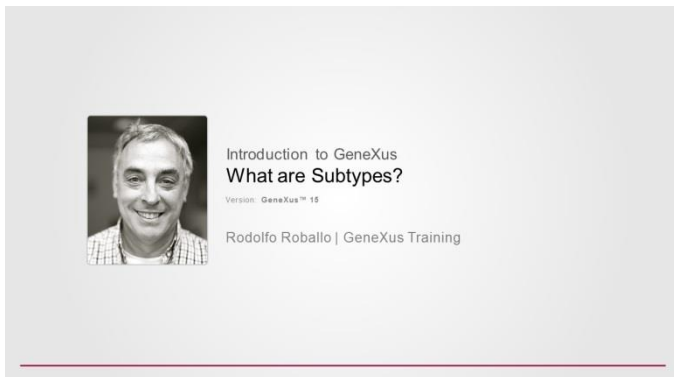


O que são os subtipos?



Até agora vimos que GeneXus estabelece relações entre transações -e entre tabelas- baseando-se nos nomes dos atributos iguais que encontra.

Por exemplo, na transação Attraction se encontra o atributo CountryId

Name	Type	Description	Formula	Nullable
Country	Country			
CountryId	Id	Country Id		No
CountryName	Name	Country Name		No
City	City	City		
CityId	Id	City Id		No
CityName	Name	City Name		No

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		

com o papel chave estrangeira, visto que com o mesmo nome está presente na transação Country, e ali não é chave primária

Por sua parte, o atributo CountryName também é encontrado em ambas as transações com o mesmo nome:

Name	Type	Description	Formula	Nullable
Country	Country	Country		
CountryId	Id	Country Id		No
CountryName	Name	Country Name		No
City	City	City		
CityId	Id	City Id		No
CityName	Name	City Name		No

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		No
CategoryId	Id	Category Id		Yes
CategoryName	Name	Category Name		No
AttractionPhoto	Image	Attraction Photo		No
CityId	Id	City Id		Yes
CityName	Name	City Name		No

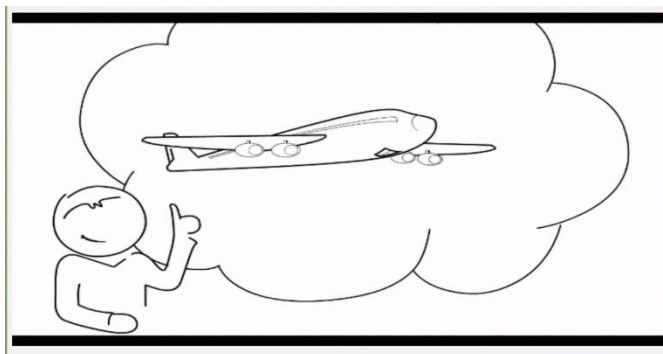
então GeneXus interpreta que se trata do mesmo atributo. Neste caso não é um atributo primário, ou seja, chave primária ou parte de uma chave primária em uma tabela, portanto, GeneXus determinará que ele deva ser armazenado na tabela COUNTRY e não na tabela ATRACTION.

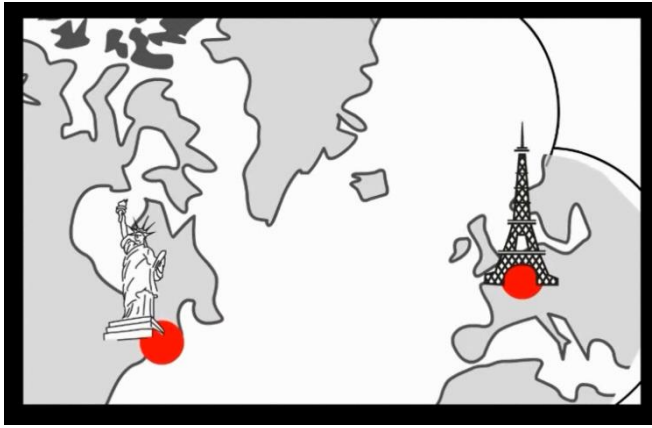
Então **GeneXus sempre pressupõe que se usarmos o mesmo nome de atributo, estamos representando o mesmo conceito.**

No entanto, há casos em que talvez precisemos usar nomes diferentes para o mesmo conceito, e dizer a GeneXus que ambos os nomes **significam o mesmo**.

Vamos ver isto.

Suponhamos que na agência de viagens nos pedem para registrar os voos que oferecem aos clientes para chegar a uma atração turística.



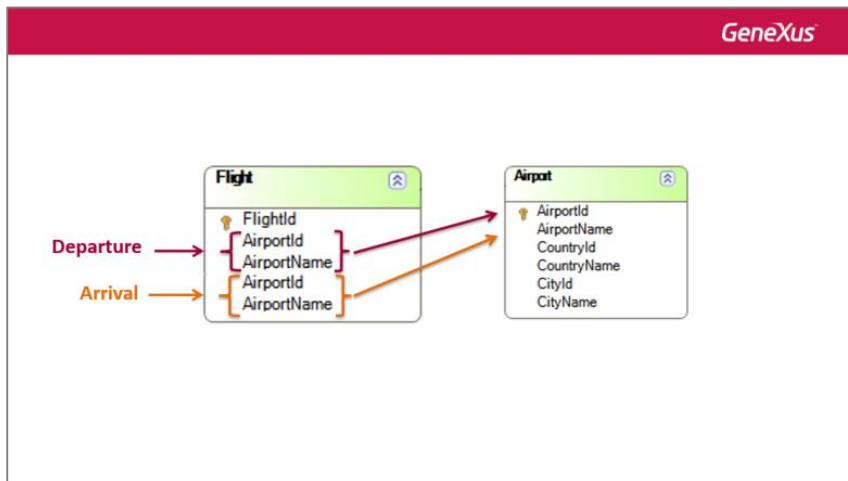


E devemos registrar para cada voo, o aeroporto de onde parte, bem como o aeroporto de chegada.

Para representar isso, criaremos em primeiro lugar uma transação de nome: Flight

Definimos o atributo FlightId, que é automaticamente baseado no domínio: Id.

E agora vamos pensar sobre que **outras informações devemos registrar**.

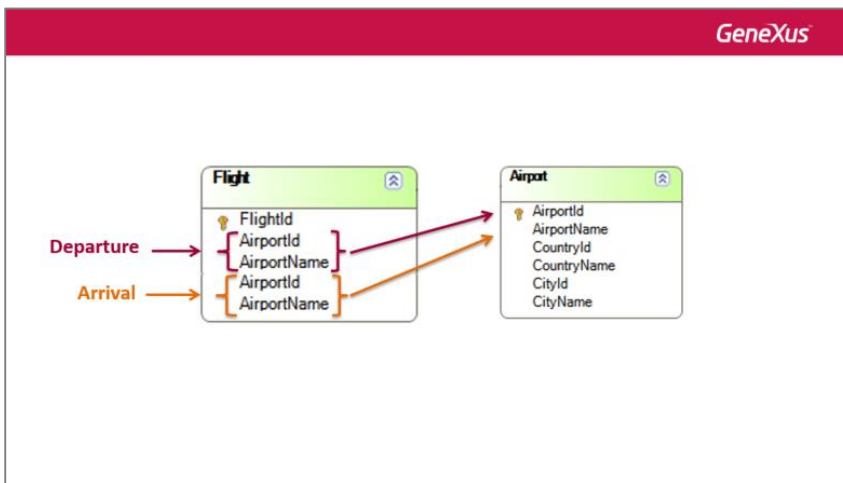


Cada voo, como dissemos, terá um aeroporto de partida e um aeroporto de chegada...

Mas para os aeroportos, teremos que ser capazes de registrá-los por si mesmos...., então podemos referenciá-los a partir dos voos.

