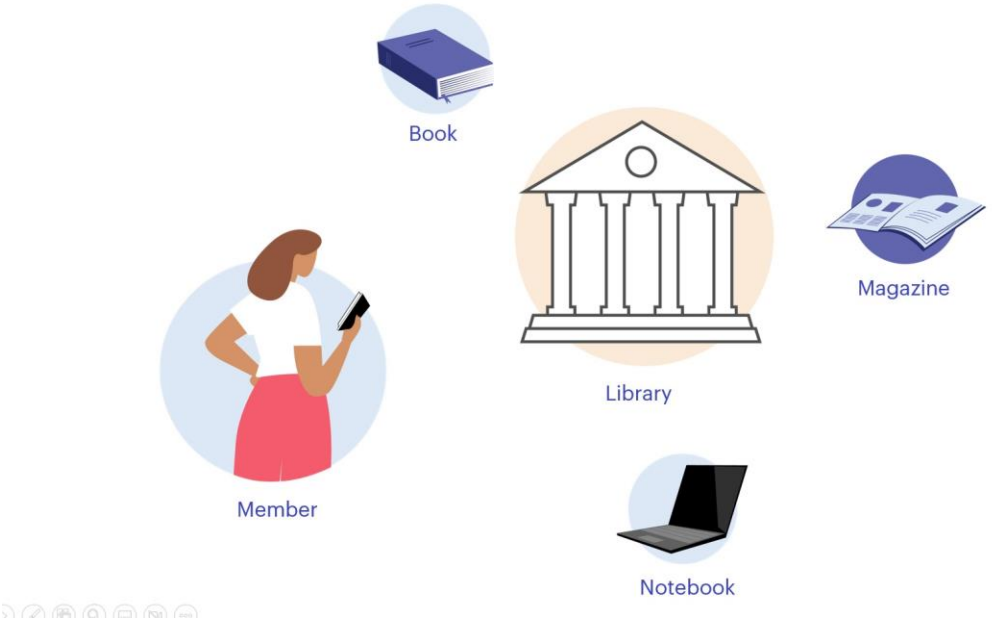
Análise do modelo de desenho de transações

Biblioteca

GeneXus

Library



Ao longo do curso anterior, estudamos tudo o que é necessário para poder modelar corretamente uma determinada realidade em GeneXus. Neste vídeo, através do estudo de uma realidade limitada, analisaremos diferentes opções para resolver o desenho de transações, utilizando um conjunto de recursos essenciais e resolvendo uma série de requisitos reais.

Suponhamos que é necessário desenhar uma aplicação GeneXus para a gestão de tarefas de uma Biblioteca em relação ao empréstimo de livros, revistas e notebooks.
Ela trabalha com Associados que podem acessar diversos materiais de leitura, sejam livros ou revistas, e equipamentos de informática que a Biblioteca oferece por empréstimo.

Library

Country

- Id
- Name

Member

- Document
- Name
- Address
- Image
- Phone
- Over 20 years old

Author

- Id
- Name
- Image
- Country
- May have several books published

As informações que precisam ser registradas são as seguintes:

País (Country)
Todo País é registrado com um identificador único e seu nome.

Associado (Member)
Todo Associado da Biblioteca é registrado com seu Documento de identidade, nome, endereço, foto e um telefone de contato.

Os associados da Biblioteca devem ter mais de 20 anos de idade.

Autor (Author)
Todo Autor é registrado com um identificador único, seu nome, foto e país de origem. Um Autor pode ter vários livros publicados.

Library

Country

- Id
- Name

Book

- Id
- Title
- Edition Date
- Copies
- Literary Genre
- Author
- Publishing House in charge of its publication

Member

- Document
- Name
- Address
- Image
- Phone
- Over 20 years old

Magazine

- Id
- Title
- Publication Date
- Image
- Copies

Author

- Id
- Name
- Image
- Country
- May have several books published

Notebook

- Id
- Image
- Description
- Status

Livro (Book)
Todo Livro é registrado com um identificador único, título, data de edição e quantidade de exemplares que dispõe a Biblioteca. Pode ser um Romance, um Ensaio, um livro de Poesia, etc., portanto, um livro corresponde a um gênero literário.

Além disso, um Livro tem um Autor e uma Editora responsável por sua publicação.

Revista (Magazine)
Toda Revista é registrada com um identificador único, título, data de publicação, a imagem da capa e a quantidade de exemplares disponíveis.

Notebook (Notebook)
A Biblioteca oferece notebooks aos seus Associados, por empréstimo, uma vez que muitos são escritores e pesquisadores. Cada notebook é registrado com um identificador único, sua imagem e uma breve descrição.

Além disso, é registrado seu estado (disponível ou emprestado).

Library

Country

- Id
- Name

Book

- Id
- Title
- Edition Date
- Copies
- Literary Genre
- Author
- Publishing House in charge of its publication

Loan

- Id
- Date
- Member
- Return Date
- 15-day deadline
- may include a maximum of 3 books, 4 magazines, and may or may not include borrowing a laptop
- Comment

Member

- Document
- Name
- Address
- Image
- Phone
- Over 20 years old

Magazine

- Id
- Title
- Publication Date
- Image
- Copies

Book Request

- Id
- Date
- Publishing House
- Book
- Number of copies

Author

- Id
- Name
- Image
- Country
- May have several books published

Notebook

- Id
- Image
- Description
- Status

Empréstimo (Loan)
A Biblioteca oferece livros, revistas e notebooks para empréstimo aos seus Associados.

Todo empréstimo é registrado com um identificador único, a data de retirada, o Associado e a data de devolução que deve ser determinada automaticamente.
Todo empréstimo é feito por um prazo de 15 dias, pode incluir no máximo 3 livros, 4 revistas, e pode ou não incluir o empréstimo de um notebook.

Pode ser retirado apenas um exemplar de cada publicação, e é possível registrar algum comentário que seja considerado necessário no nível do empréstimo de cada exemplar (por exemplo, que a capa está danificada, que está faltando alguma folha, etc.).

Além disso, a data em que se registra um novo empréstimo é sempre a atual e não deve ser possível modificá-la.

Solicitação de exemplares (BookRequest)
Muitas vezes acontece que certos livros têm alta demanda, e a Biblioteca decide solicitar mais exemplares.

Para isso, realiza uma Solicitação à Editora correspondente. O sistema deve controlar que sejam solicitados exemplares de livros a cargo da Editora indicada.

Library

Country

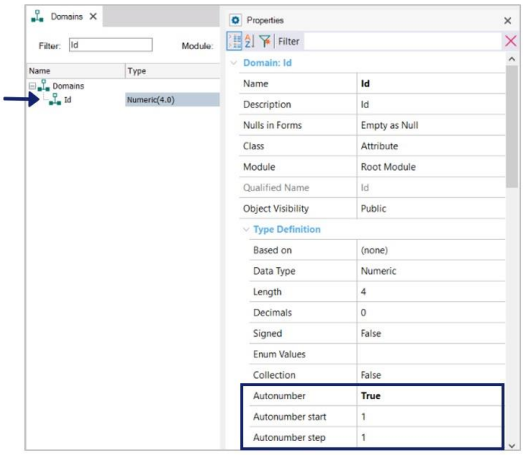
- Id
- Name

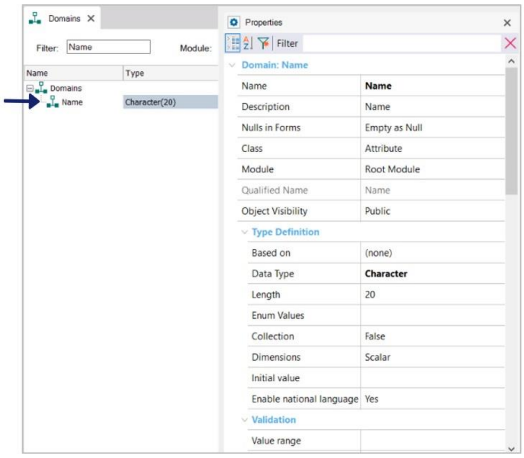
Member

- Document
- Name
- Address
- Image
- Phone
- Over 20 years old

Author

- Id
- Name
- Image
- Country
- May have several books published





Comecemos a analisar esta realidade.

A princípio, claramente se distinguem certas entidades simples que podemos começar a definir, como, por exemplo, País (Country), Associado (Member) e Autor (Author).

Mas antes, é bom levar em consideração que a maioria das entidades são registradas com um identificador que pode perfeitamente ser autonumerado, com exceção do Nro de Associado, que como mencionamos, é registrado com seu documento de identidade.
Portanto, definimos o Domínio Id, com a propriedade Autonumber em True.

