

**Up to date
development with**

GeneXusTM 15

Octubre 2017

Copyright 1988 - 2017 | GeneXus S.A.

Todos los derechos reservados. Este documento no puede ser reproducido en cualquier medio sin el consentimiento explícito de GeneXus S.A. La información contenida en este documento es para uso personal únicamente.

Marcas Registradas

GeneXus es una marca registrada (®) en múltiples países y regiones, incluyendo entre otros, América Latina, EEUU, Japón, la UE y Uruguay. Todas las demás marcas mencionadas en este documento son propiedad de sus respectivos dueños.

Objetivo

GeneXus ha crecido y evolucionado en el correr del tiempo, ofreciendo nuevas e interesantes funcionalidades. La idea es aprovechar al máximo todo lo que nos provee. No programar “a la vieja usanza” sino adoptar las nuevas técnicas y resolver nuestras necesidades de la mejor manera (la cual una vez conocida, es más fácil).

A continuación se proponen ejercicios para resolver con GeneXus 15, con el fin de aplicar las alternativas de solución actuales para aprovechar su potencia y ventajas.

La aplicación sobre la que trabajaremos permite gestionar la venta de entradas a distintos espectáculos que se presentan en distintos lugares de distintos países. La misma tiene un módulo Web y un pequeño módulo para Smart Devices.

Comencemos

Utilizaremos **GeneXus 15 U6**, el browser **Chrome** y el Emulador de Android SDK.

El ícono asociado a **GeneXus** se encuentra en el escritorio.

Comencemos por abrir GeneXus y seleccionar:

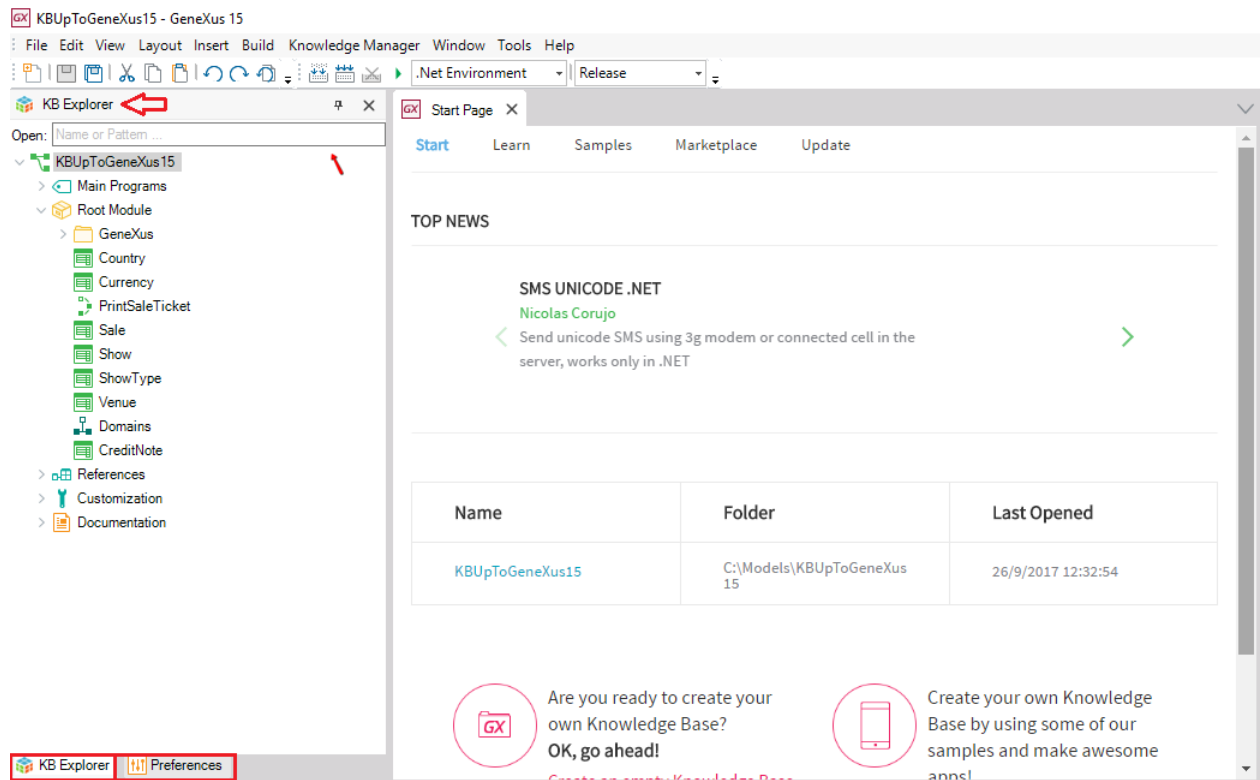
File / New / Knowledge Base from GeneXus Server