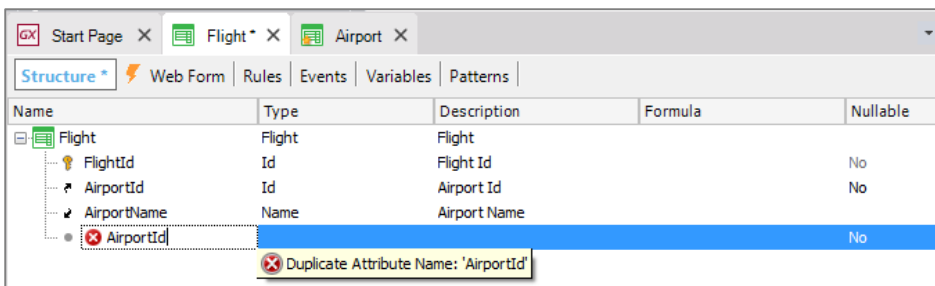
Então vamos parar de trabalhar na transação Flight por um instante, e vamos criar outra transação de nome Airport. Definimos então que cada aeroporto tem um identificador AirportId, um nome AirportName e cada aeroporto se encontra em 1 país e em 1 cidade, assim adicionaremos os atributos: CountryId, CountryName, CityId e CityName. Salvamos...

E agora voltemos a ver qual era nossa necessidade na transação Flight.



Precisamos adicionar a cada voo, seu aeroporto de partida e seu aeroporto de chegada.

Então voltamos para a transação Flight, e vamos adicionar os atributos AirportId, e AirportName.... Mas quando tentamos adicionar novamente AirportId:



GeneXus nos diz que há um erro!, que estamos adicionando um atributo com nome duplicado.

E o mesmo vai acontecer com o atributo AirportName, que pensávamos adicionar para representar o nome do aeroporto de chegada.

Como podemos então incluir 2 aeroportos na mesma transação? Obviamente, teremos que usar **nomes de atributo diferentes** para armazenar as informações de origem e destino do voo que queremos registrar.

Então, eliminaremos os atributos que originalmente havíamos incluído e definiremos atributos com novos nomes.

Chamaremos de FlightDepartureAirportId ao identificador do aeroporto de origem do voo, e FlightDepartureAirportName ao nome do aeroporto de origem.

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 ...		No

Bem, definimos novos nomes de atributo... mas para GeneXus esses nomes de atributo não estão relacionados a AirportId ou AirportName.

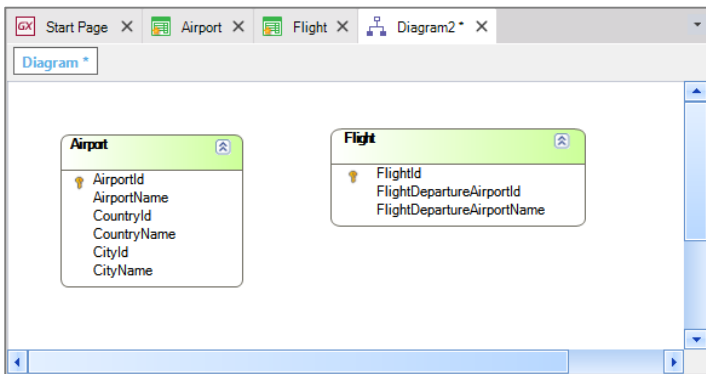
Como dissemos antes, se usamos nomes diferentes na transação Flight e na transação Airport para identificar o conceito de aeroporto, GeneXus não estabelecerá nenhuma relação entre as duas transações.

Name	Type	Description	Formula	Nullable
Airport	Airport	Airport		
AirportId	Id	Airport Id		No
AirportName	Name	Airport Name		No
CountryId	Id	Country Id		No
CountryName	Name	Country Name		
CityId	Id	City Id		No
CityName	Name	City Name		

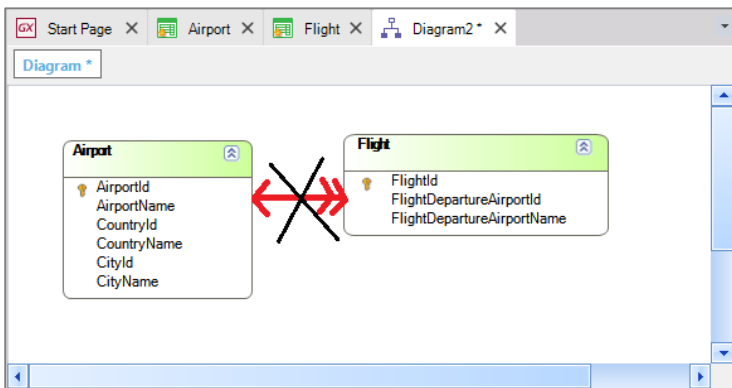
Name	Type	Description	Formula	Nullable
Flight	Flight	Flight		
FlightId	Id	Flight Id		No
FlightDepartureAirportId	Id	Flight Departure Airport Id		No
FlightDepartureAirportName	Name	Flight Departure Airport ...		No

Para verificar isso acabamos de dizer, vamos criar um diagrama de transações.

E vamos arrastar as transações Airport e Flight e vemos que, de fato, GeneXus não encontra nenhuma relação entre elas, uma vez que nenhuma chave estrangeira foi identificada em Flight que permita a relação com o Airport.



Se tivesse encontrado relação apareceria uma seta entre as duas transações”



Outra maneira de ver isto é prestar atenção à maneira como GeneXus nos mostra, na transação Flight, o atributo identificador do aeroporto.

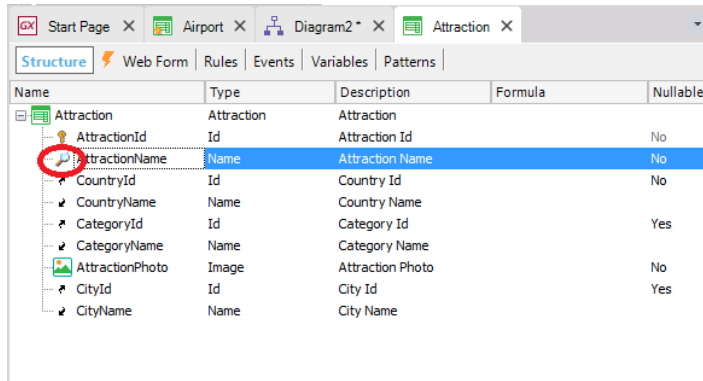
Vemos que é sinalizado com este símbolo, que na transação Airport também está em AirportName.

Name	Type	Description	Formula	Nullable
Airport	Airport	Airport		
✶ AirportId	Id	Airport Id		No
✶ AirportName	Name	Airport Name		No
✶ CountryId	Id	Country Id		No
✶ CountryName	Name	Country Name		
✶ CityId	Id	City Id		No
✶ CityName	Name	City Name		

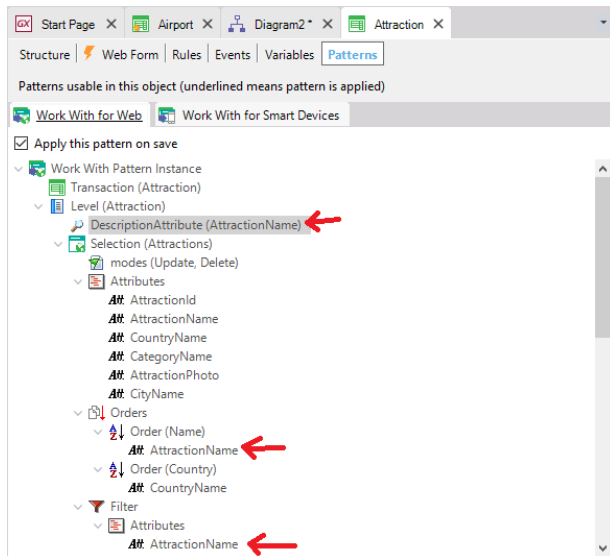
Name	Type	Description	Formula	Null...
Flight	Flight	Flight		
✶ FlightId	Id	Flight Id		No
✶ FlightDepartureAirportId	Id	Flight Departure Airp...		No
✶ FlightDepartureAirportName	Name	Flight Departure Airp...		No

Este símbolo está indicando que o atributo é o que melhor descreve o aeroporto em um caso, ou ao voo no outro. Quando criamos a estrutura de uma transação, GeneXus escolhe com base nos tipos de dados de seus atributos ao qual parece melhor descrevê-la, mas o usuário pode alterá-lo para outro, ou decidir que não há nenhum atributo desse tipo. Este "atributo descritor" é usado por exemplo pelo pattern Work With para permitir a filtragem por ele, ordenar por ele, etc.