Definimos também o domínio Name, como caractere de 20.

Transaction definition: Country

- Id
- Name

Name	Type	Description
Country	Country	Country
CountryId	Id	Country Id
CountryName	Name	Country Name

Attribute	Order
Country Indexes	
ICountry	Primary Key
CountryId	Ascending
UCountry	Unique
CountryName	Ascending

Vamos começar definindo a transação Country, com CountryId como chave primária e CountryName como atributo secundário.
CountryId fica baseado no domínio Id, e para controlar que o nome não se repita, definimos o correspondente índice unique. Esta mesma definição vale em todas as entidades onde é necessário controlar que o nome não se repita.

Transaction definition: Country

- Id
- Name

Weak 1-N

Name	Type	Description
Country	Country	Country
CountryId	Id	Country Id
CountryName	Name	Country Name
City	City	City
CityId	Id	City Id
CityName	Name	City Name

Attribute	Order
Country Indexes	
ICountry	Primary Key
CountryId	Ascending
UCountry	Unique
CountryName	Ascending

Name	Type
CountryCity Structure	
CountryId	Id
CityId	Id
CityName	Name

Se fosse necessário registrar as cidades de cada país, porque é de interesse saber a cidade de nascimento do autor, poderíamos modelá-la como uma entidade fraca em relação ao país, uma vez que uma cidade não existe fora desse contexto.

Portanto, seria adicionado um segundo nível à transação Country, mas ao considerar City como uma entidade fraca, ela não existirá como uma transação em si.

Isto significa que CityId não existirá como chave primária em nenhuma tabela e, portanto, para saber, por exemplo, a cidade de nascimento de um autor, será necessário o par composto pelos atributos CountryId, CityId, que serão a chave primária da tabela COUNTRYCITY associada ao segundo nível da transação Country.

De qualquer forma, como a descrição da realidade que estamos estudando não inclui o conceito da cidade, vamos modelar o País (Country) como uma transação simples.

Transaction definition: Member

- Document
- Name
- Address
- Image
- Phone
- Over 20 years old

Name	Type	Description	Formula
Member	Member	Member	
MemberDocument	Numeric(8.0)	Member Document	
MemberName	Name	Member Name	
MemberImage	Image	Member Image	
MemberAddress	Address, GeneXus	Member Address	
MemberPhone	Phone, GeneXus	Member Phone	
MemberBirthDate	Date	Member Birth Date	
MemberAge	Numeric(3.0)	Member Age	age(MemberBirthDate, today())

Member

Structure | Web Layout | Win Form | **Rules** | Events | Variables | Help | Documentation

```
1  Error("The member must be over 20 years old")
2      if MemberAge <= 20;
```

Passemos agora ao Associado. Para isso definimos a transação Member, com os seguintes atributos:

- MemberDocument, que corresponde ao documento de identidade e, portanto, não é autonumerado, então o definimos como numérico de 8 dígitos
- MemberName, do tipo Name
- MemberImage, do tipo Image
- MemberAddress, baseado no domínio semântico Address
- e MemberPhone, do tipo Phone

Mas, além disso, a realidade nos diz que os associados devem ter mais de 20 anos, portanto, precisamos de sua data de nascimento e da idade, que deverá ser calculada automaticamente.
Então, adicionamos à estrutura da transação os atributos:

- MemberBirthDate, do tipo Date
- e MemberAge, numérico de 3 dígitos

Como calculamos a idade do associado? Utilizando a função Age, que calcula a idade a partir da data de nascimento e a data atual.
Assim, definimos MemberAge como um atributo calculado que obtém seu valor a partir da seguinte expressão:
Age(MemberBirthDate, Today())

Por que utilizamos a função Today() e não a variável &today?

Porque não é possível utilizar variáveis na declaração de fórmulas.

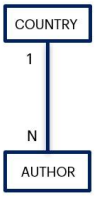
Para controlar que todo associado seja maior de 20 anos, basta declarar a seguinte regra Error:
Error(“The member must be over 20 years old”) if MemberAge <= 20;

É importante destacar que não estamos considerando nesta análise a declaração de regras para o controle básico de entrada de dados, como, por exemplo, controlar que não seja inserido um associado sem nome, etc.

Transaction definition: Author

- Id
- Name
- Image
- Country
- May have several books published

Name	Type	Description
Author	Author	Author
AuthorId	Id	Author Id
AuthorName	Name	Author Name
AuthorImage	Image	Author Image
CountryId	Id	Country Id
CountryName	Name	Country Name



Consideremos agora os Autores. Devemos definir a transação Author, com os atributos:

- AuthorId
- AuthorName
- AuthorImage

E sabemos que deve ser registrado o país dele, já que todo Autor tem um país de nascimento, portanto, adicionamos CountryId e CountryName, onde CountryId é uma chave estrangeira e CountryName um atributo que se infere a partir dessa chave estrangeira.
Desta forma, representamos que entre Country e Author existe uma relação 1-N.

Transaction definition: Book

- Id
- Title
- Edition Date
- Copies
- Literary Genre
- Author
- Publishing House in charge of its publication

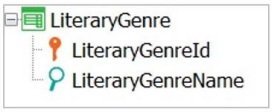
Pensemos agora sobre o conceito do Livro. Trata-se de um conceito forte que também se identifica com um valor autonumerado, tem um título, uma data de edição, um autor e também a quantidade de exemplares adquiridos pela Biblioteca.

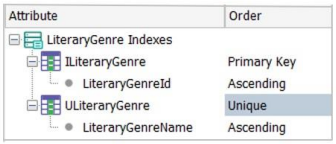
Mas, além disso, a realidade nos diz que um livro corresponde a um gênero literário, pois pode tratar-se de um Romance, um Ensaio, etc.

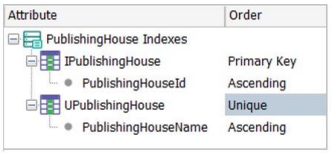
Então, como modelamos o conceito do Gênero literário? Se pensarmos que esses gêneros são finitos, uma primeira opção que podemos considerar é a criação de um domínio enumerado.

Transaction definition: Book

- Id
- Title
- Edition Date
- Copies
- Literary Genre
- Author
- Publishing House in charge of its publication







Mas a categorização dos gêneros literários tem mudado ao longo do tempo e é possível continuar a mudar, então não parece que o domínio enumerado seja a melhor opção, já que as mudanças deveriam ser feitas manualmente e não seria escalável.

Podemos considerar uma transação GeneroLiterario (LiteraryGenre), com os atributos:
- LiteraryGenreId
- e LiteraryGenreName

É uma boa decisão definir desde já um índice unique sobre o nome do gênero para evitar o registro de gêneros literários com o mesmo nome.

Mas o Livro requer registrar também o Autor e a Editora responsável por sua publicação. Já definimos o Autor como entidade, mas a Editora não. E embora não esteja declarada explicitamente, na análise surge a necessidade de modelar a entidade Editora.

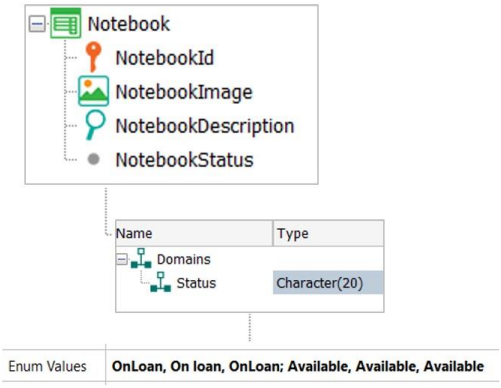
Assim, definimos a transação PublishingHouse com os atributos:
- PublishingHouseId
- e PublishingHouseName

Também podemos desde já criar o correspondente índice unique sobre o nome da editora.

