GUÍA DE INFORMES PERSONALIZADOS

Ágora Software TPV

Lista de Cambios

Versión 5.1.2

- Versión inicial de este documento.

Versión 7.2.0

- Se introduce el campo `showInAdmin` en el formato de la consulta.

Introducción

Los informes personalizados permiten crear nuevos informes en Ágora adaptados a necesidades específicas de clientes concretos.

Estos informes no forman parte de la versión oficial de Ágora. Son creados y mantenidos de forma externa, por lo que es posible desplegarlos de forma independiente a Ágora.

Utilizando ficheros XML podrá definir de forma sencilla los parámetros del informe, la sentencia SQL utilizada para obtener los resultados, y la manera en que se presentarán estos resultados al usuario.

Puesto que los informes ejecutan directamente sentencias SQL contra la base de datos de Ágora, son dependientes de la versión de Ágora y es posible que sea necesario realizar ajustes sobre los mismos en caso de actualizar Ágora.

No existe una documentación específica sobre el esquema de base de datos de Ágora. El propio esquema es autoexplicativo, con una nomenclatura clara de tablas y columnas. Usando cualquier herramienta de gestión de base de datos, por ejemplo Microsoft SQL Server Management Studio Express, es fácil comprender la estructura.

Formato

Cada informe se almacena en su propio fichero XML, el cual tiene la siguiente estructura:

```
<?xml version="1.0" encoding="utf-8" ?>
<Query name='Productos del día' group='Catálogo'
       guid='{E71A1FD9-218D-4FD3-ADA1-898375B61C2A}'
           showInAdmin='false'>
  <Description>
    <!-- Descripción/Texto de ayuda de la consulta -->
  </Description>
  <Params>
    <!-- Parámetros de la consulta -->
  </Params>
  <Sql>
    <!-- Consulta SQL -->
  </Sql>
  <Columns>
    <!-- Columnas a mostrar -->
  </Columns>
  <RowStyles>
       <!-- Estilos de resaltado de filas -->
  </RowStyles>
</Query>
```

En el elemento `Query` aparecerá la información general del informe:

name
: Nombre del informe. Es el texto que se mostrará en la sección de informes de la administración

group
: Grupo de informes donde se insertará el nuevo informe. Si ya existe un grupo con ese nombre (por ejemplo "Catálogo") en la sección de informes de la administración, el informe se añadirá a ese grupo. Si no existe, se creará un grupo nuevo para contener el informe.

guid
: Identificador único del informe. Permite identificar cada informe de forma unívoca dentro de la aplicación.

showInAdmin
: Permite indicar si el informe es visible para el cliente en la administración. Puede tomar los valores `true` o `false`. Opcional. Por defecto, se mostrará el informe.

El elemento `Description` es opcional y le permite definir un texto de ayuda que se mostrará antes que los elementos de la interfaz de usuario de los parámetros de la consulta.

Para completar la definición del informe deberá indicar los parámetros, la sentencia SQL a ejecutar, y la forma en que se mostrarán los resultados al usuario. En las siguientes secciones de este documento se explica cómo se define cada una de estas partes y cómo se relacionan entre ellas.

Parámetros

En general, la mayoría de informes permiten algún tipo de parametrización por parte del usuario. Por ejemplo, seleccionar un rango de fechas, una familia de productos o una tarifa. En los informes personalizados podrá definir el formato de estos parámetros y Ágora se encargará de generar los elementos de interfaz de usuario necesarios para poder capturar los valores.

```
<?xml version="1.0" encoding="utf-8" ?>
<Query name='Productos del día' group='Catálogo'
       guid='{E71A1FD9-218D-4FD3-ADA1-898375B61C2A}'>
  <Params>
    <Param name='businessDay' label='Fecha:'
           type='date' required='true' />
    <Param name='posGroupIds' label='Grupo TPVs:'
           type='posGroup' required='true' multipleValue='true' />
  </Params>
  <Sql>
    <!-- Consulta SQL -->
  </Sql>
  <Columns>
    <!-- Columnas a mostrar -->
  </Columns>
  <RowStyles>
    <!-- Estilos de resaltado de filas -->
  </RowStyles>
</Query>
```

En el elemento `Params` se incluirán los parámetros del informe en sucesivos elementos `Param`. Ágora generará automáticamente los elementos de interfaz de usuario necesarios para elegir los valores de los parámetros. Para cada elemento `Param` deberá indicar:

name
: Nombre del parámetro.

label
: Texto que se mostrará en el interfaz de usuario para identificar el parámetro.

type

Tipo del parámetro. En función del tipo, Ágora generará distintos elementos de interfaz de usuario y podrá ofrecer al usuario valores para seleccionar dependientes del mismo. Los tipos soportados son:

- **Date**: fecha.
- **Int**: número entero.
- **String**: texto.
- **Decimal**: número decimal.
- **Bool**: valor booleano (`true`/`false`).
- **Family**: familia de productos.
- **Category**: categoría de productos.
- **Supplier**: proveedor.
- **PosGroup**: grupo de TPVs.
- **Warehouse**: almacén.
- **PriceList**: tarifa.
- **Product**: producto.
- **SaleFormat**: formato de venta.
- **PreparationType**: tipo de preparación.
- **PreparationOrder**: orden de preparación.
- **PaymentMethod**: forma de pago.
- **SaleCenter**: centro de venta.
- **Offer**: promoción.
- **NamedDiscount**: descuento predefinido.
- **User**: usuario.
- **TimeFrameGroup**: grupo de periodos de servicio.
- **Workplace**: local.
- **Values**: valor personalizado.

required

Indica si el parámetro es obligatorio (`true`) o no (`false`). Opcional, por defecto es `false`.

multipleValue

Indica si es posible seleccionar más de un valor para el parámetro. Opcional, por defecto es `false`. Por ejemplo, si el parámetro es de tipo `Family`, se permitiría seleccionar más de una familia.

Estos parámetros serán convertidos a parámetros de la consulta SQL. Cada uno de ellos llegará a la consulta con el formato `@name`, donde `name` es el nombre establecido para el parámetro.

En el caso de tipos simples (`Date`, `Int`, `String`, `Decimal` y `Bool`), el tipo de parámetro en la consulta SQL será una traducción directa del tipo indicado en Ágora (`DateTime`, `integer`, `nvarchar`, `decimal` y `bit`, respectivamente).

En el caso de tipos complejos (`Family`, `Category`, `Supplier`, `PosGroup`, `Warehouse`, `PriceList`, `Product`, `TimeFrameGroup` etc.), el tipo de parámetro en la consulta SQL será siempre de tipo `nvarchar` y se corresponderá con los IDs de los valores seleccionados separados por comas.

Por ejemplo, en un parámetro de tipo `Family`, si sólo se ha seleccionado la familia con ID = 19, el parámetro recibido por la consulta SQL tendrá como valor '19' (tipo `nvarchar`), pero si se han seleccionado las familias con ID=19 e ID=42, el parámetro recibido por la consulta SQL tendrá como valor '19,42' (tipo `nvarchar`).

Para poder tratar con este tipo de valores, está definida una función en SQL llamada `stringToInts` que permite convertir una cadena de texto de IDs separados por comas en una tabla de IDs y escribir consultas de la forma:

```
select *
from Family f
where f.Id in (select * from stringToInts(@familyIds))
```

Es importante señalar que al utilizar tipos de datos complejos, Ágora mostrará al usuario que genera el informe sólo aquellos valores que están disponibles para sus grupos de TPVs. Eso permite limitar la información visible para cada usuario y evitar que, por ejemplo, un usuario pueda seleccionar un almacén que no esté asociado a sus grupos de TPVs. Esto es especialmente importante en instalaciones centralizadas donde suele ser necesario que cada usuario sólo pueda ver los datos de determinados locales.

Los valores personalizados se definen en el atributo type del elemento `<Param />` como una serie de valores separados por una barra vertical (|), como se muestra en el siguiente ejemplo, donde las opciones que vería el usuario serían *Hoy*, *Ayer* y *Este Mes*:

```
<Param name='period' label='Periodo:' type='values:Hoy|Ayer|Este Mes' />
```

La cadena de valores debe comenzar por `values:` para indicarle a Ágora que el parámetro está compuesto de valores personalizados. Al enviar el valor a la consulta SQL siempre será de tipo `nvarchar`.

Consulta SQL

Una vez establecidos los parámetros del informe es necesario definir la consulta SQL que se utilizará para obtener los resultados:

```
<?xml version="1.0" encoding="utf-8" ?>
<Query name='Productos del día' group='Catálogo'
       guid='{E71A1FD9-218D-4FD3-ADA1-898375B61C2A}'>
  <Params>
    <Param name='businessDay' label='Fecha:'
           type='date' required='true' />
    <Param name='posGroupIds' label='Grupo TPVs:'
           type='posGroup' required='true' multipleValue='true' />
  </Params>
  <Sql><![CDATA[
    select ps.ProductName Product, sum(ps.Quantity) Quantity
    from Productsales ps
    where ps.BusinessDay = @businessDay
      and ps.PosGroupId in (select * from stringToInts(@posGroupIds))
    group by ps.ProductName
    ]]>
  </Sql>
  <Columns>
    <!-- Columnas a mostrar -->
  </Columns>
  <RowStyles>
    <!-- Estilos de resaltado de filas -->
  </RowStyles>
</Query>
```

La consulta se indica dentro del elemento Sql y puede ser cualquier sentencia SQL válida. Es recomendable (aunque no obligatorio) introducirla dentro de un elemento `<![CDATA[ … ]]>` como se muestra en el ejemplo para no tener que codificar caracteres reservados de XML (como los símbolos < y >).