Recordemos do pattern aplicado a Attraction:



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		



Patterns usable in this object (underlined means pattern is applied)

Work With for Web Work With for Smart Devices

☒ Apply this pattern on save

Work With Pattern Instance

Transaction (Attraction)

Level (Attraction)

DescriptionAttribute (AttractionName) ←

Selection (Attractions)

modes (Update, Delete)

Attributes

Attr: AttractionId

Attr: AttractionName

Attr: CountryName

Attr: CategoryName

Attr: AttractionPhoto

Attr: CityName

Orders

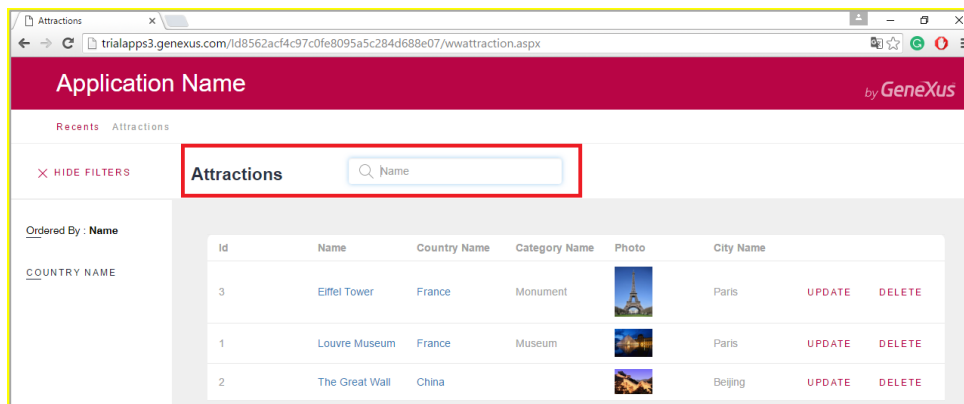
Order (Name) ←

Order (Country) ←

Filter

Attributes

Attr: AttractionName ←






Application Name by GeneXus

Recents Attractions

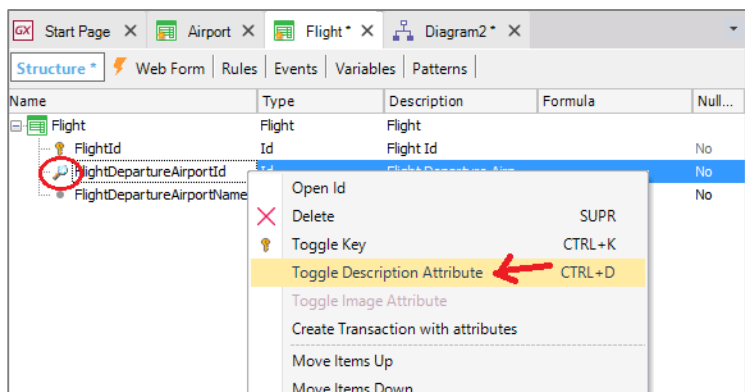
HIDE FILTERS Attractions

Ordered By: Name

COUNTRY NAME

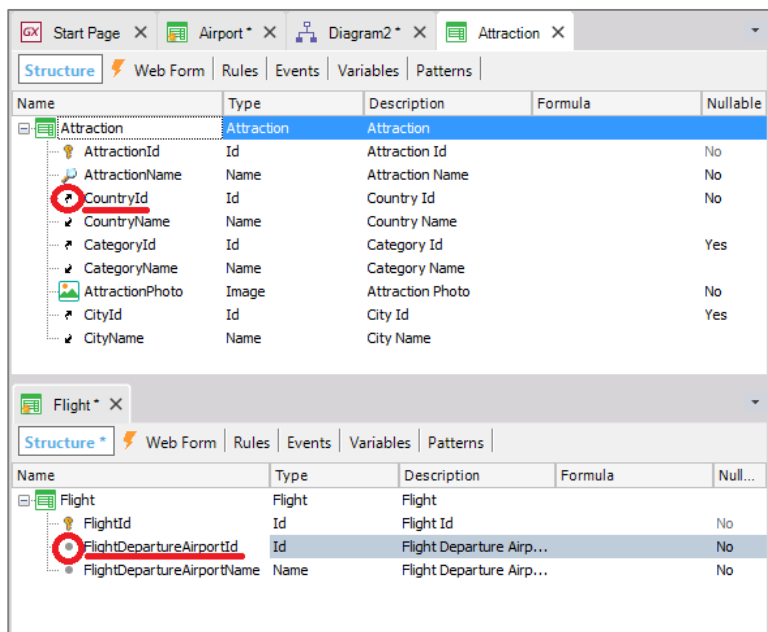
Id	Name	Country Name	Category Name	Photo	City Name		
3	Eiffel Tower	France	Monument		Paris	UPDATE	DELETE
1	Louvre Museum	France	Museum		Paris	UPDATE	DELETE
2	The Great Wall	China			Beijing	UPDATE	DELETE

Para Flight não nos interessa que o aeroporto da partida seja o atributo que melhor descreva o voo, assim o removemos.



Vemos, então, que fica sinalizado com este símbolo, que indica que é um atributo secundário e não é considerado como chave estrangeira...

Comparemos isso com a definição do identificador de país na transação Attraction



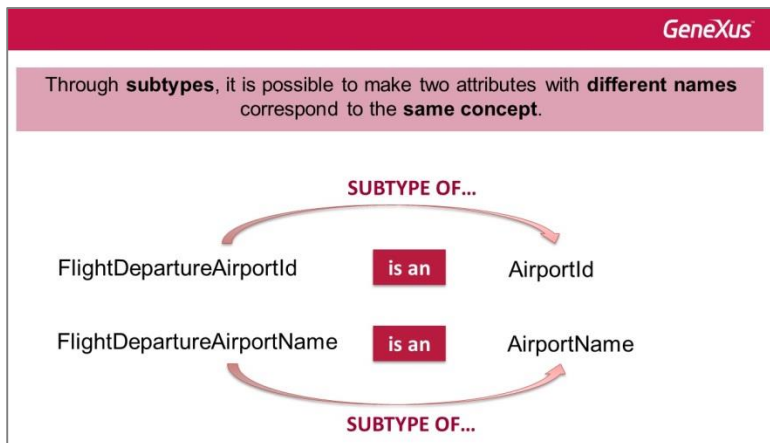
Em Attraction, o atributo CountryId tem este ícone que nos indica que ele é um atributo chave estrangeira... mas não é o caso do atributo FlightDepartureAirportId na transação Flight.

Então, **como fazemos para que GeneXus possa associar nomes distintos para o mesmo conceito?**

Precisamos que FlightDepartureAirportId mesmo se chamando diferente de AirportId, seja considerado como tal, ou seja, **como um identificador de aeroporto.**

E o mesmo vale para o nome do aeroporto.

Como podemos conseguir isso?



A resposta é: **mediante definição de subtipos.**

Quando um atributo tem nome diferente de outro já definido, mas ambos representam o mesmo conceito, **podemos "dizer" para GeneXus** que o novo atributo é subtipo do outro

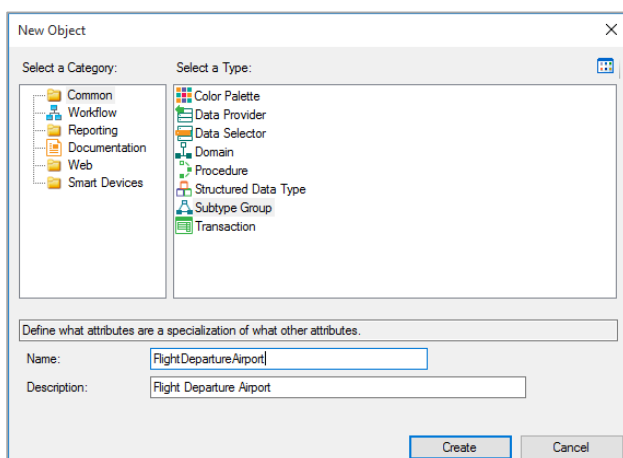
e a partir desse momento GeneXus vai considerá-los como se fossem exatamente a mesma coisa... portanto, GeneXus tratará o atributo **FlightDepartureAirportId** **exatamente como se fosse um AirportId**, ou seja, **irá identificá-lo como chave estrangeira na transação Flight.**

E o mesmo teremos com **FlightDepartureAirportName**: indicaremos que é subtipo de **AirportName**.