Mediante esta opción crearemos un KB local desde:

- el server: **<http://samples.genexusserver.com/v15>**
- nombre de KB: **KBUpToDateGeneXus15**

Observación del IDE de GeneXus 15

Observar el IDE. La sección de nombre **KB Explorer** sustituye a la que en el pasado se llamaba Folder View. Observar que se agregó un buscador de objetos, “**Open:**”, que permite además de encontrar rápidamente los objetos buscados, abrirlos directamente desde allí mismo cuando se encuentran.



Transacciones definidas en la KB

País (Country)

De cada país se registra su identificador único y el nombre.

Moneda (Currency)

De cada moneda se registra su identificador único (3 caracteres que corresponden al código universal) y el nombre.

Tipo de espectáculo (ShowType)

De cada tipo de espectáculo se registra su identificador único y el nombre (concierto, obra de teatro, circo, etc.).

Lugar (Venue)

Cada posible lugar para desarrollar un espectáculo se registra con un identificador único, un nombre, una foto (de tipo Image), país donde se encuentra y su correspondiente punto geográfico (de tipo Geography).

Espectáculo (Show)

Cada espectáculo se registra con un identificador único, un nombre, el tipo de espectáculo (concierto, obra de teatro, circo, etc.), la fecha, el lugar donde se realizará y una imagen representativa. También cada show tiene asociada una moneda de venta.

Cada espectáculo tiene distintos sectores habilitados para el mismo, cada uno de los cuales tiene una cantidad de localidades totales y un precio. Ej: platea alta - 300 localidades - \$500.

Los sectores correspondientes a un mismo espectáculo se numeran automáticamente y correlativamente. La cantidad de localidades vendidas y las que aún quedan disponibles en cada sector, se conocen siempre en atributos fórmula.

Venta (Sale)

Toda venta de una entrada se registra con un identificador único, la fecha de la venta (sugerida automáticamente por el sistema), los datos del espectáculo y una localidad comprada. El monto de la venta se deja almacenada propiamente en la venta. Los datos de cada venta efectuada se imprimen como ticket de entrada o pase para el espectáculo.

Nota de crédito (CreditNote)

Cuando se requiere devolver/cancelar una venta, es necesario crear una nota de crédito. Toda nota de crédito tiene un identificador único, una fecha, un identificador de venta.

Observación:

Las transacciones tienen algunas reglas definidas. Observe que en GeneXus 15, los tabs de acceso a las diferentes secciones de una transacción, se encuentran arriba:

```
1 error("The show name can not be empty")
2   if ShowName.IsEmpty();
3
4 error("The show date can not be previous than today")
5   if ShowDate<&Today;
6
7 serial(SectorId,ShowLastSectorId,1);
8 noaccept(SectorId);
```

Tómese unos minutos para revisar las reglas definidas en las diferentes transacciones de la base de conocimiento para familiarizarse con la aplicación.

Funcionalidades requeridas

- 1) Resolver el poblado de las tablas físicas indicadas con los datos indicados.

Currency

UYU, Uruguayan Peso
ARS, Argentine Peso
BRL, Brazilian Real
USD, US Dolar
EUR, Euro

Country

Uruguay
Argentina
Mexico
United States
Japan

ShowType

Concert
Musical Comedy
Circus
Opera
Ballet

Venue

- 1, Estadio Centenario, -56.152690 -34.894511, Uruguay, Imagen
- 2, Teatro Solis, -56.201040 -34.907680, Uruguay, Imagen
- 3, Sodre, -56.198283 -34.904363, Uruguay, Imagen
- 4, Luna Park, -58.368818 -34.602498, Argentina, Imagen
- 5, Auditorio Nacional, -99.194818 19.424795, Mexico, Imagen
- 6, Madison Square Garden, -73.993267 40.750525, United States, Imagen

Guía para resolver el poblado.

