

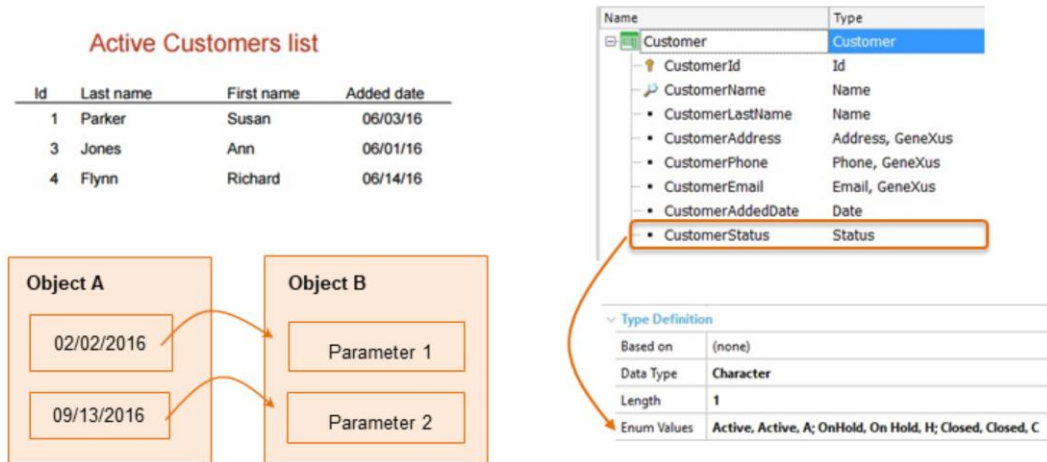
Data Selectors

Reutilizando definições

GeneXus 16

Cenário

- Clientes ativos inseridos entre duas datas fornecidas ... em diversas consultas!



Suponha que adicionamos o atributo CustomerStatus na transação Customer, a fim de representar um dos três status (ativo, em espera, fechado) que um cliente pode ter no sistema da agência de viagens. Para este fim, um tipo de dados enumerado **Status** foi criado, como mostrado na imagem.

Suponha que, em vários lugares da aplicação precisamos trabalhar com os clientes ativos, inseridos entre duas datas determinadas. Por exemplo:

1. Uma lista PDF que recebe um intervalo de datas (&start e &end) e mostra os clientes ativos que foram inseridos no sistema entre essas duas datas.

Neste exemplo, vamos falar sobre algo que será tratado mais detalhadamente mais tarde: um objeto pode receber valores do objeto chamador. Esses valores são necessários para executar sua ação (como em nosso exemplo, um intervalo de datas) mas para isso tem que ser capaz de recebê-los. Para habilitar um objeto a receber valores (o que chamamos de parâmetros), eles têm que ser indicados (vamos ver como se faz).

Cenário

- Clientes ativos inseridos entre duas datas fornecidas ... em diversas consultas!

The screenshot displays a web panel interface for data selection. On the left, there is a date range selector with 'Date From' set to 06/01/16 and 'Date To' set to 06/24/16. Below this is a table showing the results of a query:

Name	Last Name	Quantity
Susan	Parker	3
Ann	Jones	0
Richard	Flynn	0

In the center, there are two data tables. The first is the 'Invoice' table with columns: InvoiceId (Id), InvoiceDate (Date), CustomerId (Id), CustomerName (Name), CustomerLastName (Name), CustomerAddedDate (Date), CustomerStatus (Status), FlightId (Id), FlightDate (Date), and FlightPrice (Price). The second is the 'Customer' table with columns: CustomerId (Id), CustomerName (Name), CustomerLastName (Name), CustomerAddress (Address, GeneXus), CustomerPhone (Phone, GeneXus), CustomerEmail (Email, GeneXus), CustomerAddedDate (Date), and CustomerStatus (Status). The 'CustomerStatus' field in the Customer table is highlighted with a red box.

Below the tables, a 'Type Definition' window is open for the 'CustomerStatus' field. It shows the following properties:

Type Definition	
Based on	(none)
Data Type	Character
Length	1
Enum Values	Active, Active, A; OnHold, On Hold, H; Closed, Closed, C

2 - Em um web panel que mostra todos os clientes que têm faturas e seu número de faturas, com a possibilidade do usuário inserir um intervalo de datas para contar somente as faturas correspondentes a clientes ativos, inseridos no sistema entre essas datas. Se um cliente é inativo ou foi inserido fora essas datas, ele será incluído na lista, mas o número de faturas será zero.

Deve ser mencionado que um web panel é um tipo muito flexível de objeto GeneXus que permite projetar todos os tipos de consultas interativas ao banco de dados. Mais tarde neste curso, vamos ver este objeto mais detalhadamente.