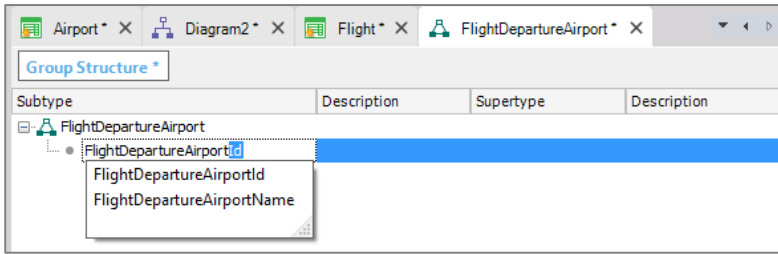
Vamos ver isso na prática.

Para definir subtipos, a primeira coisa que precisamos fazer é criar um grupo de subtipos.

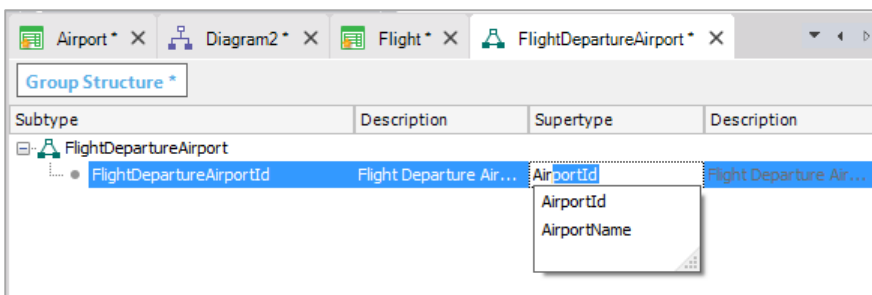
Assim, criamos um novo objeto do tipo Subtype group, e colocamos como nome **FlightDepartureAirport**:



Agora, na primeira linha digitamos a tecla com o ponto ('.') e GeneXus nos sugere os atributos que começam com "FlightDepartureAirport", que já havíamos definido na transação Flight::

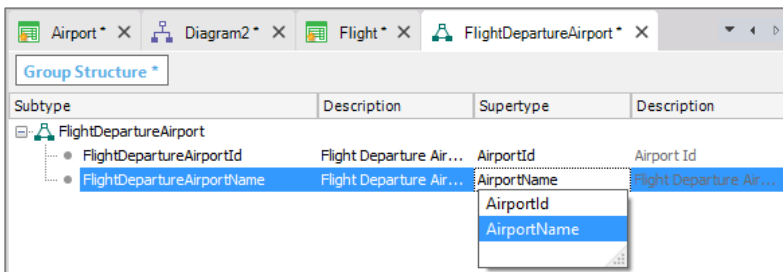


Escolhemos então FlightDepartureAirportId... pressionamos Tab e como queremos que FlightDepartureAirportId seja subtipo de AirportId, escolhemos como supertipo o atributo AirportId:



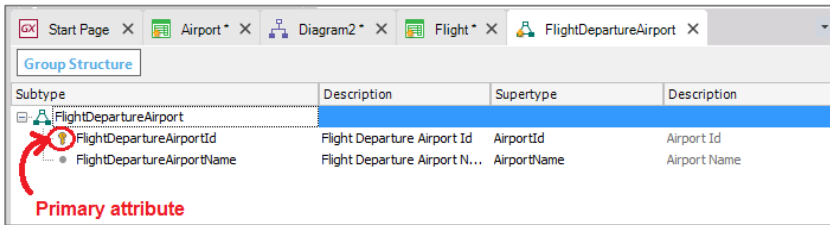
Podemos dizer que o supertipo é o atributo original, e subtipo é o atributo que conceitualmente corresponde a esse atributo original, mas que tem outro nome.

Agora adicionamos FlightDepartureAirportName, e definimos que seu supertipo é: AirportName

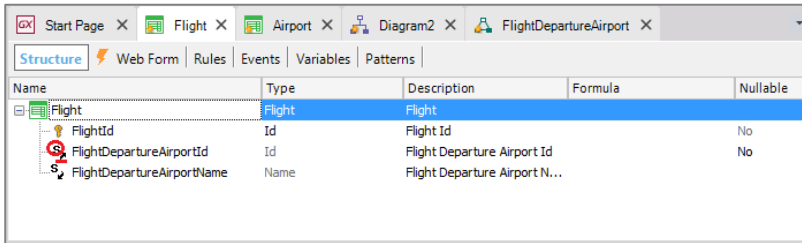


Gravamos.

Esse atributo se torna o identificador deste grupo de subtipos, portanto, o chamamos de "primário", e todos os atributos que adicionamos neste grupo, como FlightDepartureAirportName, dependerão dele, como na transação.

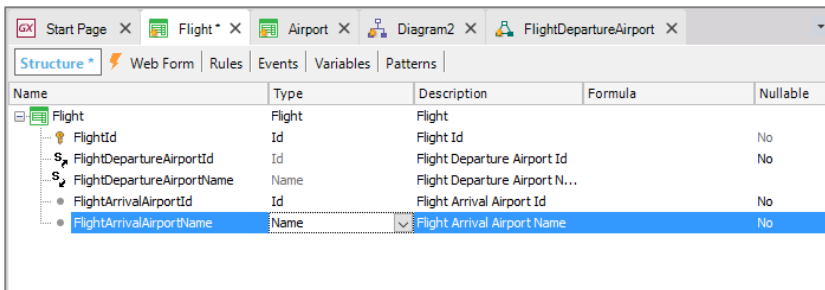


Vamos agora para a transação Flight e observamos que o atributo FlightDepartureAirportId tem o símbolo indicando que será tratado como uma chave estrangeira... e também o símbolo da letra s, que indica que ele é um atributo definido como subtipo:



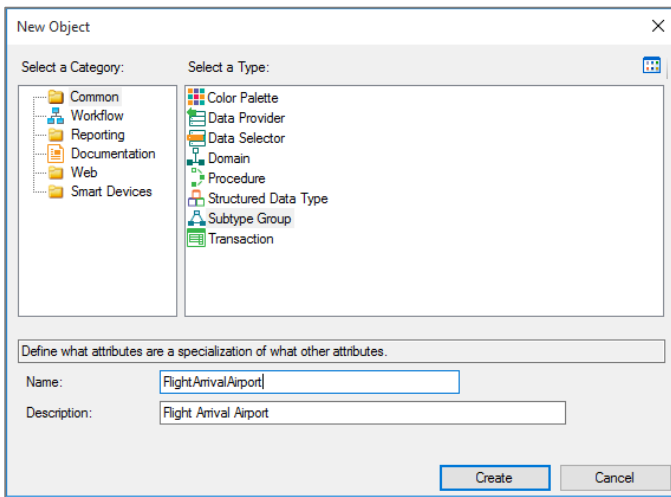
Vamos agora proceder da mesma forma para definir os atributos que permitam registrar o aeroporto onde o voo chega.

Vamos definir então os atributos FlightArrivalAirportId e FlightArrivalAirportName.



E gravamos.

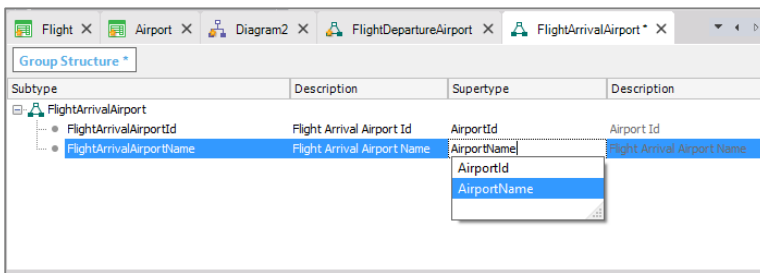
Criamos agora um novo objeto, de tipo: Subtype group e colocamos como nome FlightArrivalAirport:



Digitamos o ponto ('.')... GeneXus nos sugere os atributos que começam com "FlightArrivalAirport" e escolhemos FlightArrivalAirportId.

