# Data Selectors

GeneXus

GeneXus

## Data Selectors

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

| Domain: Status | |
|---|---|
| Name | **Status** |
| Description | **Status** |
| Nulls in Forms | Empty as Null |
| Class | Attribute |
| Module | Root Module |
| Qualified Name | Status |
| Object Visibility | Public |
| Type Definition | |
| Based on | (none) |
| Data Type | **Character** |
| Length | **1** |
| Enum Values | **Active, Active, A; OnHold, On Hold, O; Closed, Closed, C** |

CUSTOMERS REPORT

| ID | NAME | LAST NAME | ADDED DATE |
|---|---|---|---|
| 1 | John | Smith | 09/24/20 |
| 4 | Alisa | Stuart | 09/25/20 |
| 2 | Ann | Hones | 11/25/20 |

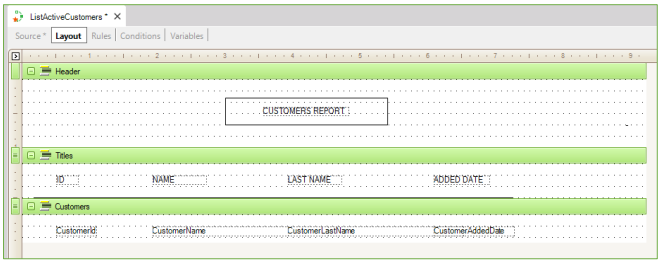
Suponhamos que na transação Customer de nossa aplicação, temos o atributo CustomerStatus para poder representar um dos três estados (active, on hold, closed) que pode ter um cliente no sistema da agência de viagens. Para isso, foi definido um novo tipo de dados enumerado.

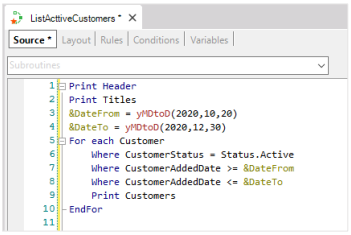
Suponhamos que em vários locais da aplicação precisamos trabalhar com os clientes ativos inseridos entre duas datas específicas.
Por exemplo:

Queremos gerar uma lista pdf que receba um intervalo de datas e mostre os clientes ativos que foram inseridos no sistema entre essas datas.

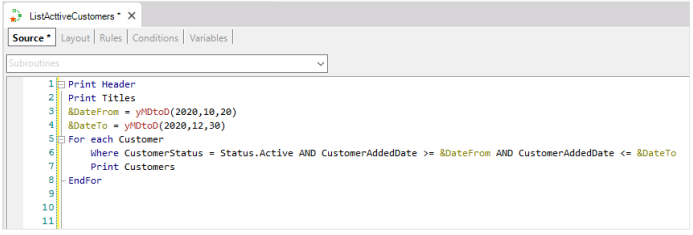
GeneXus

## Data Selectors




≈




Para isto, criamos um procedimento chamado ListActiveCustomers.

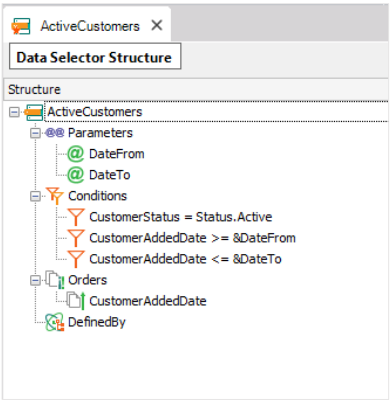
Se implementássemos as consultas com o que sabemos até o momento, seria através das seguintes três condições

Já que estas especificações vamos utilizar em vários lugares, para nos poupar o trabalho de ter que repeti-las em todos os lugares onde precisamos delas, podemos realizar essas definições em um único lugar, dando-lhes um nome, e a partir daí utilizar esse nome como referência. Esse lugar é o objeto Data Selector.

GeneXus

## Data Selectors

Structure:




Criamos um Data Selector ao que chamamos “ActiveCustomers“, e ali definimos as condições que acabamos de ver.
Observemos a estrutura que tem este objeto.