Transaction definition: Magazine and Notebook

- Id
- Title
- Publication Date
- Image
- Copies

- Id
- Image
- Description
- Status

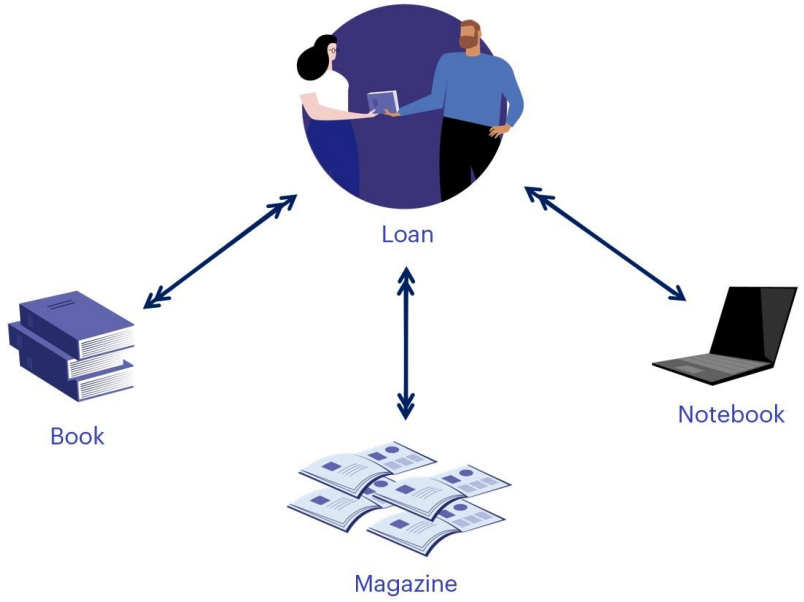


A Biblioteca também oferece Revistas e Notebooks aos seus associados, por isso definimos a transação Magazine, com os atributos:
- MagazineId
- MagazineTitle
- MagazineImage
- MagazinePublicationDate
- e MagazineCopies para registrar a quantidade de exemplares adquiridos.

E também a transação Notebook, com os atributos:
- NotebookId
- NotebookImage
- NotebookDescription do tipo LongVarChar
- e NotebookStatus

Um notebook está “Disponível” ou “Emprestado”, portanto, definimos o domínio enumerado Status com esses valores.

Transaction definition: Loan



Analisemos agora o conceito de Empréstimo, que é uma entidade fundamental nesta realidade.

A Biblioteca permite que um associado retire, a título de empréstimo e por 15 dias, um notebook, 3 livros e 4 revistas.

Temos, então, uma relação 1-N entre as entidades Empréstimo e Notebook, e uma relação N-N entre Empréstimo e Livro, e entre Empréstimo e Revista.

Como podemos modelá-la?

Transaction definition: Loan

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
NotebookDescription	Character(40)		

Analisemos uma primeira opção:

Criamos a transação Loan, e vamos definindo os seguintes atributos:
- LoanId
- LoanDate
- MemberDocument
- MemberName
- MemberAddress
- LoanReturnDate

É solicitado que a data de devolução seja calculada automaticamente. Um empréstimo é realizado por 15 dias, então declaramos o seguinte cálculo associado ao atributo LoanReturnDate:
LoanDate.AddDays(15)

Além disso, um empréstimo pode incluir ou não um notebook, então adicionamos:
- NotebookId
- e NotebookDescription

Mas nesta transação Loan, NotebookId é uma chave estrangeira não obrigatória, portanto, definimos sua propriedade Nullable com o valor Yes.

Transaction definition: Loan

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
NotebookDescription	Character(40)		
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
Book	Book		
BookId	Id		
BookTitle	Character(20)		
LoanBookComment	Character(40)		

```
Error("Only 3 books can be checked out")
    if LoanBooksQty > 3;
```

Um empréstimo inclui vários livros, com um máximo de 3. Portanto, definimos um segundo nível para registrá-los, incluindo um atributo LoanBookComment, para registrar algum comentário que seja necessário no momento de realizar o empréstimo.

Como controlamos que não sejam inseridos mais de 3 livros?

Podemos definir um novo atributo LoanBooksQty, que conte essa quantidade e, em seguida, condicionar esse valor em uma regra Error:
Error(“Only 3 books can be checked out”) if LoanBooksQty > 3;

Transaction definition: Loan

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
NotebookDescription	Character(40)		
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
LoanMagazinesQty	Numeric(4.0)	count(LoanMagazineComment)	
Book	Book		
BookId	Id		
BookTitle	Character(20)		
LoanBookComment	Character(40)		
Magazine	Magazine		
MagazineId	Id		
MagazineTitle	Character(20)		
LoanMagazineComment	Character(40)		

```
Error("Only 3 books can be checked out")
     if LoanBooksQty > 3;

Error("Only 4 magazines can be checked out")
     if LoanMagazinesQty > 4;
```

Mas o mesmo empréstimo pode incluir até 4 revistas, então podemos definir outro segundo nível, paralelo ao livro, para registrar as revistas. Da mesma forma, pode ser controlado que não sejam retiradas mais do que 4.

Transaction definition: Loan

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
NotebookDescription	Character(40)		
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
LoanMagazinesQty	Numeric(4.0)	count(LoanMagazineComment)	
Book	Book		
BookId	Id		
BookTitle	Character(20)		
LoanBookComment	Character(40)		
Magazine	Magazine		
MagazineId	Id		
MagazineTitle	Character(20)		
LoanMagazineComment	Character(40)		

```
Error("Only 3 books can be checked out")
     if LoanBooksQty > 3;

Error("Only 4 magazines can be checked out")
     if LoanMagazinesQty > 4;

Default(LoanDate, today());

noaccept(LoanDate);
```

A realidade descreve claramente que a data de registro de um empréstimo sempre deve ser a atual, sem possibilidade de modificação.

Para isso declaramos as seguintes regras:
Default(LoanDate, today());
noaccept(LoanDate);

Transaction definition: Loan

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
LoanMagazinesQty	Numeric(4.0)	count(LoanMagazineComment)	
Book	Book		
BookId	Id		
BookTitle	Character(20)		
LoanBookComment	Character(40)		
Magazine	Magazine		
MagazineId	Id		
MagazineTitle	Character(20)		
LoanMagazineComment	Character(40)		

Control Info	
Control Type	**Dynamic Combo Box**
Data Source From	Attributes
Item Values	NotebookId
Item Descriptions	**NotebookDescription**
Sort Descriptions	True
Conditions	**NotebookStatus = Status.Available;**
Instantiated Attributes	
Empty Item	**True**

Embora o objetivo deste vídeo seja nos concentrar em analisar o desenho de transações, vamos sugerir uma opção de implementação para que, no momento de registrar um empréstimo, sejam oferecidos apenas os notebooks que se encontrem no estado disponível (Available). Essa implementação, podemos fazer a partir da definição da própria transação e por isso a incluímos.

Vamos remover o atributo NotebookDescription, pois iremos "disfarçar" o identificador de um notebook com sua descrição. Selecionamos NotebookId e o definimos como um Dynamic Combo Box, com NotebookId na propriedade Item Values e NotebookDescription na propriedade Item Descriptions.

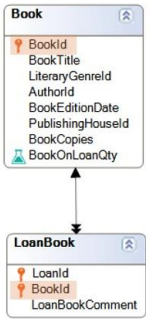
Isso carregará todos os notebooks registrados. Para que somente sejam oferecidos aqueles com estado disponível, declaramos a seguinte condição:
NotebookStatus = Status.Available;

Embora esta não seja a única forma de controlar que um notebook se encontre disponível no momento de registrar um empréstimo, é uma implementação que podemos resolver a partir da definição da própria transação.

Também devemos configurar a propriedade Empty Item com o valor True, pois NotebookId pode não ser escolhido, pois pode ser nulo.

Available quantities of Books and Magazines

Name	Type	Formula
Book	Book	
BookId	Id	
BookTitle	Character(20)	
BookEditionDate	Date	
AuthorId	Id	
AuthorName	Name	
BookCopies	Numeric(4.0)	
LiteraryGenreId	Id	
LiteraryGenreName	Name	
PublishingHouseId	Id	
PublishingHouseName	Name	
BookOnLoanQty	Numeric(4.0)	count(LoanBookComment)



Name	Type	Formula
Magazine	Magazine	
MagazineId	Id	
MagazineTitle	Character(20)	
MagazineImage	Image	
MagazinePubicationDate	Date	
MagazineCopies	Numeric(4.0)	

Outro requisito necessário é saber a disponibilidade dos livros ou revistas de interesse.

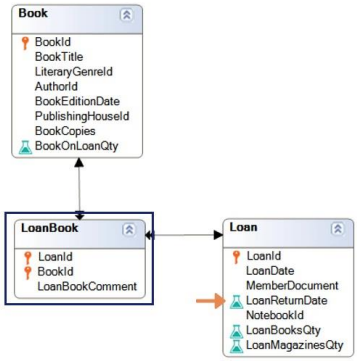
Sabemos a quantidade de exemplares adquiridos pela Biblioteca para cada livro e revista, mas não sabemos a quantidade que resta disponível.

Podemos obter esses valores de uma forma simples e com base no desenho de transações?
Se na transação Book, definirmos um novo atributo, BookOnLoanQty, podemos calcular a quantidade de exemplares emprestados e, portanto, saber a quantidade que está disponível.