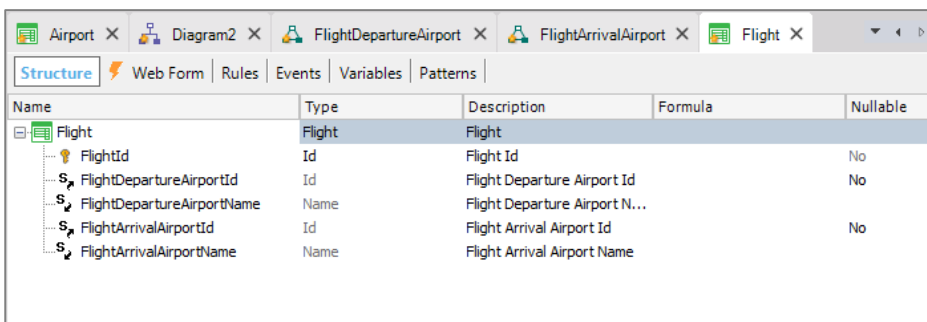
Pressionamos Tab e declaramos que seja subtipo de AirportId

Agora adicionamos FlightArrivalAirportName... e definimos que seu supertipo é: AirportName.



Gravamos.

Vejamos novamente a estrutura da transação Flight...

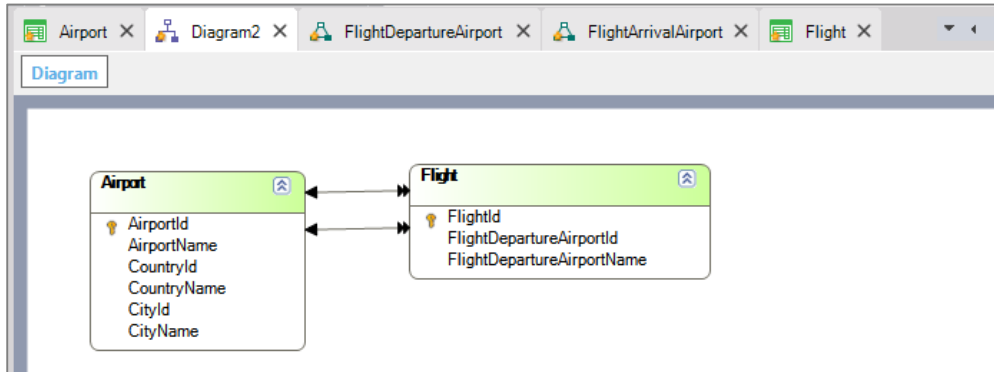


E voltamos agora a analisar o diagrama de transações que havíamos criado antes.

Vemos que agora GeneXus sim coloca a seta: GeneXus considera os atributos subtipos de identificadores de aeroporto em Flight, exatamente como se tivéssemos referenciado AirportId.

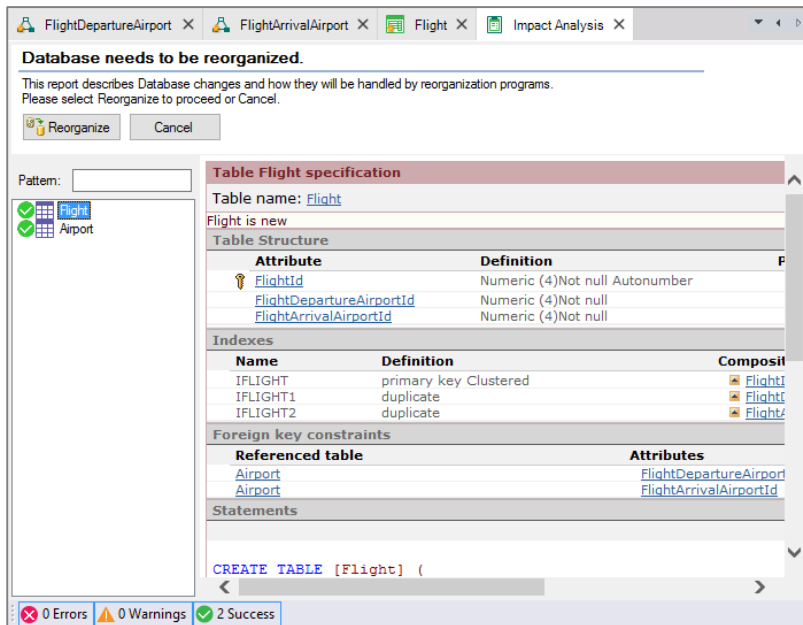
Vemos então que GeneXus encontrou a relação entre o Flight e o Airport.

Observemos que, embora em GeneXus esteja aparecendo uma única seta no diagrama, sendo rigorosos deveriam aparecer duas.



Vejamos em funcionamento tudo isto. Pressionemos F5...

O banco de dados deverá ser reorganizado pois deverão criar-se tabelas Flight e Airport. Concordamos, então, pressionamos Reorganize.





Em primeiro lugar definiremos aeroportos, então por isso executamos a transação Airport.

Incluimos o aeroporto "Guarulhos", indicamos o país: Brasil e a cidade: São Paulo:

Airport

« < > » SELECT



Id	1
Name	Guarulhos
Country Id	1 
Country Name	Brazil
City Id	2 
City Name	Sao Paulo

CONFIRM CANCEL DELETE

Agora vamos registrar o aeroporto "Charles de Gaulle"... selecionamos: França e a cidade: Paris. Confirmamos.

Airport

« < > » SELECT

Id	2
Name	Chales de Gaulle
Country Id	2 
Country Name	France
City Id	1 
City Name	Paris

CONFIRM CANCEL DELETE

Agora vamos registrar um voo.

Executamos a transação Flight... Como Aeroporto de partida vamos escolher "Guarulhos" e como aeroporto de chegada: "Charles de Gaulle":

Flight

SELECT

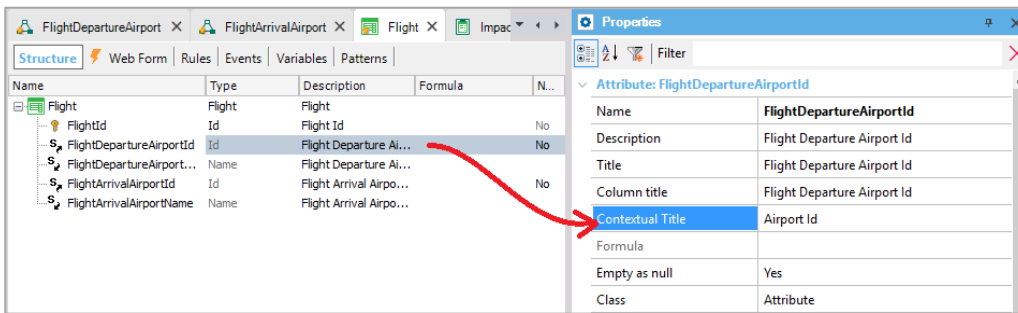
Id	1
Airport Id	<input type="text" value="i"/> <input type="text" value="u"/>
Airport Name	Guarulhos
Airport Id	2 <input type="text" value="u"/>
Airport Name	Chales de Gaulle

CONFIRM CANCEL DELETE

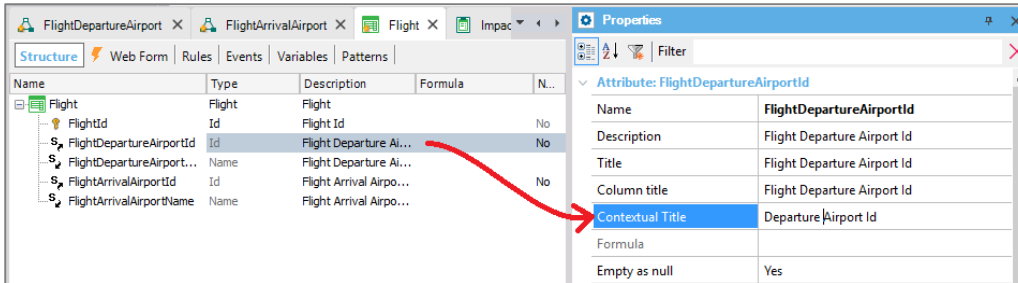
Vemos que as etiquetas dos atributos não nos indicam que este é o aeroporto de partida e este de chegada. Se formos ao form da transação em GeneXus e nos posicionamos no campo do primeiro aeroporto:

The screenshot shows the GeneXus IDE with the 'Flight' form and the 'Properties' window. The 'Flight' form contains fields for 'Id', 'Airport Id', 'Airport Name', and 'Airport Id' (again), with 'CONFIRM', 'CANCEL', and 'DELETE' buttons at the bottom. The 'Properties' window shows the 'Attribute/Variable: FlightDepartureAirportId' with the 'Label Caption' property set to '=FlightDepartureAirportId.ContextualTitle'. A red arrow points from the 'FlightDepartureAirportId' field in the form to the 'Label Caption' property in the Properties window.