Em parameters, declaramos os parâmetros que receberá o Data Selector, que então serão utilizados em Conditions. Neste exemplo serão duas variáveis, &DateFrom e &DateTo.

Em Conditions são inseridas as condições que queremos se cumpram para filtrar os dados a serem recuperados. Neste caso indicamos que o status do cliente deverá ser ativo. E que a data de entrada do cliente deverá ser maior ou igual à variável DateFrom, e menor ou igual a DateTo.

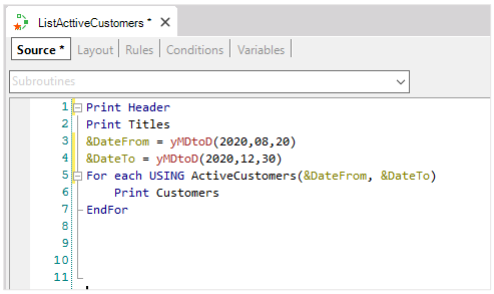
Em seguida, em Orders, poderemos indicar em que ordem queremos receber os dados que serão recuperados. Para este caso inserimos CustomerAddedDate, para ordenar os dados por data.

E, por último, temos Defined By, onde poderemos inserir um atributo ou lista de atributos, os quais colaboram para definir a tabela base final. Isto pode servir para resolver algum problema de ambiguidade na determinação da tabela base. Para este exemplo o deixamos vazio.

GeneXus

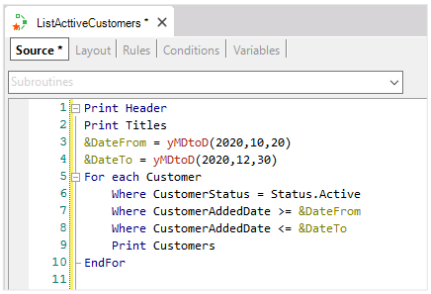
## “Using” clause

With Data Selectors

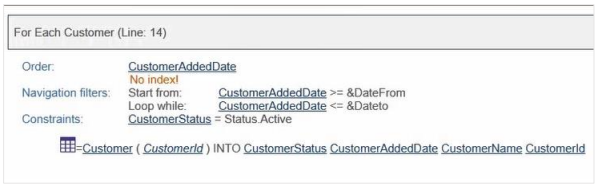



≈

Without Data Selector




Navigation View




Como dissemos, esta definição centralizada nos permitirá reutilizá-la em todos os lugares onde se necessita dessa consulta, facilitando a manutenção (se for necessário alterar algo da definição, realiza-se em um único lugar e aplica-se automaticamente a todos os lugares da KB onde se utiliza).

Um Data Selector especifica, com base nos parâmetros recebidos, um conjunto de condições e ordenações para a informação de maneira centralizada, de modo a não ter que repetir as cláusulas order, where e defined by em cada lugar em que se necessitem.

Vejamos como, uma vez definido o data selector, o utilizaríamos a partir de nosso procedimento que lista os clientes.

O utilizamos através da cláusula using. Aqui vemos como deveria ser utilizado para o exemplo apresentado.
Dentro deste for each fazemos um Using ao Data Selector ActiveCustomers, passando-lhe por parâmetros duas variáveis, que corresponderão às datas que queremos filtrar os dados.
O comportamento é em tudo análogo à especificação anterior.

mesmo na hora de determinar a tabela base, ou de resolver sua

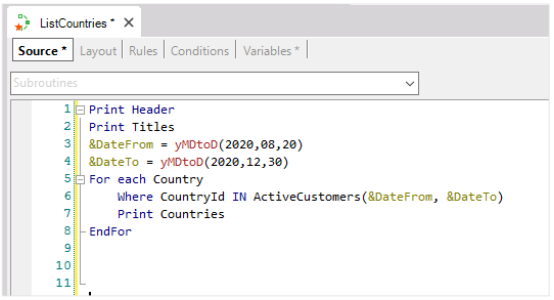
navegação, é igual a se, em vez do DataSelector, tivessem sido escritas diretamente suas cláusulas.
Podem ser adicionadas cláusulas order e cláusulas where ao for each, além do que já vem do Data Selector? A resposta é sim. As cláusulas se somam.