Cenário

- Clientes ativos inseridos entre duas datas fornecidas ... em diversas consultas!

Id	Last name	First name	Quantity
1	Parker	Susan	3
2	Smith	Peter	0
4	Flynn	Richard	0

Name	Type
Invoice	Invoice
InvoiceId	Id
InvoiceDate	Date
CustomerId	Id
CustomerName	Name
CustomerLastName	Name
CustomerAddedDate	Date
CustomerStatus	Status
FlightId	Id
FlightDate	Date
FlightPrice	Price

Name	Type
Customer	Customer
CustomerId	Id
CustomerName	Name
CustomerLastName	Name
CustomerAddress	Address, GeneXus
CustomerPhone	Phone, GeneXus
CustomerEmail	Email, GeneXus
CustomerAddedDate	Date
CustomerStatus	Status

Type Definition	
Based on	(none)
Data Type	Character
Length	1
Enum Values	Active, Active, A; OnHold, On Hold, H; Closed, Closed, C

3. Em uma lista PDF, nós precisamos mostrar o mesmo que o web panel anterior.

SOLUÇÃO usar o que conhecemos até o momento

- Listagem de clientes Ativos inseridos entre duas datas.

```
Print Title
For each Customer
  Where CustomerStatus = Status.Active
  Where CustomerAddedDate >= &DateFrom
  Where CustomerAddedDate <= &DateTo
  Print Customer
Endfor
```

Parm(in: &DateFrom, in: &DateTo);
Valores recebidos de outro objeto
(parâmetros)

- Listagem de clientes Ativos inseridos entre duas datas e sua quantidade de faturas.

```
Print Title
For each Invoice
  Unique CustomerId
  &Qty = Count(InvoiceDate,
  CustomerStatus=Status.Active and CustomerAddedDate>=&DateFrom and CustomerAddedDate<=&DateTo)
  Print Customer
Endfor
```

Se implementamos as consultas mencionadas acima usando o conhecimento que temos até agora, poderíamos ver que, em ambos os casos, as três condições acima indicadas são repetidas.

Lembre-se que escrever três cláusulas Where é o mesmo que escrever apenas uma cujas condições são unidas por AND.

Nota:

Em geral, um objeto estabelece os parâmetros usados para troca de informações com o chamador por meio de uma regra: a regra parm. Neste caso, em ambas as listas vamos criar duas variáveis: &DateFrom e &DateTo, para receber (que é por isso que "in" está digitado) o intervalo de datas do chamador.

Definição

Data Selector

```
CustomerStatus = Status.Active  
and  
CustomerAddedDate >= &DateFrom  
and  
CustomerAddedDate <= &DateTo
```

- ✓ Economia e reuso
- ✓ Melhor manutenção

Data Selector Structure		Documentation
Structure	Type	Description
ActiveCustomers		Active Customers
Parameters		
DateFrom	Date	Date From
DateTo	Date	Date To
Conditions		
CustomerStatus=Status.Active		
CustomerAddedDate>=&DateFrom		
CustomerAddedDate<=&DateTo		
Orders		
CustomerStatus		
DefinedBy		

Para evitar ter que repetir as mesmas especificações em todo lugar que precisamos delas (o web panel e procedimentos anteriores, bem como em objetos de outro tipo, que veremos mais tarde), nós podemos fazer essas definições em um único lugar, dando-lhes um nome, e a partir daí podemos usar esse nome como referência. Aquele lugar é o objeto Data Selector.

Para otimizar a consulta em um comando For Each poderíamos ordenar por CustomerStatus, porque temos um filtro de igualdade por esse atributo.

Como podemos ver, no exemplo, criamos um data selector chamado "ActiveCustomers", e lá declaramos as condições e a ordem, e declaramos os parâmetros &DateFrom e &DateTo, variáveis que são usadas em duas das condições.

Esta definição centralizada nos permitirá reutilizá-la em todos os lugares que essa consulta for necessária, facilitando sua manutenção (se precisamos mudar algo na definição, é feito em um só lugar e automaticamente aplicado em todos os lugares em que é usado na KB).

Vamos ver como, uma vez definido o data selector, nós usáremos nos exemplos mencionados.

Uso

Em cláusula de comando For Each

```
Print Title
For each Customer
  Using ActiveCustomers(&DateFrom, &DateTo)
  Print Customer
Endfor
```

Parm(in: &DateFrom, in: &DateTo);

Active Customers list			
Id	Last name	First name	Added date
1	Parker	Susan	06/03/16
3	Jones	Ann	06/01/16
4	Flynn	Richard	06/14/16

Active Customers list			
Id	Last name	First name	Added date
1	Parker	Susan	06/03/16
3	Jones	Ann	06/01/16
4	Flynn	Richard	06/14/16

O utilizamos através da cláusula **using**. Seu comportamento é o mesmo que a especificação anterior. Aqui vemos o caso da primeira lista.