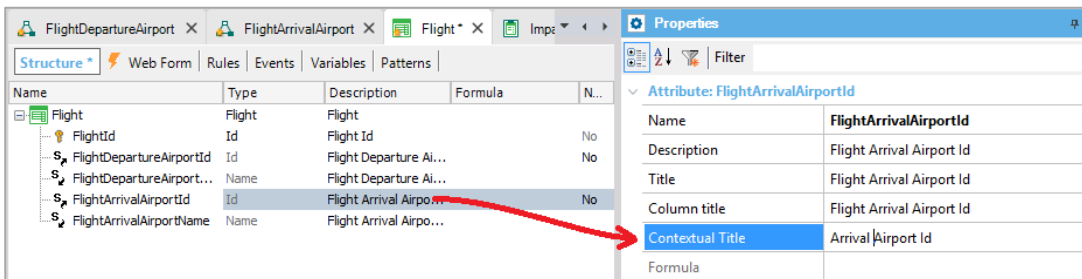
vemos que o que diz o rótulo, ou seja, seu Caption, está sendo extraído da propriedade ContextualTitle do atributo. Se vamos pesquisar é aqui onde ele aparece:



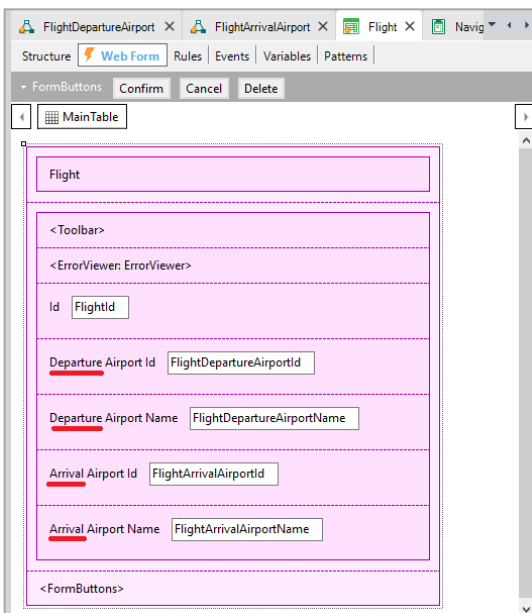
Lhe adicionamos “Departure”:



E fazemos o mesmo com o nome e adicionamos “Arrival” ao aeroporto de chegada:



Vemos o Form:



Pressionamos F5 novamente.

Vamos agora incluir outro voo.

Tentamos digitar no Aeroporto, 15... aparece o aviso de que este aeroporto não existe

The screenshot shows a 'Flight' form with the following fields and values:

- Id:** 0
- Departure Airport Id:** 15. A yellow tooltip message says "No matching 'Flight Departure Airport'." with a small 'u' icon.
- Departure Airport Name:** (empty)
- Arrival Airport Id:** (empty, with a dropdown arrow icon)
- Arrival Airport Name:** (empty)

At the bottom, there are two buttons: **CONFIRM** (in a red box) and **CANCEL**.

A consistência dos dados está sendo controlada. E a lista de seleção está sendo oferecida..... Vemos, assim, que temos os mesmos controles e ajudas que apareceram quando os atributos eram chaves estrangeiras com seus nomes originais... como vimos neste vídeo... **mas agora se trata de atributos subtipos deles.**

E esta é justamente a ideia: **que definindo subtipos que conseguimos definir que nomes de atributos diferentes correspondem ao mesmo conceito.**

É por isso que este atributo e este outro serão interpretados como chaves estrangeiras e que estes serão inferidos a partir dos primeiros.

The screenshot shows the 'Flight' form with the following fields and values:

- Id:** 1
- Departure Airport Id:** 1. A red 'FK' label is to the left. A red arrow points from this field to the 'Departure Airport Name' field.
- Departure Airport Name:** Guarulhos
- Arrival Airport Id:** 2. A red 'FK' label is to the left. A red arrow points from this field to the 'Arrival Airport Name' field.
- Arrival Airport Name:** Chales de Gaulle

At the bottom, there are three buttons: **CONFIRM** (in a red box), **CANCEL**, and **DELETE**.

Mas como é que GeneXus sabe que o nome deste aeroporto deve ser inferido neste e não a partir deste?

Flight

<< < > >> SELECT

Id 1

Departure Airport Id 1

Departure Airport Name **Guarulhos**

Arrival Airport Id 2

Arrival Airport Name Chales de Gaulle

CONFIRM CANCEL DELETE

Não é pela ordem em que os atributos aparecem nem pelo nome que lhes demos. Sabe disso porque esse atributo foi definido em um grupo com este. E este, em **outro** grupo, com este outro:

Flight X

Structure Web Form Rules Events Variables Patterns

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		
FlightArrivalAirportId	Id	Flight Arrival Airport Id		No
FlightArrivalAirportName	Name	Flight Arrival Airport Name		

FlightDepartureAirport X

Group Structure

Subtype	Description	Supertype	Description
FlightDepartureAirport			
FlightDepartureAirportId	Flight Departure Airport Id	AirportId	Airport Id
FlightDepartureAirportName	Flight Departure Airport Name	AirportName	Airport Name

FlightArrivalAirport X

Group Structure

Subtype	Description	Supertype	Description
FlightArrivalAirport			
FlightArrivalAirportId	Flight Arrival Airport Id	AirportId	Airport Id
FlightArrivalAirportName	Flight Arrival Airport Name	AirportName	Airport Name

Portanto, é essencial agrupar no mesmo grupo de subtipos os atributos que se correspondem.

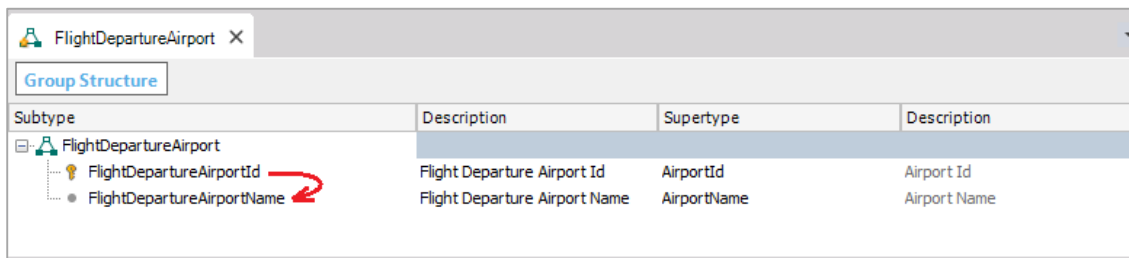
O uso de subtipos permitiu-nos representar uma situação que ocorre na realidade: neste caso, um voo tem dois aeroportos que cumprem um papel diferente: um é o aeroporto de partida e outro é o aeroporto de chegada. E os grupos nos permitem diferenciar ambos os papéis e conectar os atributos de cada papel.

Vamos destacar que não incluímos todos os subtipos no mesmo grupo, nem os dois atributos de subtipos primários de um lado em um grupo e os dois atributos secundários em outro. Não.

Agrupamos os atributos que definem o aeroporto de partida juntos em um grupo e os atributos que definem o aeroporto de chegada juntos em outro grupo.

Repito: isto é assim porque GeneXus entende com este grupo:

que quando se insere valor para este identificador de aeroportoFlightDepartureAirportId



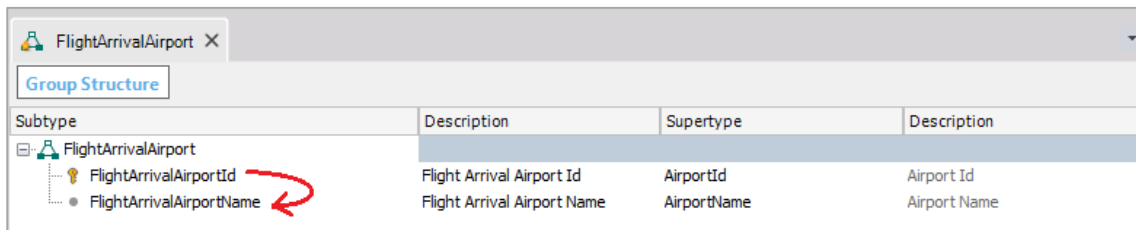
Subtype	Description	Supertype	Description
FlightDepartureAirport			
FlightDepartureAirportId	Flight Departure Airport Id	AirportId	Airport Id
FlightDepartureAirportName	Flight Departure Airport Name	AirportName	Airport Name

o nome do aeroporto correspondente a **este identificador**, deve ser carregado **neste atributo FlightDepartureAirportName**

e não no outro nome de aeroporto que está na transação.