Na lista de navegação, vemos que será percorrida a tabela Customer, será classificada por Customer AddedDate, serão percorridos os registros de acordo com os filtros que aplicamos dentro de Conditions e, como restrição, filtrará apenas os clientes com status ativo.

"IN" operator

PDF Report:


ListActiveCustomers (Source)




Navigation View


Agora, suponhamos que aos clientes adicionamos o país de origem, e gostaríamos de listar os países que tenham clientes ativos e inseridos no sistema entre um par de datas dadas. Para resolver isto aproveitando nosso Data Selector faríamos o seguinte:

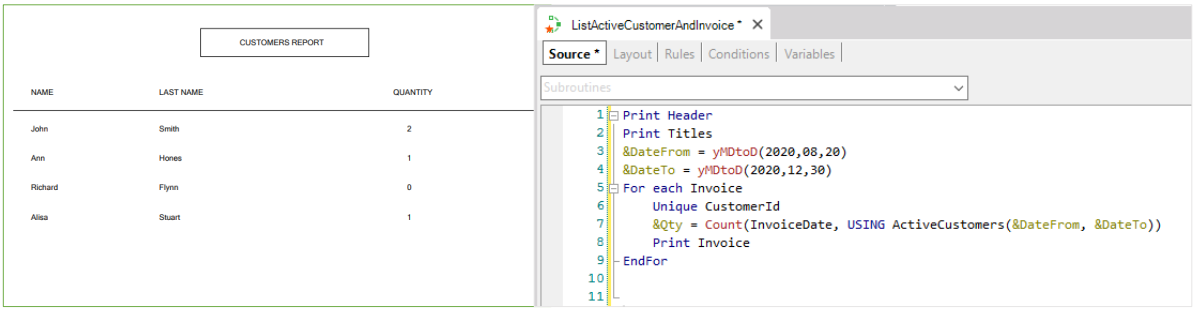
For each Country
Where CountryId in ActiveCustomers e lhe passamos por parâmetro as variáveis DateFrom e DateTo.
Neste caso, em vez de utilizar a cláusula Using, estamos utilizando o operador IN.
Desta forma utilizamos o Data Selector como se fosse uma consulta independente à base de dados.

Tendo aqui duas consultas, uma do Data Selector, que devolverá o conjunto de clientes ativos e inseridos entre o par de datas indicadas e seus respectivos países, e outra, a do for each, que filtrará os países que estejam dentro desse conjunto.

Na lista de navegação, vemos que é percorrida toda a tabela Country, ordenando por CountryId e, como restrição(Constraint), aparece o que declaramos dentro do where do for each, que nos interessa filtrar apenas os países que se encontrem dentro da lista de clientes do Data Selector.