Para cada una de las transacciones: Currency, Country, ShowType y Venue, hay que configurar sus propiedades (se encuentran agrupadas bajo el grupo Data):

- Data Provider = True
- Used To = Populate data

Por haber configurado la propiedad Data Provider = True de cada una de las cuatro transacciones anteriores, se podrá visualizar en el KB Explorer:

- Bajo la transacción Currency, el objeto Currency_DataProvider creado.
- Bajo la transacción Country, el objeto Country_DataProvider creado.
- Bajo la transacción ShowType, el objeto ShowType_DataProvider creado.
- Bajo la transacción Venue, el objeto Venue_DataProvider creado.

Tan sólo restará abrir cada uno de los Data Providers que se crearon automáticamente y completarlos con los datos a cargar en las respectivas tablas físicas:

Currency_DataProvider X

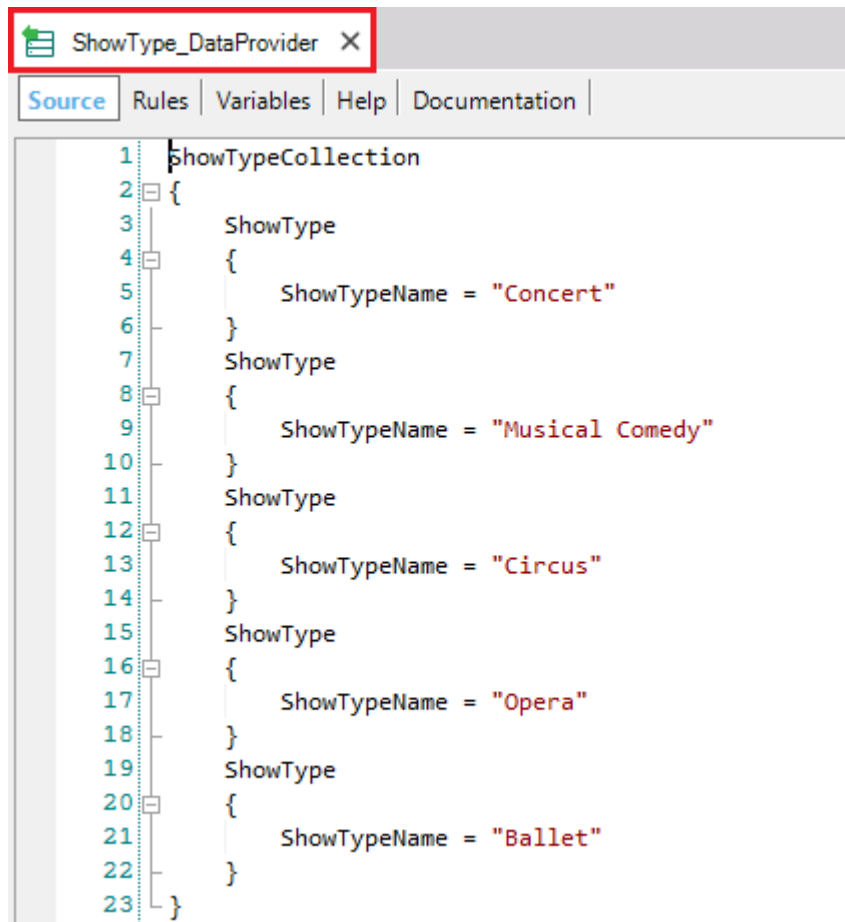
Source Rules Variables Help Documentation

```
1 CurrencyCollection
2 {
3     Currency
4     {
5         CurrencyCode = "UYU"
6         CurrencyName = "Uruguayan Peso"
7     }
8     Currency
9     {
10        CurrencyCode = "ARS"
11        CurrencyName = "Argentine Peso"
12    }
13    Currency
14    {
15        CurrencyCode = "BRL"
16        CurrencyName = "Brazilian Real"
17    }
18    Currency
19    {
20        CurrencyCode = "USD"
21        CurrencyName = "US Dollar"
22    }
23    Currency
24    {
25        CurrencyCode = "EUR"
26        CurrencyName = "Euro"
27    }
28 }
29
```




```
1 CountryCollection
2 {
3     Country
4     {
5         CountryName = "Uruguay"
6     }
7     Country
8     {
9         CountryName = "Argentina"
10    }
11    Country
12    {
13        CountryName = "Mexico"
14    }
15    Country
16    {
17        CountryName = "United States"
18    }
19    Country
20    {
21        CountryName = "Japan"
22    }
23 }
```