Da mesma forma, GeneXus entende que ao digitar valor para o identificador do aeroporto: FlightArrivalAirportId

o nome do aeroporto correspondente deve ser carregado no atributo FlightArrivalAirportName.



Subtype	Description	Supertype	Description
FlightArrivalAirport			
FlightArrivalAirportId	Flight Arrival Airport Id	AirportId	Airport Id
FlightArrivalAirportName	Flight Arrival Airport Name	AirportName	Airport Name

Bem. Agora vamos supor que na transação Flight, queremos para cada aeroporto, ver, além de seu nome, seu país e sua cidade.

Isso simplesmente se resolve **definindo mais atributos de subtipos em cada grupo, ajudando o desenvolvedor, dando-lhes nomes significativos e não se esquecendo de indicar quem são seus supertipos...** e desta forma GeneXus vai entender que para o subtipo primário do grupo, deverá inferir todo o resto da informação associada.

Vamos fazê-lo.

Neste grupo de subtipos, iremos definir os atributos

Subtype	Description	Supertype	Description
FlightDepartureAirportIdId	Flight Departure Airport Id	AirportId	Airport Id
FlightDepartureAirportName	Flight Departure Airport Name	AirportName	Airport Name
FlightDepartureCountryId	Flight Departure Country Id	CountryId	Country Id
FlightDepartureCountryName	Flight Departure Country Name	CountryName	Country Name
FlightDepartureCityId	Flight Departure City Id	CityId	City Id
FlightDepartureCityName	Flight Departure City Name	CityName	City Name

FlightDepartureCountryId como subtipo de CountryId,

FlightDepartureCountryName como subtipo de CountryName,

FlightDepartureCityId como subtipo de CityId

e FlightDepartureCityName como subtipo de CityName.

Gravamos.

E agora vamos adicionar estes novos atributos à estrutura da transação Flight

Gravamos.

Name	Type	Description	Formula	Nullable
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		

Subtype	Description	Supertype	Description
FlightDepartureAirportIdId	Flight Departure Airport Id	AirportId	Airport Id
FlightDepartureAirportName	Flight Departure Airport Name	AirportName	Airport Name
FlightDepartureCountryId	Flight Departure Country Id	CountryId	Country Id
FlightDepartureCountryName	Flight Departure Country Name	CountryName	Country Name
FlightDepartureCityId	Flight Departure City Id	CityId	City Id
FlightDepartureCityName	Flight Departure City Name	CityName	City Name

E fazemos o mesmo para o outro grupo de subtipos....

Subtype	Description	Supertype	Description
FlightArrivalAirport			
FlightArrivalAirportId	Flight Arrival Airport Id	AirportId	Airport Id
FlightArrivalAirportName	Flight Arrival Airport Name	AirportName	Airport Name
FlightArrivalCountryId	Flight Arrival Country Id	CountryId	Country Id
FlightArrivalCountryName	Flight Arrival Country Name	CountryName	Country Name
FlightArrivalCityId	Flight Arrival City Id	CityId	City Id
FlightArrivalCityName	Flight Arrival City Name	CityName	City Name

Definimos

FlightArrivalCountryId como subtipo de CountryId,

FlightArrivalCountryName subtipo de CountryName,

FlightArrivalCityId subtipo de CityId

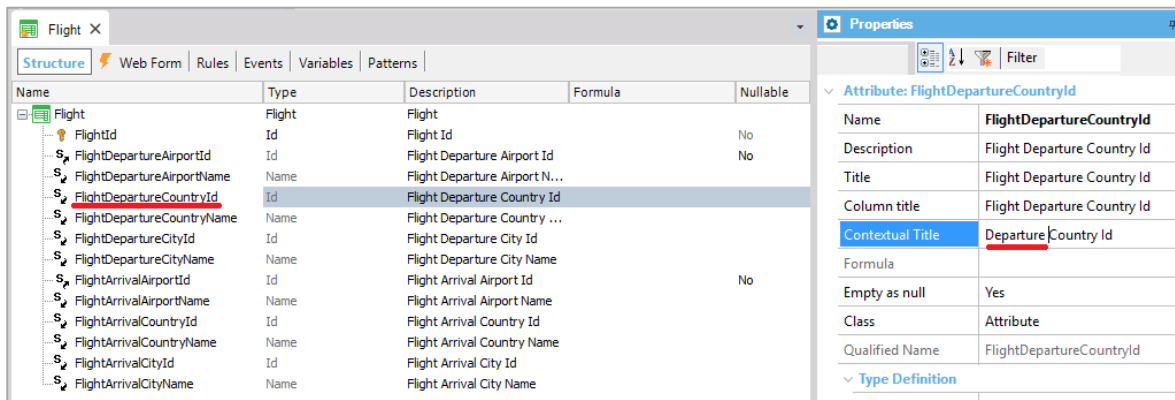
e FlightArrivalCityName subtipo de CityName.

Salvamos e também os adicionamos à estrutura da transação de Flight.

Gravamos.

Name	Type	Description	Formula	Nullable
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		

Como fizemos antes, a cada atributo adicionado alteramos o título contextual para que o rótulo na tela indique a função:



Novamente pressionamos F5 para executar a aplicação.

Abrimos a transação Flight, consultamos nosso primeiro voo e podemos ver de cada aeroporto seu país e sua cidade.

