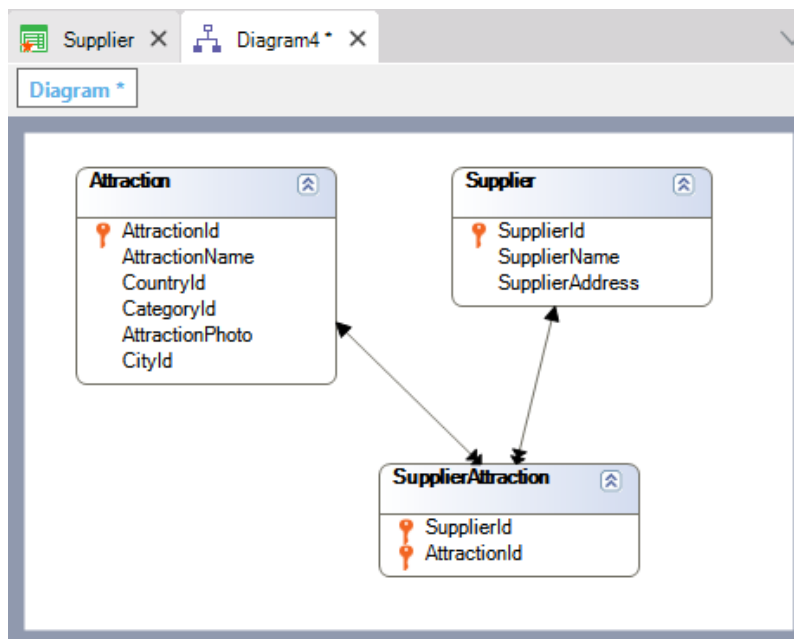
Tabelas criadas por GeneXus baseadas no desenho Implementado



Vejamos agora as tabelas que GeneXus cria a partir do “desenho” anterior...

Encontramos a tabela ATTRACTION, a tabela SUPPLIER e uma tabela chamada SUPPLIERATTRACTION.

Relação entre tabelas: N a N



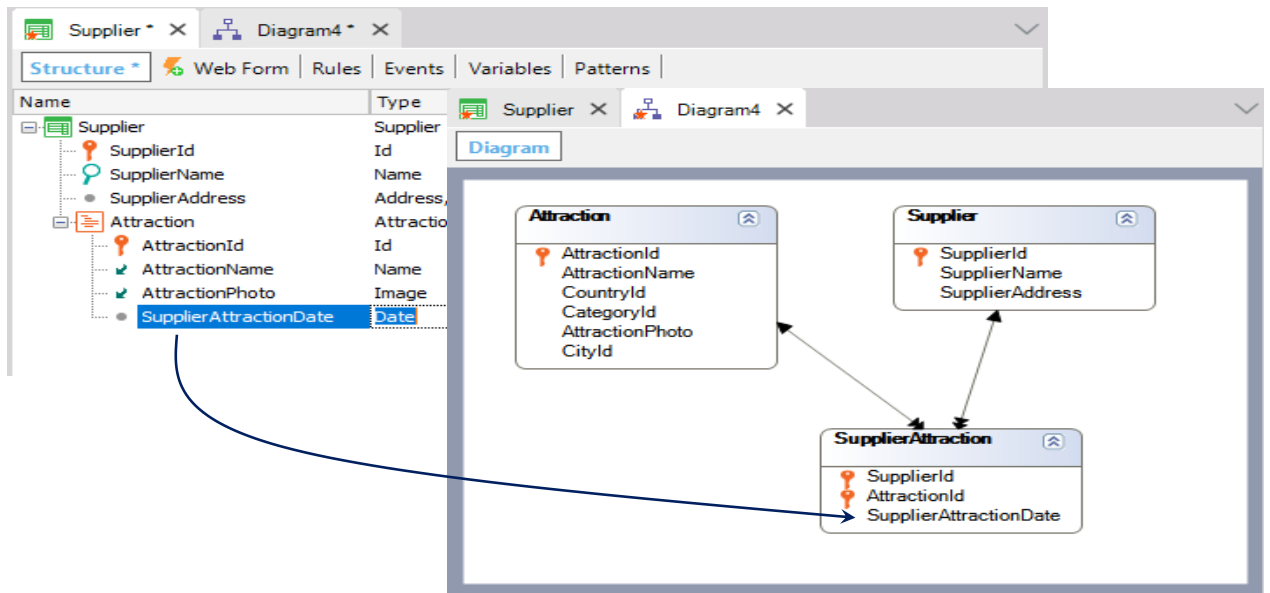
Criemos um objeto diagrama novo e arrastemos as três tabelas ao diagrama...