Los parámetros del informe estarán disponibles como parámetros de SQL tal y como se ha indicado en la sección anterior de este documento.

Visualización de Resultados

Para configurar la forma en que se muestran los resultados al usuario es necesario indicar las columnas que aparecerán en la tabla de resultados y su formato:

```
<?xml version="1.0" encoding="utf-8" ?>
<Query name='Productos del día' group='Catálogo'
       guid='{E71A1FD9-218D-4FD3-ADA1-898375B61C2A}'>
  <Params>
    <Param name='businessDay' label='Fecha:'
           type='date' required='true' />
    <Param name='posGroupIds' label='Grupo TPVs:'
           type='posGroup' required='true' multipleValue='true' />
  </Params>
  <Sql><![CDATA[
select ps.ProductName Product, sum(ps.Quantity) Quantity
from Productsales ps
where ps.BusinessDay = @businessDay
 and ps.PosGroupId in (select * from stringToInts(@posGroupIds))
group by ps.ProductName
]]>
  </Sql>
  <Columns>
    <Column name='Product' label='Producto'
            width='400' totalize='count'/>
    <Column name='Quantity' label='Cant. Total'
            width='120' format='asQuantity'
            align='right' totalize='sum'/>
  </Columns>
  <RowStyles>
    <!-- Estilos de resaltado de filas -->
  </RowStyles>
</Query>
```

Cada columna que se mostrará al usuario aparece definida en un elemento `Column` que debe incluir:

name
: Nombre de la columna en el resultado generado por la consulta SQL.

label
: Texto que se mostrará como cabecera de la columna en el interfaz de usuario.

fixed
: Indica si la columna debe permanecer fija (`true`) o no (`false`) al realizar scroll horizontal. Opcional, por defecto es `false`.

width
: Anchura (en píxeles de la columna). Opcional, por defecto es 100.

align
: Alineado de los valores de la columna. Opcional, por defecto es `left`. Los valores soportados son:

- **left**: izquierda.

- **center**: centrado.
- **right**: derecha.

format

Indica cómo se formatearán los valores de la columna. Opcional, por defecto no se aplica ningún formato. Los valores soportados son:

- **asInt**: entero (sin decimales). Ejemplo: 9 o 13.215.
- **asNumber**: número (con decimales). Ejemplo: 1.700,2 o 84,123.
- **asPrice**: precio (siempre dos decimales). Ejemplo: 9,00 o 1.010,12.
- **asQuantity**: cantidad (siempre tres decimales). Ejemplo: 19,000 o 8,001.
- **asMoney**: moneda (siempre dos decimales y símbolo). Ejemplo: 9,00 € o 1.010,12 €.
- **asDate**: fecha. Ejemplo: 29/12/2019.
- **asTime**: hora. Ejemplo: 23:41.
- **asDateAndTime**: fecha y hora. Ejemplo: 29/12/2019 23:41.

totalize

Función de agregación mostrada en el pie de la columna. Opcional, por defecto no es `none`. Los valores soportados son:

- **none**: no habrá ningún totalizador.
- **sum**: calcular la suma de los valores.
- **count**: indicará el número de filas del resultado.
- **avg**: calculará la media de los valores.

href

URL a la que enlazará la columna. Opcional, por defecto la columna no enlaza ninguna URL. Se pueden usar marcadores de posición para generar URLs dependientes del valor de la fila. Por ejemplo, `...href='#/admin/products/${ProductId}'...` generaría una URL en para navegar la ficha de producto cuyo Id se ha calculado en la columna `ProductId` de la consulta SQL.

Para consultas sencillas o consultas en las que el número de columnas no se puede conocer a priori (por ejemplo, un `PIVOT` dinámico), es posible utilizar un tipo de columna especial:

```
<Column name='*' width='400' totalize='count'/>
```

Esta columna será reemplazada en el conjunto de resultados por todas aquellas columnas que no se estén mostrando explícitamente, usando como cabecera de la columna el nombre de la columna en el conjunto de resultados de la consulta SQL.