Flight

<<

<

>

>>

SELECT

Id

1

Departure Airport Id

Departure Airport Name

Guarulhos

Departure Country Id

1

Departure Country Name

Brazil

Departure City Id

2

Departure City Name

Sao Paulo

Arrival Airport Id

2

Arrival Airport Name

Chales de Gaulle

Arrival Country Id

2

Arrival Country Name

France

Arrival City Id

1

Arrival City Name

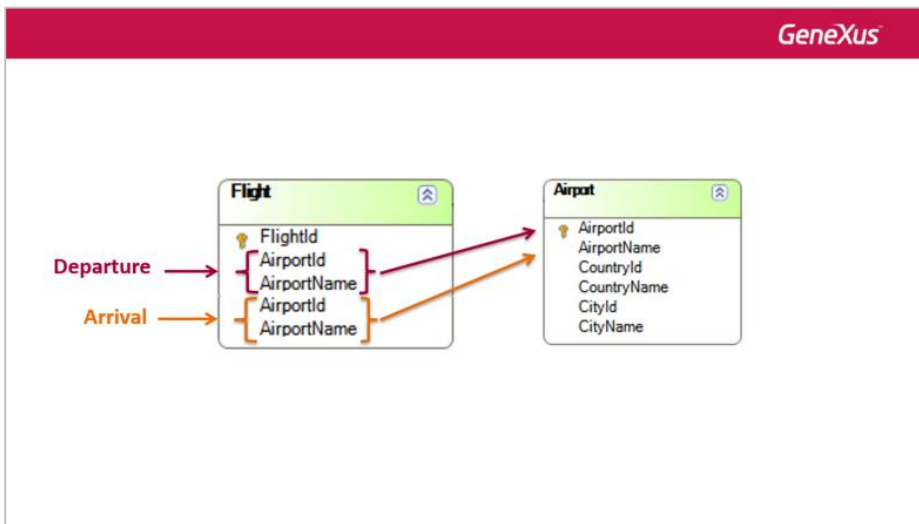
Paris

CONFIRM

CANCEL

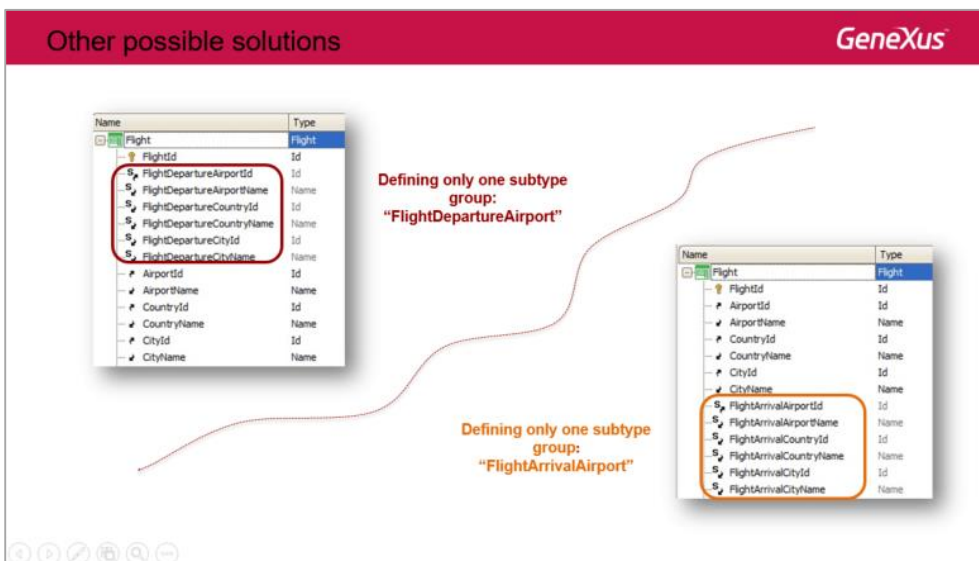
DELETE

Desta forma, vimos como resolver uma dupla referência ao mesmo conceito, mas com diferentes papéis, uma vez que os dois aeroportos deviam ser obtidos a partir da mesma tabela



mas cada um tinha um papel diferente.

Por último, é importante saber que **estas** poderiam também ter sido soluções válidas:



Nesta primeira proposta

foi definido um único grupo de subtipos: o grupo correspondente ao aeroporto de partida... e foi deixado aos atributos com o nome original (airportid, airportname, countryId, etc.) para a entrada do destino.

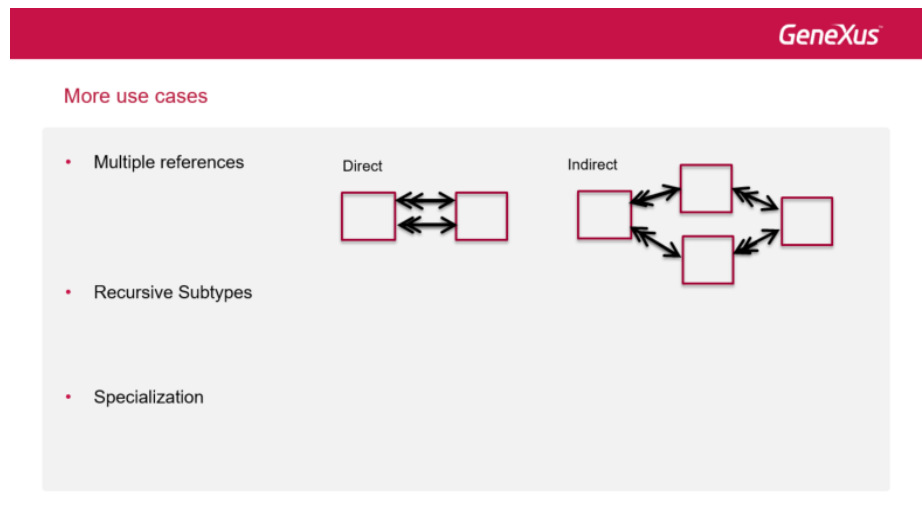
E isso é completamente válido.

Nesta segunda proposta... um único grupo de subtipos foi definido também, mas neste caso para o grupo que corresponde ao **aeroporto da chegada**.

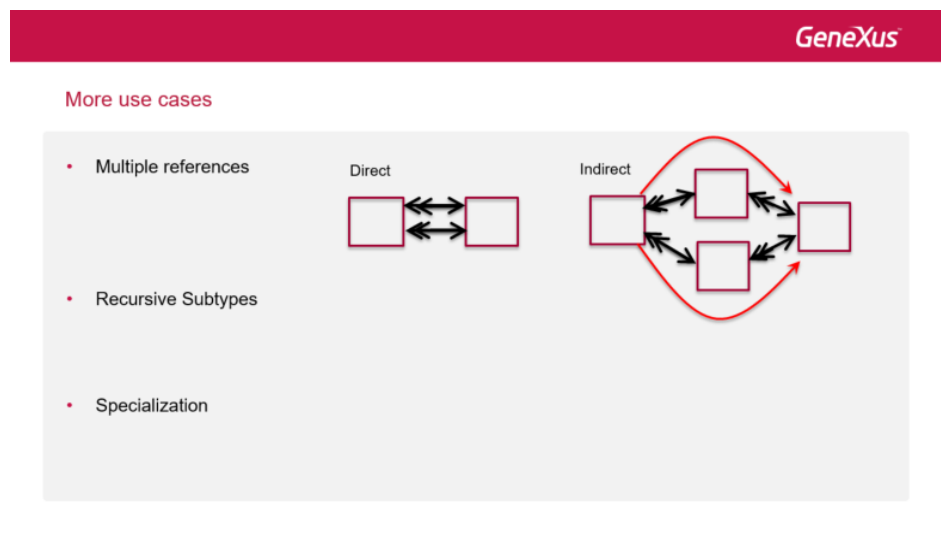
Em resumo, subtipos nos permitem dizer a GeneXus como associar diferentes nomes de atributo a um

mesmo conceito. E como vimos, as validações e todo o comportamento dos subtipos serão idênticos como se tivéssemos usado os atributos originais.

Há muitos outros casos em que é necessário modificar o nome de um atributo para evitar conflito ou ambiguidade. Neste vídeo vimos o caso de referências múltiplas de uma tabela a outra, mas estas referências não precisam ser diretas.



A partir desta tabela temos duas maneiras de chegar a esta outra tabela. Então foram necessários subtipos para diferenciá-los.



Temos também o caso de uma entidade que deve referenciar a si mesma:

More use cases

- Multiple references

Direct



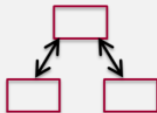
Indirect



- Recursive Subtypes



- Specialization



por exemplo, uma transação de funcionários, na qual entre as informações do funcionário deve ser registrado o chefe, que também é um funcionário.

Ou o caso em que temos uma entidade que registra informações gerais, por exemplo, de pessoas (como o nome, número de telefone, endereço, etc.) e então temos entidades que são uma especialização do outro, por exemplo, clientes e passageiros, que em particular são pessoas.

More use cases

- Multiple references

Direct



Indirect



- Recursive Subtypes



- Specialization

Estes são apenas alguns exemplos de muitos que existem. Apenas os mencionamos.. Não vamos estudá-los neste curso.

Para finalizar, atualizamos as alterações no GeneXus Server. Adicionamos os comentários:

e pressionamos Commit:

Flight X
Attraction X
Team Development X

Commit to: http://sandbox.genexusserver.com/15/home.aspx?TravelAgency_0

Airport and Flight transactions added.
FlightDepartureAirport and FlightArrivalAirport subtype groups added. |

Pattern:
Recent Comments

Category:
Folder:

Pending Commits (24/24)
Ignored Objects
Refresh

<input type="checkbox"/>	<input type="checkbox"/>	Name /	Type	Description	Modified On	Module	Action	Last Synchronize	User
<input checked="" type="checkbox"/>		Airport	Table	Airport	11/07/2016 04:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		Airport	Transaction	Airport	11/07/2016 05:...	Root Module	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		AirportId	Attribute	Airport Id	11/07/2016 04:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		Airport...	Attribute	Airport Name	11/07/2016 04:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		Diagra...	Diagram	Diagram1	08/07/2016 05:...	Root Module	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		Diagra...	Diagram	Diagram2	11/07/2016 05:...	Root Module	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		Flight	Table	Flight	12/07/2016 12:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		Flight	Transaction	Flight	12/07/2016 01:...	Root Module	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		FlightA...	Subtype Group	Flight Arrival Airp...	12/07/2016 12:...	Root Module	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		FlightA...	Attribute	Flight Arrival Airp...	12/07/2016 11:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		FlightA...	Attribute	Flight Arrival Airp...	12/07/2016 11:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...
<input checked="" type="checkbox"/>		FlightA...	Attribute	Flight Arrival Airp...	12/07/2016 01:...	None	Inserted	05/07/2016 01:0...	ARTECH:CFern...

☐ Add Knowledge Base properties to list
Remind me to move changes to...
Cancel
Commit

Commit
Update
History
Versions