Para o comando For Each, podemos usar o Data Selector, executando-o como uma consulta que é independente do banco de dados. Não falaremos sobre isso neste curso, mas é importante notar que o operador **in** é usado neste caso. Por exemplo, se adicionamos o país dos clientes, e queremos listar os países que têm os clientes ativos que foram inseridos no sistema entre duas datas determinadas, entraríamos este código:

```
For each Country
Where CountryId in ActiveCustomers( &DateFrom, &DateTo)
...
endfor
```

Aqui temos duas consultas ao banco de dados: uma do Data Selector, que irá retornar o conjunto de clientes ativos que foram inseridos entre as duas datas indicadas e seus países correspondentes. A outra, correspondente ao Comando For Each, filtrará os países incluídos no conjunto.

Você pode encontrar mais documentação sobre [Data selectors em comandos For Each](http://wiki.genexus.com/commwiki/servlet/wiki?5312,Data+Selectors+in+For+Each+command) em nosso wiki <http://wiki.genexus.com/commwiki/servlet/wiki?5312,Data+Selectors+in+For+Each+command>

Em Fórmula

```
Print Title
For each Invoice
  Unique CustomerId
  &Qty = Count(InvoiceDate, using ActiveCustomers(&DateFrom, &DateTo))
  Print Customer
Endfor
```

Active customers and invoices

From: 06/01/16 to: 06/12/16

<u>Id</u>	<u>Last name</u>	<u>First name</u>	<u>Quantity</u>
1	Parker	Susan	3
2	Smith	Peter	0
4	Flynn	Richard	0

Aqui podemos ver o caso da segunda lista, onde estamos usando o data selector dentro da fórmula Count. Lembre-se que o segundo parâmetro de uma fórmula aggregate é para escrever as condições que devem ser atendidas pelos registros para serem "agregados".

Não mostramos o exemplo de web panel aqui porque não estudamos este objeto. Quando estudarmos, vamos explicar onde usar o Data Selector para filtrar os dados que serão mostrados em um grid.

Sintaxe do comando ForEach

```

For each   BaseTransaction
    order att1, att2, ... , attn [when condition]
    order att1, att2, ... , attn [when condition]
    unique att1, att2, ... , attn
    using DataSelector(parm1, parm2, ... , parmn)
    where condition [when condition]
    where condition [when condition]
    where att IN DataSelector(parm1, parm2, ... , parmn)

    main code

When none
    ...

Endfor

```

Um Data Selector específico, baseado nos parâmetros recebidos, um conjunto de condições e ordens para os dados de forma centralizada, a fim de evitar ter que repetir cláusulas Order, Where e Defined by em todos os lugares em que são necessárias.

Quando indicamos ao comando For Each para usar (using) um Data Selector, dizemos para adicionar suas ordens e filtros para aquele comando For Each. Por esta razão, os atributos que estão incluídos no Data Selector terão que pertencer à tabela estendida da tabela base do comando For Each.

A outra possibilidade que tínhamos mencionado, que implicou no uso do operador para filtrar em uma cláusula Where, é mostrada na sintaxe, mas foi deixado de fora dessa explicação. Se você quiser, você pode ler sobre isso no seguinte link: <http://wiki.genexus.com/commwiki/servlet/wiki?5312,Data+Selectors+in+For+Each+command>.

Escopo dos Data Selectors

Data Selectors podem ser usados em :

- ✓ **Grids**
 - Web panels: padrão e freestyle grids
 - Panels for Smart Devices e Work With for Smart Devices
- ✓ **For eachs**
 - Using
 - in
- ✓ **Grupos de Data Providers**
- ✓ **Fórmulas**

O Data Selector é um objeto para armazenar um conjunto de parâmetros, condições, ordens e cláusulas defined by, para ser usado/invocado a partir de diferentes consultas e cálculos, e reutilizar as mesmas definições (navegação) várias vezes.

Então, onde um Data Selector pode ser usado? Em todos os lugares onde as consultas ao banco de dados são especificadas.

Até agora só conhecemos for eachs e fórmulas. Nos capítulos posteriores vamos estudar os Grids e Data Providers.

Mais documentação sobre Data selectors :

<http://wiki.genexus.com/commwiki/servlet/wiki?5271,Category%3AData+Selector+object>

GeneXus™

The power of doing.

Vídeos

training.genexus.com

Documentação

wiki.genexus.com

Certificações

training.genexus.com/certifications