Observemos que nesse caso, GeneXus cria uma tabela para cada transação que intervém na relação muitos a muitos (ATTRACTION e SUPPLIER), mas além disso, cria uma terceira tabela chamada SUPPLIERATTRACTION, para estabelecer a relação.

Se observarmos a estrutura dessa terceira tabela, visualizamos que somente são incluídos os atributos identificadores das outras duas tabelas.

Portanto, cada vez que GeneXus estabelece uma relação de muitos a muitos, essa relação será representada na base de dados por três tabelas, uma por cada entidade interveniente e uma terceira com os identificadores de ambas. Essa terceira tabela poderá ter, eventualmente, atributos próprios, como por exemplo, a data em que esse fornecedor começou a fornecer essa atração. Vamos até a estrutura de Supplier e adicionamos no segundo nível (usando o ponto para que apareça automaticamente o prefixo) o atributo `SupplierAttractionDate`.

Relação entre tabelas: N a N



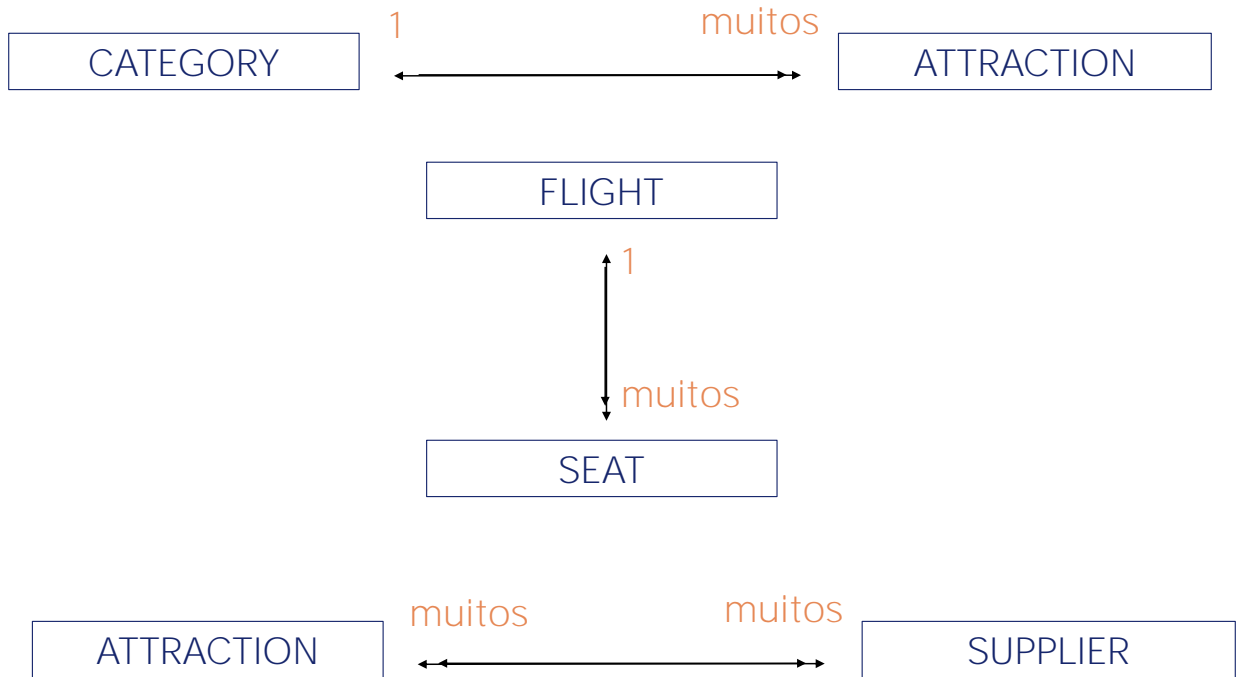
Se voltarmos a abrir o diagrama...