Nota: Se ha decidido que los países se autonumeren (CountryId tiene Autonumber=True) y es razonable definir un índice único para la tabla Country por el atributo CountryName. Defina ese índice único para controlar que cada nombre de país no se pueda repetir y a la vez ya sirve para evitar almacenamientos de nombres de países repetidos con diferente identificador al ejecutarse el poblado en varias oportunidades.



```
1 ShowTypeCollection
2 {
3   ShowType
4   {
5     ShowTypeName = "Concert"
6   }
7   ShowType
8   {
9     ShowTypeName = "Musical Comedy"
10  }
11  ShowType
12  {
13    ShowTypeName = "Circus"
14  }
15  ShowType
16  {
17    ShowTypeName = "Opera"
18  }
19  ShowType
20  {
21    ShowTypeName = "Ballet"
22  }
23 }
```

Nota: Se ha decidido que los tipos de shows se autonumeren (ShowTypeId tiene Autounumber=True) y es razonable definir un índice único para la tabla ShowType por el atributo ShowTypeName. Defina ese índice único para controlar que cada nombre de tipo de show no se pueda repetir y a la vez ya sirve para evitar almacenamientos de nombres de tipos de shows repetidos con diferente identificador al ejecutarse el poblado en varias oportunidades.

```
1 VenueCollection
2 {
3     Venue
4     {
5         VenueId = 1
6         VenueName = "Estadio Centenario"
7         CountryId = Find(CountryId, CountryName="Uruguay")
8         VenueGeography = Geography.FromString("POINT(-56.152690 -34.894511)")
9         VenueImage = Centenario.Link()
10    }
11    Venue
12    {
13        VenueId = 2
14        VenueName = "Teatro Solis"
15        CountryId = Find(CountryId, CountryName="Uruguay")
16        VenueGeography = Geography.FromString("POINT(-56.201040 -34.907680)")
17        VenueImage = Solis.Link()
18    }
19    Venue
20    {
21        VenueId = 3
22        VenueName = "Sodre"
23        CountryId = Find(CountryId, CountryName="Uruguay")
24        VenueGeography = Geography.FromString("POINT(-56.198283 -34.904363)")
25        VenueImage = Sodre.Link()
26    }
27    Venue
28    {
29        VenueId = 4
30        VenueName = "Luna Park"
31        CountryId = Find(CountryId, CountryName="Argentina")
32        VenueGeography = Geography.FromString("POINT(-58.368818 -34.602498)")
33        VenueImage = LunaPark.Link()
34    }
35    Venue
36    {
37        VenueId = 5
38        VenueName = "Auditorio Nacional"
39        CountryId = Find(CountryId, CountryName="Mexico")
40        VenueGeography = Geography.FromString("POINT(-99.194818 19.424795)")
41        VenueImage = AuditorioNacional.Link()
42    }
43    Venue
44    {
45        VenueId = 6
46        VenueName = "Madison Square Garden"
47        CountryId = Find(CountryId, CountryName="United States")
48        VenueGeography = Geography.FromString("POINT(-73.993267 40.750725)")
49        VenueImage = MadisonSquare.Link()
50    }
51 }
```

Nota: Las imágenes a ser asignadas al atributo VenueImage (Centenario, Solis, Sodre, LunaPark, AuditorioNacionalMexico, MadisonSquare) se encuentran incluidas dentro de la KB. Si desea verificarlo, puede seleccionar la opción **View / Images** en el menú principal del IDE de GeneXus.

Presione F5.

La siguiente imagen muestra los datos a completar para crear la base de datos local:

Set required properties in Build process

Set required properties in Build process

The following properties must be set to start the Build process. Please fill them with appropriate values.