O que acontece se a esse novo atributo BookOnLoanQty associarmos o cálculo Count(LoanBookComment)? Somará efetivamente a quantidade de exemplares desse livro que estão emprestadas?
A resposta é SIM, porque a tabela associada à transação onde definimos o cálculo é BOOK. E a tabela onde o cálculo é resolvido é LOANBOOK, já que fica determinada pelo atributo LoanBookComment.
Entre as duas tabelas há um atributo em comum, que é BookId e, portanto, é um filtro implícito que GeneXus aplicará, ou seja, esse cálculo retornará o total de exemplares do referido livro que foi emprestado.

Available quantities of Books and Magazines

Name	Type	Formula
Book	Book	
BookId	Id	
BookTitle	Character(20)	
BookEditionDate	Date	
AuthorId	Id	
AuthorName	Name	
BookCopies	Numeric(4.0)	
LiteraryGenreId	Id	
LiteraryGenreName	Name	
PublishingHouseId	Id	
PublishingHouseName	Name	
BookOnLoanQty	Numeric(4.0)	count(LoanBookComment,LoanReturnDate >= servernow())



Name	Type	Formula
Magazine	Magazine	
MagazineId	Id	
MagazineTitle	Character(20)	
MagazineImage	Image	
MagazinePubicationDate	Date	
MagazineCopies	Numeric(4.0)	

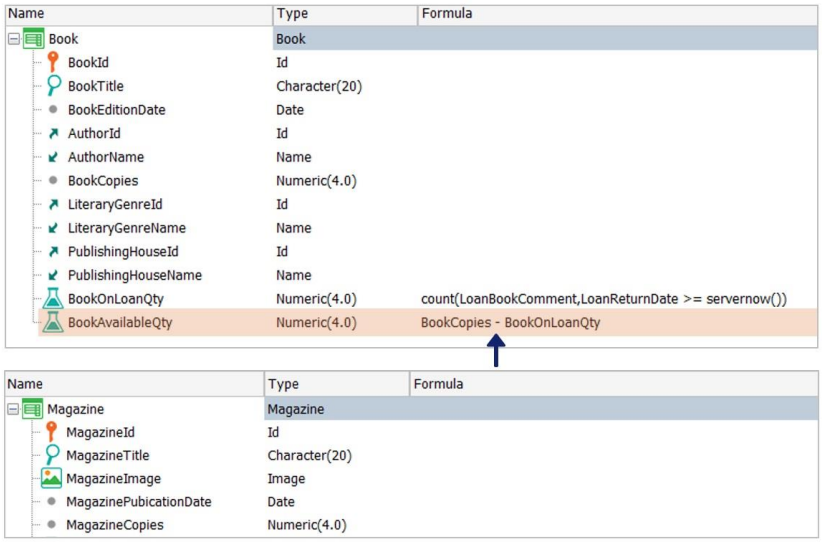
Mas precisamos saber a quantidade de exemplares atualmente em empréstimo, portanto, precisamos adicionar uma condição ao cálculo que nos permita contar a quantidade de exemplares em empréstimos atualmente vigentes. Para isso consideramos os empréstimos cuja data de devolução seja maior ou igual à atual.

Podemos declarar o seguinte:
Count(LoanBookComment, LoanReturnDate >= servernow())

A função servernow() permite obter a data e hora atual do server. Poderia ser utilizada também a função Today().

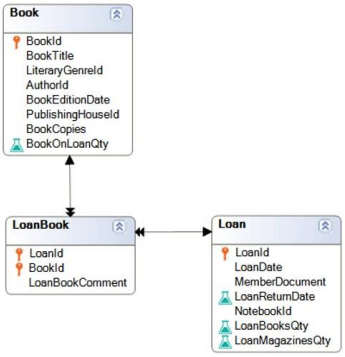
Podemos declarar essa condição de cálculo? Sim, porque o atributo envolvido (LoanReturnDate) pertence à tabela estendida da tabela onde é resolvido o cálculo declarado, que é LOANBOOK.

Available quantities of Books and Magazines



Name	Type	Formula
Book	Book	
BookId	Id	
BookTitle	Character(20)	
BookEditionDate	Date	
AuthorId	Id	
AuthorName	Name	
BookCopies	Numeric(4.0)	
LiteraryGenreId	Id	
LiteraryGenreName	Name	
PublishingHouseId	Id	
PublishingHouseName	Name	
BookOnLoanQty	Numeric(4.0)	count(LoanBookComment,LoanReturnDate >= servernow())
BookAvailableQty	Numeric(4.0)	BookCopies - BookOnLoanQty

Name	Type	Formula
Magazine	Magazine	
MagazineId	Id	
MagazineTitle	Character(20)	
MagazineImage	Image	
MagazinePubicationDate	Date	
MagazineCopies	Numeric(4.0)	



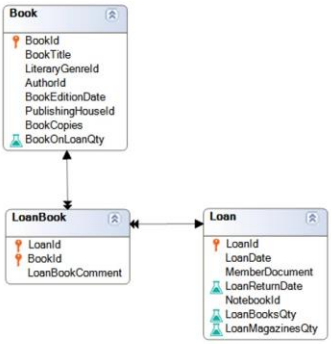
Portanto, se sabemos a quantidade total de exemplares e a quantidade de exemplares em empréstimos vigentes, então podemos saber a quantidade de exemplares atualmente disponíveis.

Portanto, a estrutura da transação Book fica assim:

Available quantities of Books and Magazines

Name	Type	Formula
Book	Book	
BookId	Id	
BookTitle	Character(20)	
BookEditionDate	Date	
AuthorId	Id	
AuthorName	Name	
BookCopies	Numeric(4.0)	
LiteraryGenreId	Id	
LiteraryGenreName	Name	
PublishingHouseId	Id	
PublishingHouseName	Name	
BookOnLoanQty	Numeric(4.0)	count(LoanBookComment,LoanReturnDate >= servernow())
BookAvailableQty	Numeric(4.0)	BookCopies - BookOnLoanQty

Name	Type	Formula
Magazine	Magazine	
MagazineId	Id	
MagazineTitle	Character(20)	
MagazineImage	Image	
MagazinePubicationDate	Date	
MagazineCopies	Numeric(4.0)	
MagazineOnLoanQty	Numeric(4.0)	count(LoanMagazineComment, LoanReturnDate >= servernow())
MagazineAvailableQty	Numeric(4.0)	MagazineCopies - MagazineOnLoanQty



Da mesma forma, podemos saber a quantidade de exemplares disponíveis de uma determinada revista:

Transaction definition: Loan

Name	Type	Formula
Loan	Loan	
LoanId	Id	
LoanDate	Date	
MemberDocument	Numeric(8.0)	
MemberName	Name	
MemberAddress	Address, GeneXus	
LoanReturnDate	Date	LoanDate.adddays(15)
NotebookId	Id	
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)
LoanMagazinesQty	Numeric(4.0)	count(LoanMagazineComment)
Book	Book	
BookId	Id	
BookTitle	Character(20)	
LoanBookComment	Character(40)	
BookAvailableQty	Numeric(4.0)	BookCopies - BookOnLoanQty
Magazine	Magazine	
MagazineId	Id	
MagazineTitle	Character(20)	
LoanMagazineComment	Character(40)	
MagazineAvailableQty	Numeric(4.0)	MagazineCopies - MagazineOnLoanQty