GeneXus

Using Data Selectors in formula

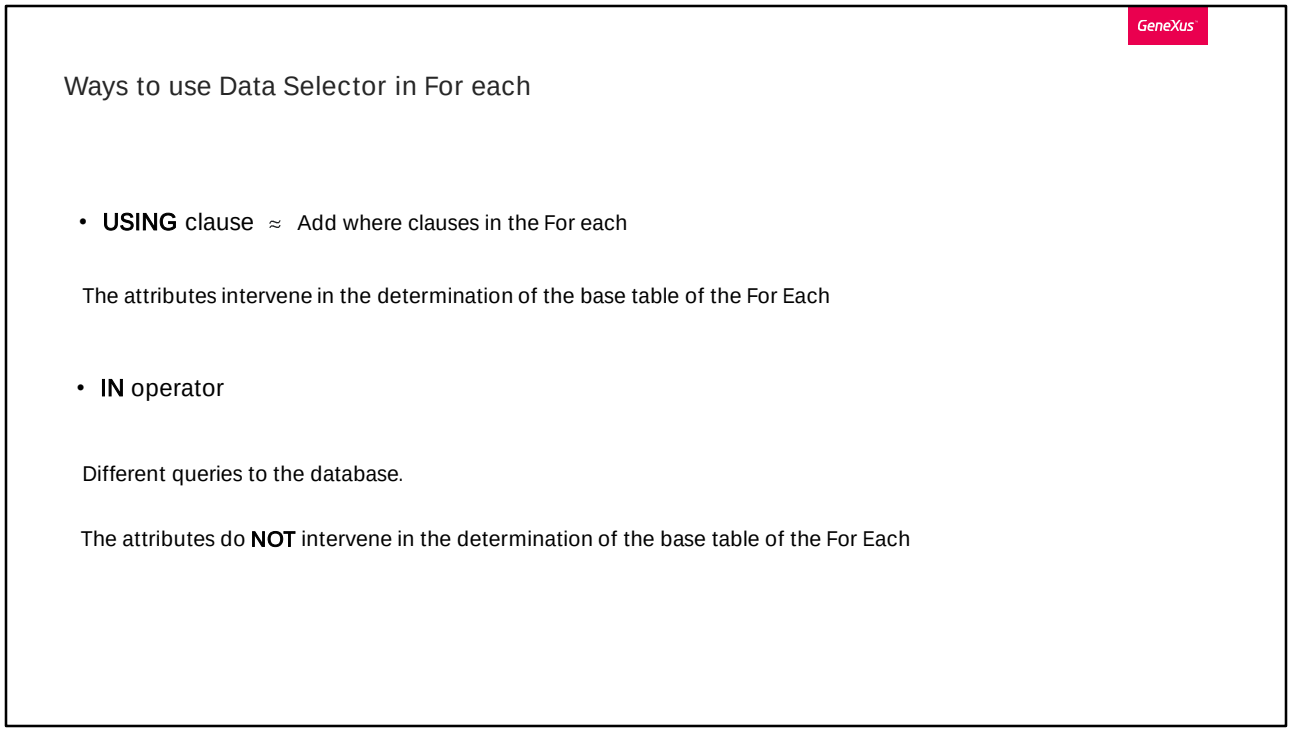



Apresentamos agora um terceiro exemplo onde podemos utilizar nosso Data Selector.
Para este caso, suponhamos que necessitamos mostrar em uma lista PDF, todos os clientes com faturas e sua quantidade, inserindo um intervalo de datas para poder contar unicamente as faturas dos clientes ativos inseridos no sistema entre essas datas. Se o cliente não está ativo ou foi inserido fora dessas datas, será listado, mas sua quantidade de faturas mostrará zero.

Aqui utilizamos o data seletor dentro de uma fórmula, especificamente dentro da fórmula Count.
Lembremos que o segundo parâmetro de uma fórmula aggregate, é para escrever as condições que deverão cumprir os registros para ser "agregados".

Se observamos atentamente este caso, estamos percorrendo no for each a tabela Invoice. Como cada cliente pode ter N faturas, colocamos a cláusula Unique para ficarmos com uma dessas faturas para o cliente, assim com a fórmula Count sobre a mesma tabela, contamos as faturas desse cliente, que também cumprem com as cláusulas do DataSelector: ou seja, se o cliente não está ativo, o Count dará 0, pois para todas as suas faturas o cliente não será um ativo. Se, por outro lado, está ativo, apenas

serão contadas as faturas cujo intervalo esteja entre o considerado.




Conceitualizando o que vimos até o momento, temos dois usos muito diferentes do Data Selector, dependendo de como ele é utilizado. Se for através da cláusula USING, não haverá nenhuma diferença em ter diretamente dentro do for each, as cláusulas order ou where do Data Selector. Ou como acabamos de ver, se o usamos dentro de uma fórmula aggregate, é análogo a inserir ali as condições que colocamos dentro de "Conditions" no Data Selector.

Como mencionamos, ao declarar a cláusula Using, estaremos dizendo que adicione as ordens e filtros do Data Selector aos do for each. Por este motivo, os atributos que compõem o Data Selector deverão pertencer à tabela estendida da tabela base do For each.

Em vez disso, se utilizamos o operador IN em uma cláusula where, por exemplo, estamos sim, estabelecendo uma consulta independente, de modo que esse Data Selector, para ser resolvido, terá que navegar sua tabela base. Como se fosse um for each independente.
O IN é utilizado quando se deseja realizar uma subconsulta, cujo resultado será utilizado em seguida como filtro no For each, como acabamos de ver.