Database name	DBUpToGeneXus15
Server name	localhost
Use trusted connection	No
User id	gx27
User password	••••

Edit connection Cancel < Back Next > Finish

User id: gx27

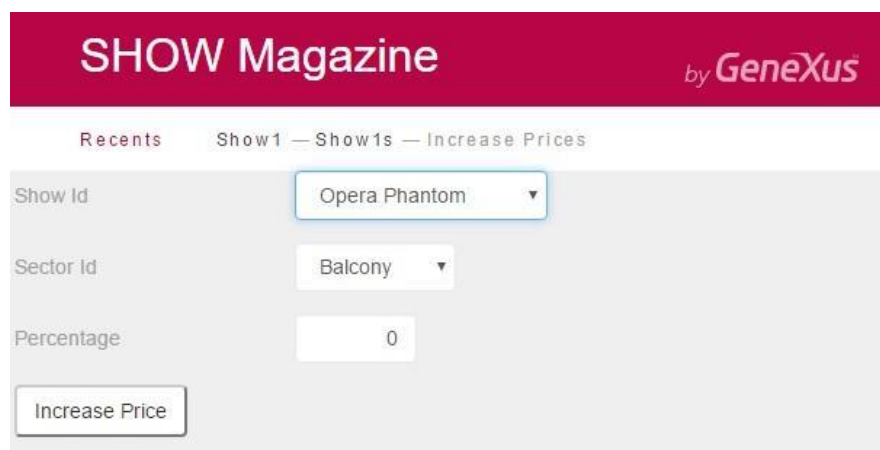
User password: gx27

2) La gerencia de la compañía solicita que la aplicación permita emitir un listado como el que se muestra en la imagen. Es decir, un producto cartesiano entre Country y Currency para analizar, decidir y marcar cuáles países aceptan cuáles monedas de pago.

Uruguay	Argentine Peso	<input type="checkbox"/>
Uruguay	Brazilian Real	<input type="checkbox"/>
Uruguay	Euro	<input type="checkbox"/>
Uruguay	US Dolar	<input type="checkbox"/>
Uruguay	Uruguayan Peso	<input type="checkbox"/>
Argentina	Argentine Peso	<input type="checkbox"/>
Argentina	Brazilian Real	<input type="checkbox"/>
Argentina	Euro	<input type="checkbox"/>
Argentina	US Dolar	<input type="checkbox"/>
Argentina	Uruguayan Peso	<input type="checkbox"/>
Mexico	Argentine Peso	<input type="checkbox"/>
Mexico	Brazilian Real	<input type="checkbox"/>
Mexico	Euro	<input type="checkbox"/>
Mexico	US Dolar	<input type="checkbox"/>
Mexico	Uruguayan Peso	<input type="checkbox"/>
United States	Argentine Peso	<input type="checkbox"/>
United States	Brazilian Real	<input type="checkbox"/>
United States	Euro	<input type="checkbox"/>
United States	US Dolar	<input type="checkbox"/>
United States	Uruguayan Peso	<input type="checkbox"/>
Japan	Argentine Peso	<input type="checkbox"/>
Japan	Brazilian Real	<input type="checkbox"/>
Japan	Euro	<input type="checkbox"/>
Japan	US Dolar	<input type="checkbox"/>
Japan	Uruguayan Peso	<input type="checkbox"/>

Nota: Recuerde que es posible referenciar más de una transacción base para resolver con eficiencia esta solicitud.

- 3) Se requiere que la aplicación permita aumentar el precio de un determinado sector de un espectáculo. Para ello se desea contar con una página que permita seleccionar un espectáculo y un determinado sector habilitado para dicho espectáculo y que el usuario pueda ingresar un porcentaje. El usuario debe contar con un botón para que al presionarlo se procese el aumento del precio de ese sector del espectáculo, aplicando el porcentaje de aumento solicitado.



Notas:

- Observe que el segundo Dynamic Combo Box requiere que se le defina una condición ya que los sectores a mostrar deben corresponder al Show elegido previamente.
- Recuerde que GeneXus 15 ofrece el método GetByKey para recuperar un registro puntual correspondiente a una línea (previamente se debe haber cargado la estructura completa en memoria con Load). Se solicita resolver la actualización del precio del sector haciendo uso de dicho método:

Event 'Increase Price'

```
&Show.Load(&ShowId)
&OneSector=&Show.Sector.GetByKey(&SectorId)
&OneSector.SectorPrice = &OneSector.SectorPrice * (1 + &Percentage/100)
&Show.Save()
commit
```

Endevent

4) Transacciones dinámicas.