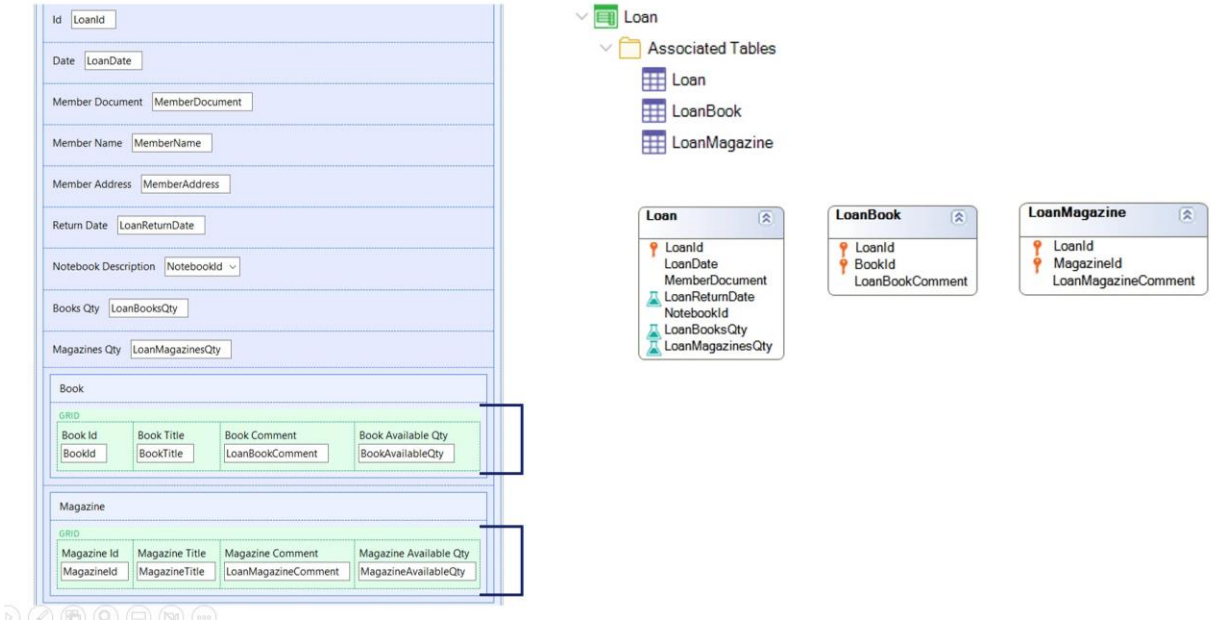
```
Error("Only 3 books can be checked out")
     if LoanBooksQty > 3;

Error("Only 4 magazines can be checked out")
     if LoanMagazinesQty > 4;

Default(LoanDate, today());

noaccept(LoanDate);

Error("There are no available copies of this book")
     if BookAvailableQty < 0;

Error("There are no available copies of this magazine")
     if MagazineAvailableQty < 0;
```

Agora temos disponíveis todas as informações de que precisamos para validar um empréstimo, então vamos adicionar os atributos necessários para validar que exista disponibilidade dos exemplares que se desejam retirar no empréstimo:

- BookAvailableQty, no nível correspondente aos livros, e
- MagazineAvailableQty, no nível correspondente às revistas

E controlamos com as seguintes regras Error:

Loan transaction: Form and Table structure



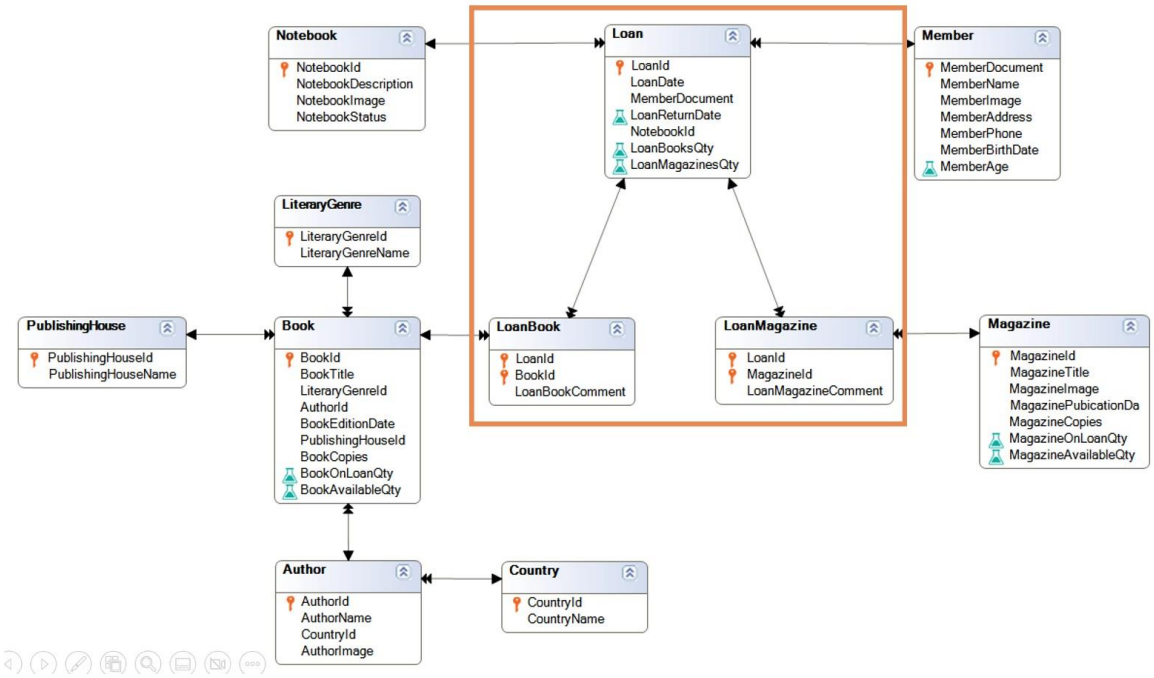
Até agora, resolvemos os requisitos para registrar Empréstimos. Mas observemos o form gerado para o desenho que fizemos.

Que tabelas associadas cria GeneXus para esta transação Loan?

Cria 3 tabelas:
- LOAN, associada ao primeiro nível, com LoanId como PK.
- LOANBOOK, associada ao nível dos livros, com LoanId e BookId como PK composta
- LOANMAGAZINE, associada ao nível das revistas, com LoanId e MagazineId como PK composta.

Ao definir dois níveis paralelos, vemos no form, grids paralelos. Isto talvez pode ser complicado para o usuário final, que pode solicitar simplificar o referido desenho e manipular as informações em diferentes telas mais simples.

Table Diagram



Para o desenho atual, GeneXus criou as seguintes tabelas. Observemos em particular as tabelas LOAN, LOANBOOK e LOANMAGAZINE, que correspondem às tabelas associadas aos três níveis da transação Loan definida.

Podemos propor outro desenho para a transação Empréstimo? Sim.

Loan entity – Option 2: Three transactions

- A screen for recording general loan information

- A screen for recording the books borrowed

- A screen for recording the magazines borrowed

Vejamos então uma segunda opção para modelar o Empréstimo(Loan).

Vamos supor que o usuário final nos solicita “nivelar” ou “aliviar” a carga desta tela e exibir as informações em três telas:

- uma tela para o registro geral do empréstimo, na qual se registra o empréstimo, com sua data, os dados do associado e se inclui notebook ou não.
- uma tela para o registro dos livros que se emprestam, na qual se registram os livros com os controles necessários já definidos.
- e uma tela para o registro das revistas que se emprestam, na qual se registram as revistas com os controles necessários já definidos.

Como podemos fazer isso?

Loan entity – Option 2: Three transactions

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes

```
Default(LoanDate, today());

noaccept(LoanDate);
```

Associated Tables:

Loan

A primeira tela corresponde aos dados gerais do empréstimo, assim, para a transação Loan apagamos os níveis, e deixamos as informações gerais, com as seguintes regras.

É então gerado o seguinte form:

Loan entity – Option 2: Three transactions

Name	Type	Formula	Nullable
BookLoan	BookLoan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		No
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
Book	Book		
BookId	Id		No
BookTitle	Character(20)		
LoanBookComment	Character(40)		No
BookAvailableQty	Numeric(4.0)	BookCopies - BookOnLoanQty	

Associated Tables:

- Loan
- BookLoanBook

```
Default(LoanDate, today());

noaccept(LoanDate);

Error("Only 3 books can be checked out")
    if LoanBooksQty > 3;

Error("There are no available copies of this book")
    if BookAvailableQty < 0;
```

Para oferecer outra tela com o registro de livros, precisamos de uma nova transação. A chamaremos de BookLoan.

Deseja-se inserir o identificador do empréstimo, ver suas informações gerais e inserir os livros. Portanto, a chave primária também será LoanId e, dessa forma, não será criada uma nova tabela pois já existe uma onde LoanId é chave primária. Dessa forma estaremos referenciando a mesma entidade Loan e apenas será criada a tabela associada ao segundo nível.

A transação BookLoan tem então a seguinte estrutura, com as seguintes regras.