Dependendo da forma como o invocamos, os atributos definidos dentro do Data Selector irão ou não intervir na determinação da tabela base do

for each:

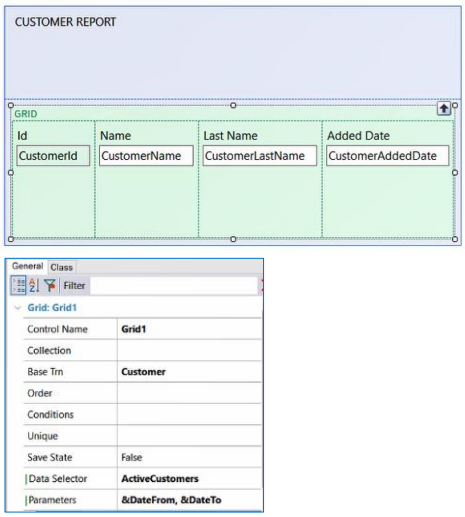
Se invocamos o Data Selector através da cláusula USING, os atributos declarados dentro deste objeto interferem na determinação da tabela base do for each de onde o estamos chamando.

Se o invocamos através da cláusula IN, os atributos do Data Selector não interferem na determinação da tabela base do for each.

GeneXus

## Data Selector In Web Panel

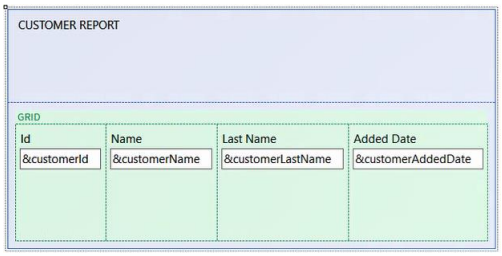
Web Panel with base table







Web Panel without base table




```
Event Start
      &DateFrom = yMDtoD(2020,08,20)
      &Dateto = yMDtoD(2020,12,30)
Endevent


Event Grid1.Load()
      For each USING ActiveCustomers(&DateFrom, &DateTo)
            &customerId = CustomerId
            &customerName = CustomerName
            &customerLastName = CustomerLastName
            &customerAddedDate = CustomerAddedDate
            Load
      Endfor
EndEvent
```

Os três exemplos que acabamos de apresentar, os fizemos em objetos procedimento, pois queríamos listar as informações em arquivos PDF. Mas se simplesmente quiséssemos exibir as informações em tela por meio de um Web Panel, como faríamos?.

Voltemos um momento sobre o primeiro exemplo, no qual nos interessava listar os clientes ativos, em um intervalo de datas determinado, utilizando para isto o Data Selector que criamos. Mas desta vez mostraremos os registros em tela com um Web Panel.

No Web Layout de nosso Web Panel, declaramos um grid e adicionamos os atributos que nos interessam listar.
Dentro das propriedades do Grid, vemos uma chamada Data Selector. Ali devemos inserir o nome do nosso Data Selector que queremos utilizar. Isto é análogo a quando em um objeto procedimento o invocávamos a partir do for each através da cláusula Using.

Executemos para testar. Vemos que nos lança um erro, pois, lembremos que o Data Selector ActiveCustomers recebia por parâmetro duas variáveis, aquelas que conterão as datas. Portanto, deveremos lhe passar esta informação.
Para isto criamos as variáveis DateFrom e DateTo do tipo Date. No evento Start as declaramos e para este teste as inicializamos com estas datas.