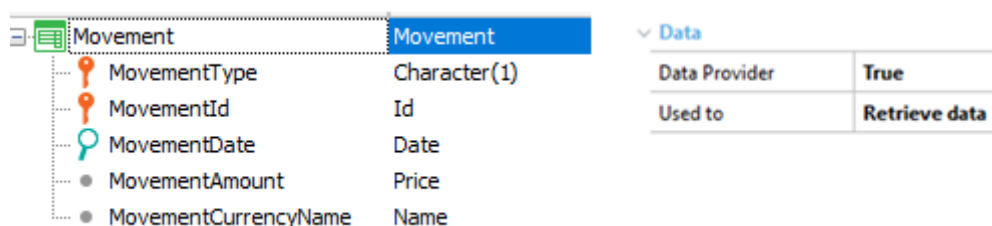
Las transacciones en GeneXus 15 cuentan con nuevas propiedades (bajo el grupo Data) para ser definidas como Transacciones Dinámicas. La particularidad de las Transacciones Dinámicas es que no provocarán la creación de tablas físicas asociadas a las mismas, sino vistas para recuperar los datos en tiempo de ejecución a partir de lo que definamos en el Data Provider asociado a la transacción. Esto nos permite resolver muchos y muy variados escenarios de consultas.

4.1) Definición de Transacción Dinámica: Movement

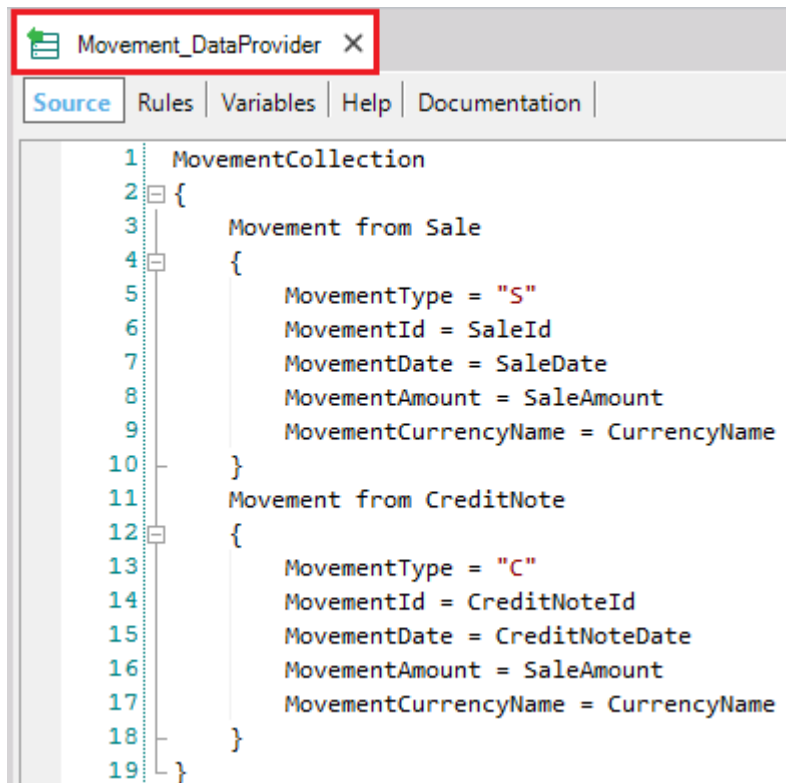
Se desea resolver la unión de todas las ventas (Sale) y de todas las notas de crédito (CreditNote) para contar con todos los movimientos juntos y poder trabajar con los mismos.

Guía:

- Cree una transacción de nombre Movement con los atributos que se muestran en la imagen y configure las propiedades necesarias para que sea una transacción dinámica.



- Complete el Movement_DataProvider, cargando en los atributos de Movement los datos de todas la ventas (Sale) y notas de crédito.



Web - Usando la transacción dinámica Movement

Ahora simplemente utilizaremos los atributos de la transacción dinámica Movement como lo solemos hacer habitualmente.

- Cree un listado que muestre los movimientos realizados en el día de hoy.
- Aplique el patrón WW for Web a la transacción Movement.
- Incluya acción en el WW for Web (fuera del grid) que invoque al listado.