É então gerado o seguinte form:

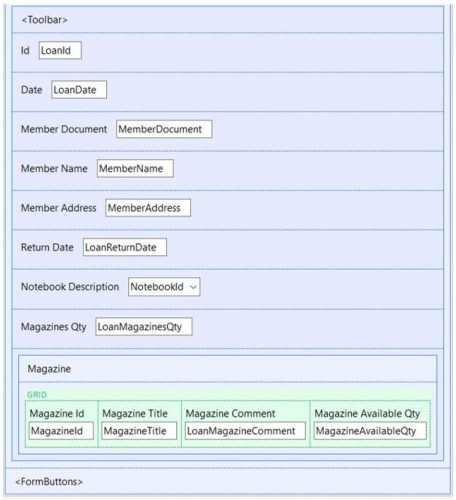
Loan entity – Option 2: Three transactions

Name	Type	Formula	Nullable
MagazineLoan	MagazineLoan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
LoanMagazinesQty	Numeric(4.0)	count(LoanMagazineComment)	
Magazine	Magazine		
MagazineId	Id		No
MagazineTitle	Character(20)		
LoanMagazineComment	Character(40)		No
MagazineAvailableQty	Numeric(4.0)	MagazineCopies - MagazineOnLoanQty	

Associated Tables:

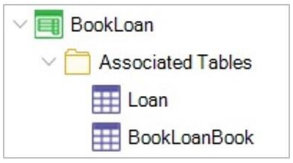
- Loan
- MagazineLoanMagazine

```
Default(LoanDate, today());

noaccept(LoanDate);

Error("Only 4 magazines can be checked out")
    if LoanMagazinesQty > 4;

Error("There are no available copies of this magazine")
    if MagazineAvailableQty < 0;
```



Faremos algo semelhante para registrar o empréstimo das revistas. Criamos a transação MagazineLoan com a seguinte estrutura, e as seguintes regras:

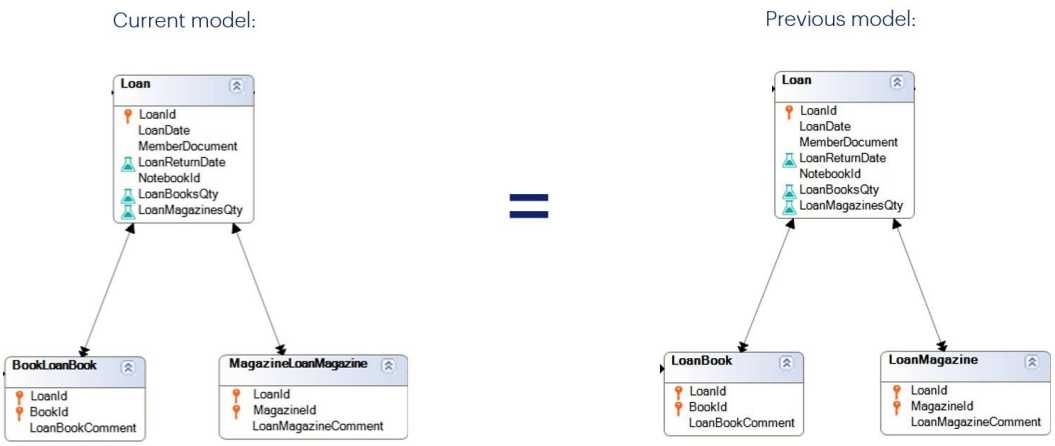
Loan entity – Option 2: Three transactions



Antes de analisar o diagrama de tabelas gerado, observemos as tabelas associadas às transações criadas nesta proposta. GeneXus também cria 3 tabelas:

- LOAN
- BOOKLOANBOOK
- e MAGAZINELOANMAGAZINE

Table Diagram

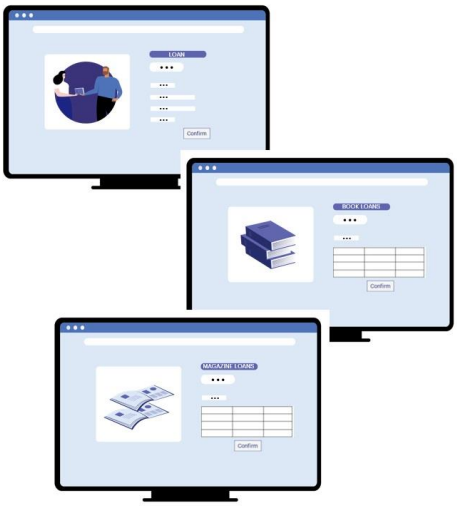


Observemos o diagrama de tabelas.

Se paramos nas tabelas LOAN, BOOKLOANBOOK e MAGAZINELOANMAGAZINE, podemos ver que possuem exatamente a mesma estrutura que as geradas no desenho proposto anteriormente.

Table Diagram

Current model:



Previous model:

Isto significa que a diferença entre os dois desenhos não está nas estruturas geradas, mas nas telas oferecidas ao usuário final para a manipulação das informações. Será sempre o usuário final quem nos indica qual desenho se adapta melhor ao seu negócio.

Do ponto de vista da estrutura da base de dados, é exatamente igual.

Loan entity – Option 3: Two transactions

Mas também podemos propor uma terceira opção para modelagem do Empréstimo.

Talvez possamos considerar que a Biblioteca baseia os empréstimos nos livros, e o usuário final solicita uma única tela onde são inseridos os dados gerais do empréstimo e também dos livros que se retiram. Em seguida, a partir de outra tela, são registradas as revistas que se incluem no empréstimo.

Loan entity – Option 3: Two transactions

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
Book	Book		
BookId	Id		No
BookTitle	Character(20)		
LoanBookComment	Character(40)		No
BookAvailableQty	Numeric(4.0)	BookCopies - BookOnLoanQty	

Associated Tables:

- Loan
- LoanBook

```
Default(LoanDate, today());

noaccept(LoanDate);

Error("Only 3 books can be checked out")
    if LoanBooksQty > 3;

Error("There are no available copies of this book")
    if BookAvailableQty < 0;
```

Para isso, consideramos a transação Loan, com um segundo nível para registrar os livros. Sua estrutura seria a seguinte, com as seguintes regras:

Loan entity – Option 3: Two transactions

Name	Type	Formula	Nullable
Loan	Loan		
LoanId	Id		No
LoanDate	Date		No
MemberDocument	Numeric(8.0)		No
MemberName	Name		
MemberAddress	Address, GeneXus		
LoanReturnDate	Date	LoanDate.adddays(15)	
NotebookId	Id		Yes
LoanBooksQty	Numeric(4.0)	count(LoanBookComment)	
Book	Book		
BookId	Id		No
BookTitle	Character(20)		
LoanBookComment	Character(40)		No
BookAvailableQty	Numeric(4.0)	BookCopies - BookOnLoanQty	

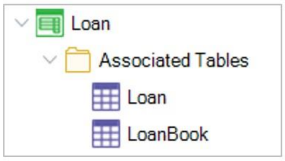
Associated Tables:

- Loan
- LoanBook

```
Default(LoanDate, today());

noaccept(LoanDate);

Error("Only 3 books can be checked out")
     if LoanBooksQty > 3;

Error("There are no available copies of this book")
     if BookAvailableQty < 0;
```

Para o registro das revistas que são incluídas no empréstimo, temos a transação MagazineLoan com a seguinte estrutura, e as seguintes regras:

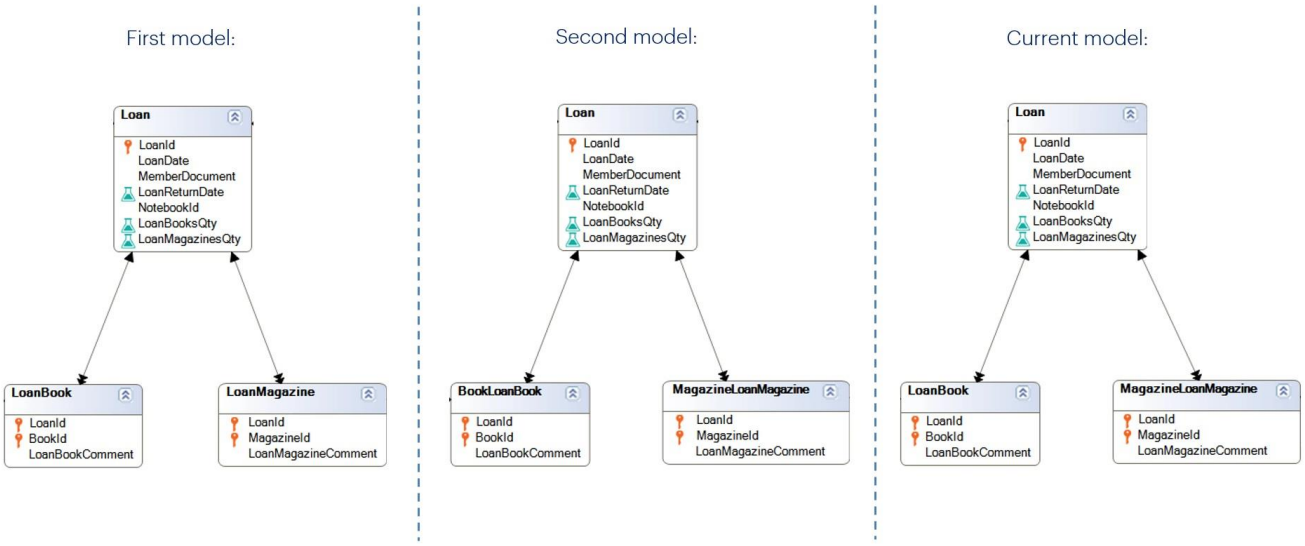
Loan entity – Option 3: Two transactions



Que tabelas cria GeneXus, associadas a estas transações? 3 tabelas: LOAN, LOANBOOK e MAGAZINELOANMAGAZINE.