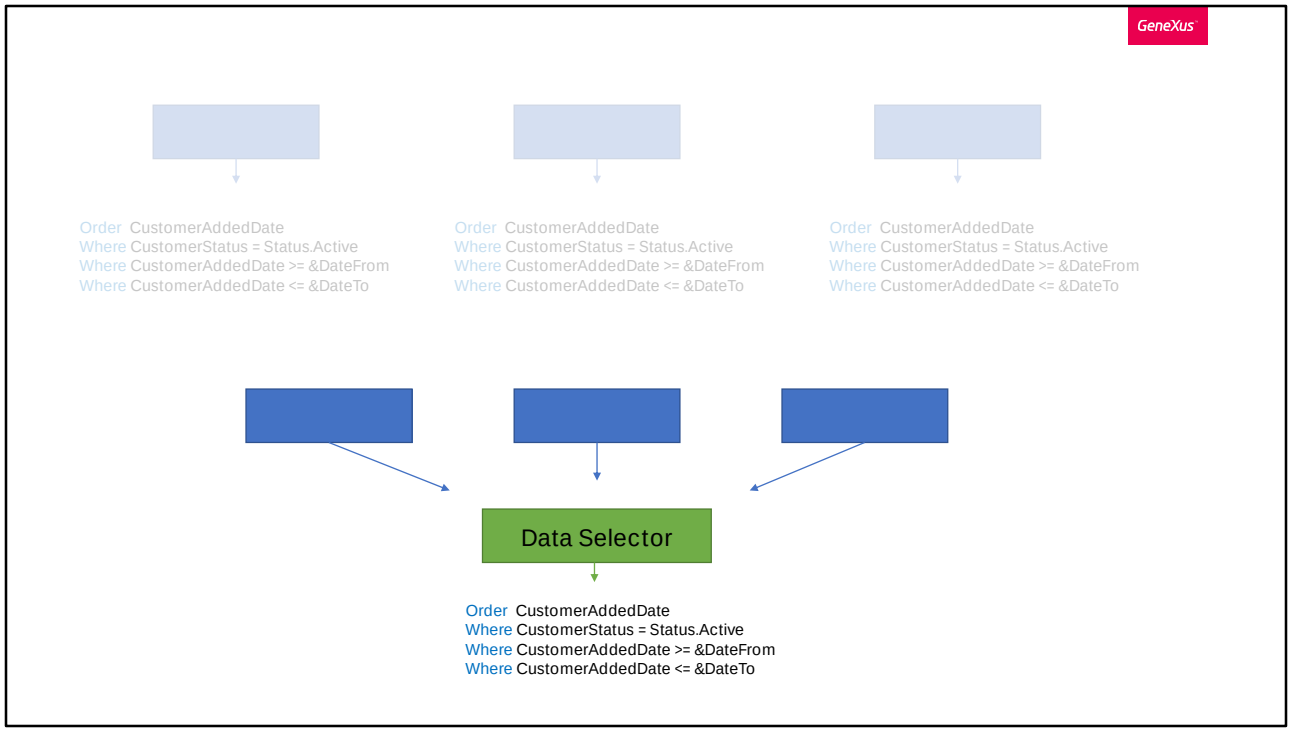
Para passar os parâmetros ao Data Selector, fazemos isso a partir da propriedade Parameters do grid.

Agora sim, voltemos a executar.
Vemos que são mostrados em tela os clientes, filtrando pelos ativos e compreendidos entre as datas indicadas. Ordenados por data de entrada, conforme definimos no Data Selector.

Suponhamos o mesmo exemplo, mas que o Web Panel em vez de ter atributos, tenha variáveis. Nesse caso, trata-se de um Web Panel sem tabela base, ao contrário do anterior. Por este motivo, aqui não nos servirá utilizar a propriedade Data Selector do grid, já que para percorrer todos os clientes deveremos fazer com um for each, e é ali de onde devemos chamar nosso Data Selector. Portanto, nestes casos, a declaração será similar à que vimos com o objeto procedimento.

Na seção eventos, devemos declarar o evento Load do grid. Dentro, um for each com a cláusula USING seguida do nome de nosso Data Selector, lhe passando por parâmetro as variáveis que este receberá para filtrar as datas.
E dentro do for each, às variáveis do grid atribuímos o valor dos atributos que nos interessa mostrar.
Por último, definimos o comando Load e fechamos o for each e o evento Load.

Executamos e vemos os resultados desejados.




Resumindo, o Data selector é um objeto que nos permite armazenar um conjunto de parâmetros, conditions, orders, para utilizá-lo/invocá-lo a partir de diferentes consultas e cálculos, e reutilizar a mesma navegação várias vezes.

Portanto, em que lugares poderemos utilizar um Data Selector? Em todos aqueles que especifiquem consultas à base de dados. Por exemplo, em grids de painéis e web panels ou em grupos de Data Providers.

Pode encontrar mais informações sobre Data selectors em nossa wiki.

GeneXus™

training.genexus.com
wiki.genexus.com