Visualizamos o atributo na tabela de relacionamento.

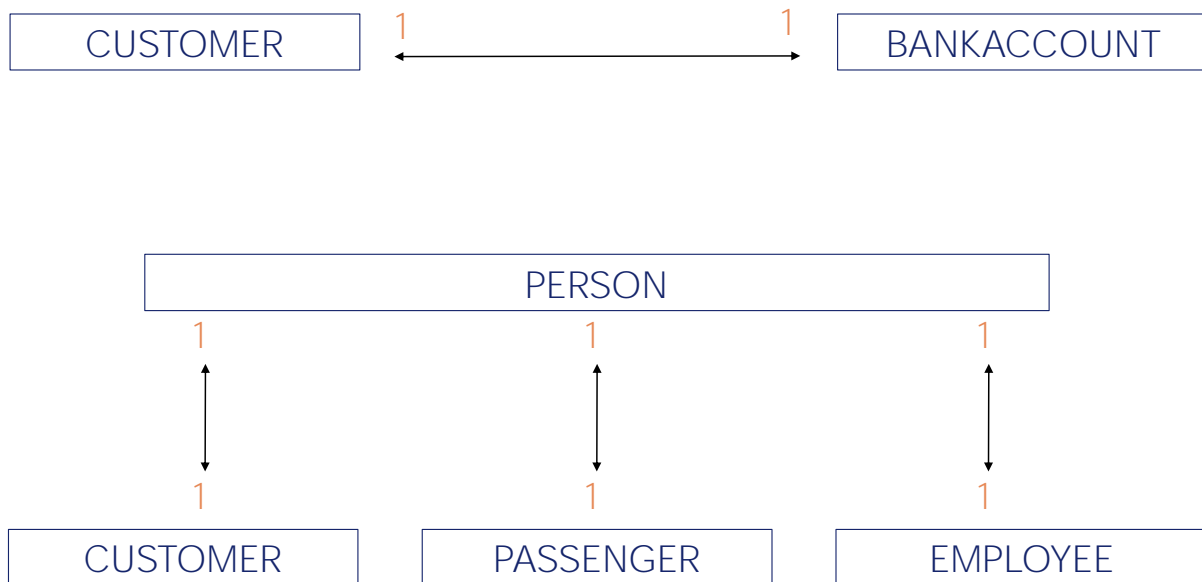
A relação de muitos a muitos entre **Attraction** e **Supplier** foi quebrada em 2 relações de um a muitos, utilizando a tabelas **SUPPLIERATTRACTION** para estabelecer a relação entre as anteriores.

Para finalizar, atualizamos a nossa KB no GeneXus Server...

E reorganizamos para que as tabelas sejam criadas...



Aprendemos assim que mediante as transações e seus atributos podemos representar diferentes relações entre os atores da nossa realidade.



Por exemplo, quando a agência de viagens necessita associar a cada cliente a conta bancária que é aberta com o objetivo de realizar os pagamentos dos serviços contratados.

Outro tipo de cenário de relações 1 a 1, mencionamos quando estudamos os subtipos. Era o caso da especialização: quando uma entidade é um caso particular de outra.

Passemos agora ao nosso próximo tema.

*GeneXus*TM

training.genexus.com
wiki.genexus.com