- LOAN e LOANBOOK associadas à transação Loan
- LOAN e MAGAZINELOANMAGAZINE associadas à transação MagazineLoan

Table Diagram



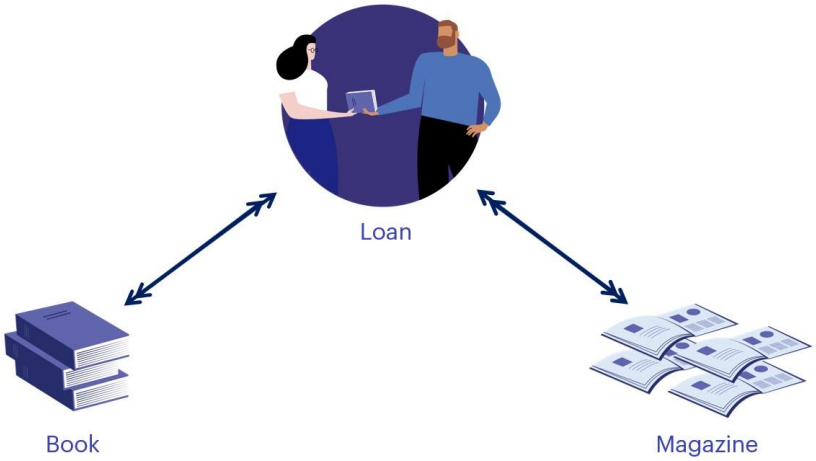
Observemos mais uma vez o diagrama de tabelas gerado.

Se novamente observamos a estrutura das tabelas LOAN, LOANBOOK e MAGAZINELOANMAGAZINE, vemos que são iguais às tabelas geradas nas propostas anteriores.

Estas propostas, então, mudam as telas oferecidas ao usuário final, mas não a estrutura da base de dados.

Se em vez de priorizar o empréstimo de livros, tivesse sido feito com as revistas, a estrutura das tabelas geradas permanecerá a mesma.

Reality

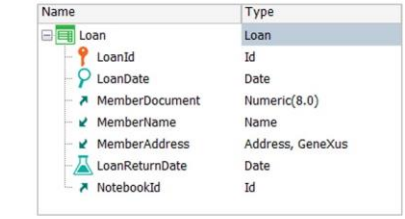


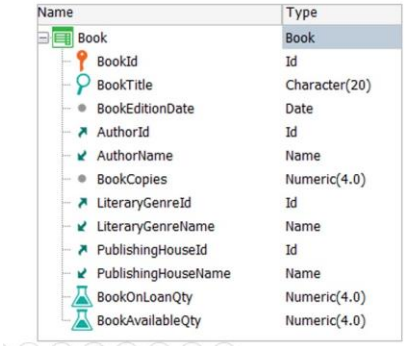
Agora, existe alguma outra opção totalmente plana que não inclua segundos níveis? Se o usuário final solicita telas simples sem grids, o que podemos lhe oferecer?

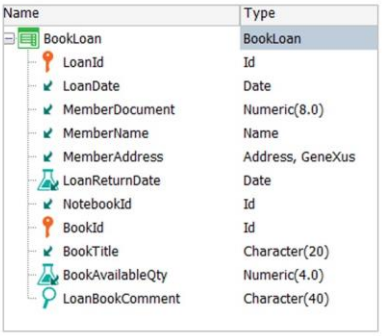
Lembremos que entre as entidades Empréstimo e Livro existe uma relação N-N, e entre Empréstimo e Revista também.

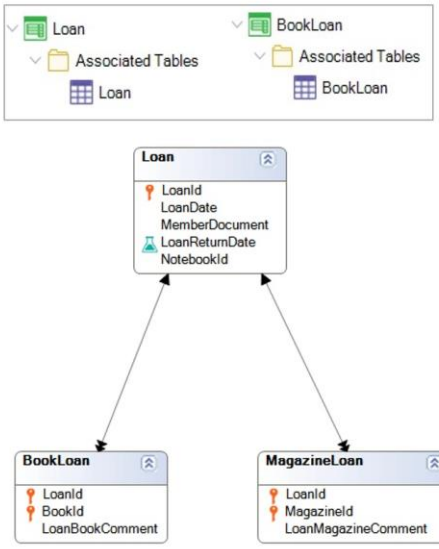
Vamos analisar para Empréstimo e Livro e será análogo para Empréstimo e Revista.

Loan entity – Option 4: Three simple transactions (one level transactions)









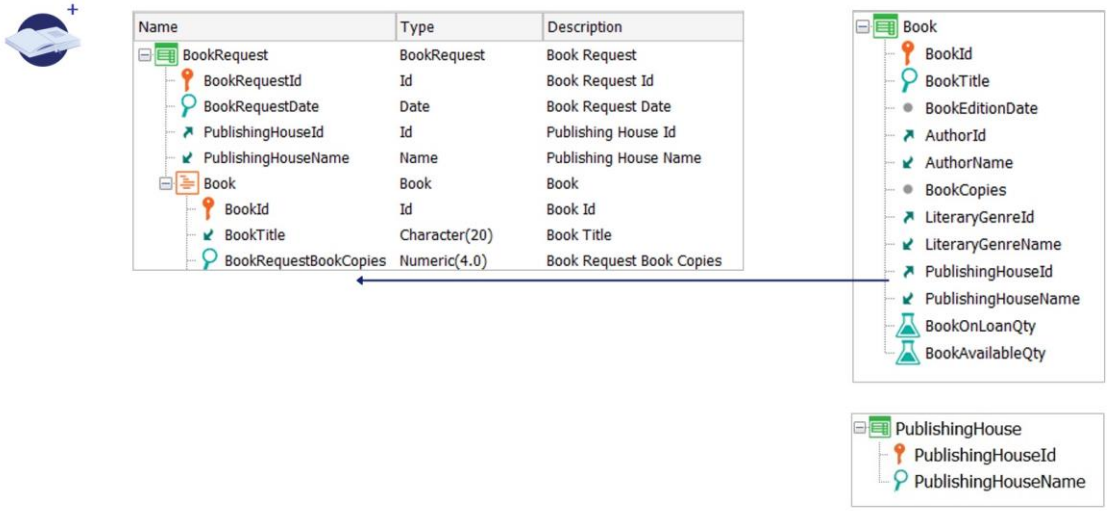
Consideremos as seguintes estruturas.

Para que entre elas exista uma relação N-N, precisamos de uma tabela de relação com a chave composta LoanId, BookId.

Definimos então a transação BookLoan com a chave primária composta de LoanId, BookId e a seguinte estrutura.

Quais tabelas são geradas? As mesmas que vimos anteriormente.

BookRequest transaction: Structure



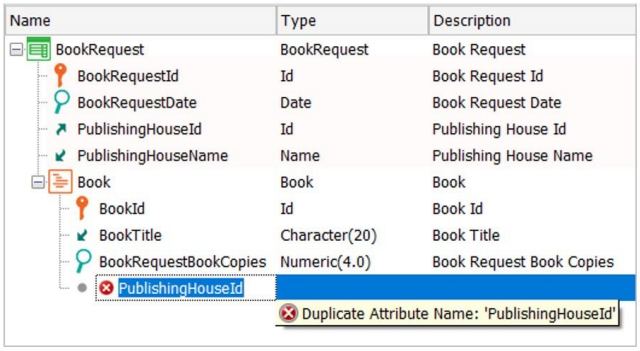
Nesse ponto, nos resta resolver a modelagem para a solicitação de novos exemplares de livros para uma determinada Editora.

Para isso vamos definir a transação BookRequest, que representará uma solicitação a uma Editora, com os seguintes atributos:
- BookRequestId, baseado no domínio Id
- BookRequestDate, com a data atual por padrão
- PublishingHouseId, já que a solicitação é a uma Editora
- PublishingHouseName, que será um atributo inferido a partir da chave estrangeira PublishingHouseId.

Entre esta nova entidade e o Livro (Book) existe uma relação N-N, uma vez que uma solicitação inclui vários livros e, por sua vez, um livro pode estar incluído em várias solicitações. Já analisamos que existem várias opções de desenho.
Vamos adicionar Book como segundo nível na Solicitação. Lembremos que é um requisito importante controlar que os livros indicados sejam publicados pela Editora responsável.
Se nos lembrarmos da estrutura da transação Book, vemos que possui PublishingHouseId como FK, já que um livro é publicado por uma Editora.