SD- Usando la transacción dinámica Movement

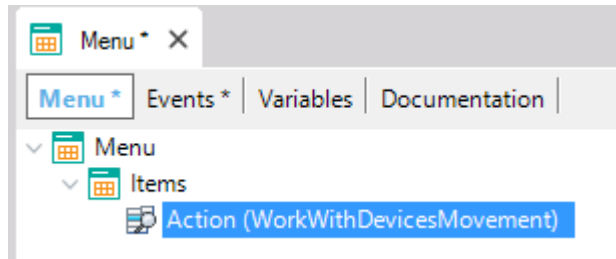
- Aplique el patrón WW for Smart Devices a la transacción Movement.
- Agregue en el grid del nodo List, los atributos de Movement que no estén por defecto.

Nota: Para que los atributos dentro del grid no tengan etiqueta, puede configurar la property **Label Position** de cada atributo incluido en el grid con valor = **None**.

- Cree un objeto Menu for Smart Devices (recuerde que el objeto llamado **Dashboard**

en versiones anteriores cambió de nombre y actualmente se llama: **Menu for Smart Devices**).

Agregue bajo el nodo **Items** una **Action** que invoque al objeto → **WorkWithDevicesMovement**



Ejecute Run para el **Menu for Smart Devices** que ha definido.

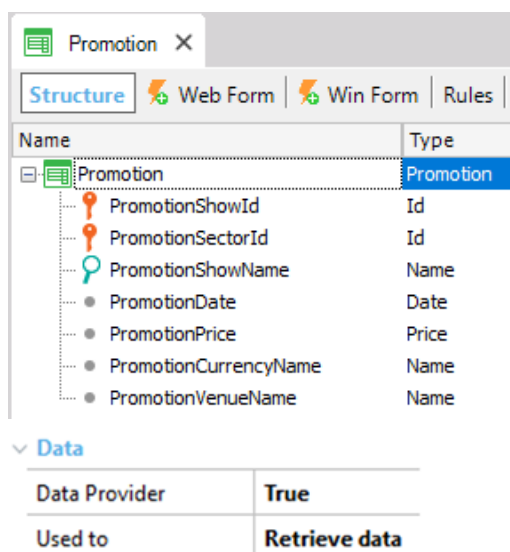
4.2) Definición de Transacción Dinámica: Promotion

La empresa define promociones de la siguiente manera: Aquellos shows cuya fecha de presentación es para dentro de 5 días o menos, venderán sus localidades con 50% de descuento con respecto al precio original particularmente para aquellos sectores que tengan 100 o más localidades disponibles al.

Defina una transacción dinámica para modelar las promociones en base a lo descripto.

Guía:

- Cree una transacción de nombre Promotion con los atributos que se muestran en la imagen y configure las propiedades necesarias para que sea una transacción dinámica.



- Complete el Promotion_DataProvider, cargando en los atributos de Promotion los datos de todas los sectores de shows que están en promoción al momento de realizar la consulta:

```

1 PromotionCollection
2 {
3     Promotion from Show.Sector
4         where (ShowDate-ServerDate()) <= 5 and (ShowDate-ServerDate()) >=0
5         where SectorAvailableQuantity >=100
6     {
7         PromotionShowId = ShowId
8         PromotionSectorId = SectorId
9         PromotionShowName = ShowName
10        PromotionDate = ShowDate
11        PromotionPrice = SectorPrice/2
12        PromotionCurrencyName = CurrencyName
13        PromotionVenueName = VenueName
14    }
15 }

```

Web - Usando la transacción dinámica Promotion

Ahora simplemente utilizaremos los atributos de la transacción dinámica Promotion como lo solemos hacer habitualmente. En particular le pedimos definir un web panel que muestre los datos de las promociones vigentes.


50 %

Promotion Date	Promotion Show Id	Promotion Show Name	Promotion Sector Id	Sector Name	Promotion Venue Name	Promotion Price	Promotion Currency Name
10/03/17	1	Madonna in Concert	1	Orchestra Platinum	Madison Square Garden	100	US Dollar
10/03/17	1	Madonna in Concert	2	Orchestra Gold	Madison Square Garden	75	US Dollar
10/06/17	2	David Bisbal - Tour 2017	1	America Rosrtrum	Estadio Centenario	3500	Uruguayan Peso

¡Disfrute GeneXus 15 y el curso!