Podemos então adicionar esse atributo também nesta nova transação para que seja inferido por BookId e ser possível controlar se efetivamente trata-se de um livro publicado pela Editora à qual é feita a solicitação?

BookRequest transaction: Structure



Multiple reference conflict:
Where to define the subtype group?

O que acontece quando tentamos adicioná-lo? GeneXus nos dá um erro, porque o atributo PublishingHouseId já está declarado na estrutura da transação e não podemos adicioná-lo novamente.

Temos um problema de referência múltipla. Precisamos fazer referência, mais de uma vez, ao conceito da Editora. Então, como podemos resolver um conflito deste tipo? Com o uso de subtipos. Isto significa definindo novos atributos, com outros nomes, que façam referência ao mesmo conceito da Editora.

Mas... onde definimos o subtipo? No conceito da Editora que é referenciado no cabeçalho, ou que é referenciado no 2º nível? É o mesmo?

BookRequest transaction: Structure

Multiple reference conflict:
Where to define the subtype group?

Observemos o desenho de transações: o atributo PublishingHouseId está presente na estrutura da transação Book como FK, portanto, de BookId GeneXus infere este valor. Se posteriormente tanto BookId como PublishingHouseid se encontram presentes em outra transação, como ocorreria neste caso no 2º nível de BookRequest, então GeneXus aplicará essa relação e PublishingHouseId será uma FK inferida.

Portanto, seria correto mudar aqui o nome dos atributos para definir o grupo de subtipos necessário? Não, porque estaríamos rompendo esta relação e permitiríamos ao usuário adicionar um valor de PublishingHouseId diferente daquele definido previamente em Book.

BookRequest transaction: Subtype group in the first level

BookRequest	BookRequest
BookRequestId	Id
BookRequestDate	Date
BookRequestPublishingHouseId	Id
BookRequestPublishingHouseName	Name
Book	Book
BookId	Id
BookTitle	Character(20)
BookRequestBookCopies	Numeric(4.0)
PublishingHouseId	Id
PublishingHouseName	Name

```
Default(BookRequestDate, Today());

Error("The book doesn't belong to the Publishing House")
     if PublishingHouseId <> BookRequestPublishingHouseId;
```

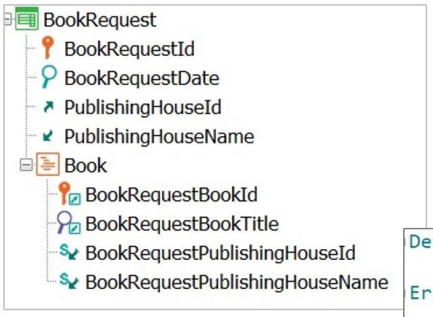
Group Structure

Subtype	Description	Supertype
BookRequestPublishingHouse		
BookRequestPublishingHouseId	Book Request Publishing House Id	PublishingHouseId
BookRequestPublishingHouseName	Book Request Publishing House Name	PublishingHouseName

Então, vamos olhar o cabeçalho de BookRequest. Aqui o atributo PublishingHouseId é uma FK comum, que não é inferida de nenhum lugar, portanto podemos removê-la perfeitamente, definir novos atributos e declará-los como subtipos de PublishingHouseId e PublishingHouseName. Então poderemos compará-los em uma regra Error para validar que os livros solicitados sejam da Editora indicada.

Para finalizar o requisito, declaramos as seguintes regras.
Como esta regra Error envolve atributos de ambos os níveis, então é uma regra que GeneXus associa ao 2º nível. Portanto, será disparada para cada linha. Isto significa que ao inserir um novo BookId, será avaliada a condição e, caso seja verdadeira, será disparado o erro.

BookRequest transaction: Subtype group throughout the second level




```
Default(BookRequestDate, Today());

Error("The book doesn't belong to the Publishing House")
     if PublishingHouseId <> BookRequestPublishingHouseId;
```

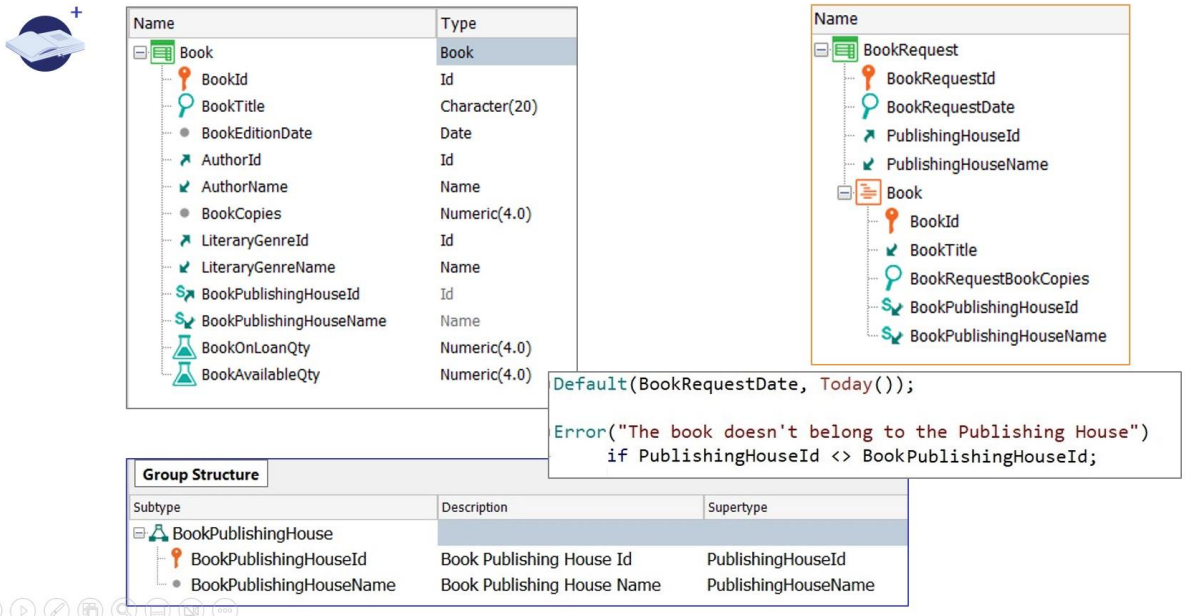
Group Structure

Subtype	Description	Supertype
BookPublishingHouse		
BookRequestBookId	Book Request Book Id	BookId
BookRequestBookTitle	Book Request Book Title	BookTitle
BookRequestPublishingHouseId	Book Request Publishing House Id	PublishingHouseId
BookRequestPublishingHouseName	Book Request Publishing House Name	PublishingHouseName

Resolvemos o requisito. E se tivéssemos definido todo o segundo nível como um grupo de subtipos?

Também o resolveríamos, com a diferença que (tal como analisamos detalhadamente no vídeo dos subtipos) embora esta solução possa ser menos óbvia e levar a ter que criar mais subtipos para poder inferir onde sejam necessários, tem como principal vantagem que a ambiguidade é resolvida na própria tabela que a causa.

BookRequest transaction: Subtype group in the Book transaction



Poderíamos propor alguma outra opção para evitar a definição dos subtipos na transação de dois níveis?

Sim, mas devemos deixar claro que como critério de trabalho sugerimos sempre abordar o conflito de referência na transação que o apresenta. Por isso o resolvemos na própria transação BookRequest.

Contudo. Poderíamos observar a transação Book e definir ali o grupo de subtipos.

O que ganhamos? Que na transação BookRequest não exista conflito de referência.
Mas devemos ter em mente que no caso de já termos consultas definidas sobre Book, por exemplo, um catálogo de livros, deverão ser atualizadas porque foram alterados os nomes de alguns atributos.

Por último, poderia ser evitado o uso de subtipos? Sim, por exemplo, chamando em cada linha, e antes de sua gravação, um processo que retorne se o livro é publicado pela Editora indicada ou não.
É evitada a definição do grupo de subtipos, mas é invocado em cada linha um processo para que avalie uma condição que pode ser resolvida na definição da própria transação.

Summary

Analyze, evaluate the reality to be modeled and implement the option we consider correct, always working together with the end user.

Por tudo o que foi visto, e para finalizar, lembremos que sempre devemos analisar, avaliar a realidade a ser modelada e implementar a opção que consideramos correta, e para isso é fundamental trabalhar sempre junto com o usuário final, que será aquele que nos orienta na seleção do desenho.

Não podemos esquecer que a aplicação deve ser uma ferramenta de apoio ao usuário final para uma melhor gestão e desenvolvimento de seu negócio.

GeneXus™

training.genexus.com
wiki.genexus.com