Resaltado de Filas

Si necesita resaltar filas con un formato especial, por ejemplo para remarcar una fila que contiene totales, o una fila con un valor concreto, puede definir estilos de filas:

```
<?xml version="1.0" encoding="utf-8" ?>
<Query name='Productos del día' group='Catálogo'
       guid='{E71A1FD9-218D-4FD3-ADA1-898375B61C2A}'>
  <Params>
    <Param name='businessDay' label='Fecha:'
           type='date' required='true' />
    <Param name='posGroupIds' label='Grupo TPVs:'
           type='posGroup' required='true' multipleValue='true' />
  </Params>
  <Sql><![CDATA[
select ps.ProductName Product, sum(ps.Quantity) Quantity
from Productsales ps
where ps.BusinessDay = @businessDay
  and ps.PosGroupId in (select * from stringToInts(@posGroupIds))
group by ps.ProductName
]]>
  </Sql>
  <Columns>
    <Column name='Product' label='Producto'
            width='400' totalize='count'/>
    <Column name='Quantity' label='Cant. Total'
            width='120' format='asQuantity'
            align='right' totalize='sum'/>
        </Columns>
        <RowStyles>
      <When column='Product' equals='Fanta'
            applyCssClass='font-bold'/>
      <When column='Product' equals='Coca Cola'
            applyCssClass='font-extra-bold'/>
  </RowStyles>
</Query>
```

Cada regla de estilo de filas se define en un elemento When que debe incluir:

column
: Nombre de la columna en el resultado generado por la consulta SQL sobre la que se evaluará la condición.

equals
: Valor que deberá tener la columna indicada para que se aplique la regla.

applyCssClass

Clase css a aplicar a la fila. Puede introducir varias clases separadas por espacio. Entre las clases disponibles, se encuentran:

- `font-trans`, `font-light`, `font-bold`, `font-extra-bold`: distintos pesos de fuente.
- `text-primary`, `text-primary2`, `text-info`, `text-success`, `text-warning`, `text-warning2`, `text-danger`, `text-danger2`: colores para el texto incluidos en la paleta de colores de Ágora.
- `row-primary`, `row-primary2`, `row-info`, `row-success`, `row-warning`, `row-warning2`, `row-danger`, `row-danger2`, `row-white`, `row-gray-100`, `row-gray-200`, `row-gray-300`, `row-gray-400`, `row-gray-500`: colores de fondo incluidos en la paleta de colores de Ágora.
- `row-border-top-primary2`, `row-border-top-info`, `row-border-top-success`, `row-border-top-warning`, `row-border-top-warning2`, `row-border-top-danger`, `row-border-top-danger2`, `row-border-top-primary`, `row-border-top-gray-100`, `row-border-top-gray-200`, `row-border-top-gray-300`, `row-border-top-gray-400`, `row-border-top-gray-500`: borde superior con los colores incluidos en la paleta de colores de Ágora.
- `row-border-bottom-primary`, `row-border-bottom-primary2`, `row-border-bottom-info`, `row-border-bottom-success`, `row-border-bottom-warning`, `row-border-bottom-warning2`, `row-border-bottom-danger`, `row-border-bottom-danger2`, `row-border-bottom-primary`, `row-border-bottom-gray-100`, `row-border-bottom-gray-200`, `row-border-bottom-gray-300`, `row-border-bottom-gray-400`, `row-border-bottom-gray-500`: borde inferior con los colores incluidos en la paleta de colores de Ágora.

Las reglas se estilo se evalúan comparando en cada fila el valor de una columna concreta con un valor prefijado.

Por ejemplo, la siguiente regla se cumplirá en aquellos registros en los que la columna Product tenga el valor Fanta, y en ese caso pondrá el texto en negrita:

```
<When column='Product' equals='Fanta' applyCssClass='font-bold'/>
```

Puede incluir tantas reglas como desee, y en cada fila se aplicarán todas aquellas que cumplan la condición de que la columna tiene el valor indicado.

Si necesita utilizar reglas más complejas, por ejemplo formatear celdas cuando un valor sea negativo, puede calcular la condición en la consulta SQL y utilizar el resultado luego en la regla de estilo. Por ejemplo:

```
<Sql><![CDATA[
  select
  ...,
  case when Quantity < 0 then 1 else 0 end IsNegative
  ...
]]>
</Sql>
<RowStyles>
  <When column='IsNegative' equals='1' applyCssClass='text-danger'/>
</RowStyles>
```

Instalación

Para desplegar un informe personalizado en una instalación de Ágora basta con copiar el fichero que contiene el informe dentro de la carpeta:

```
[Carpeta de instalación de Ágora]\custom-queries
```

Ágora detectará automáticamente la aparición del nuevo fichero y procederá a la instalación del informe. Si en algún momento desea desinstalar el informe, deberá eliminar el fichero correspondiente de esa carpeta.

Al instalar el informe Ágora creará las entradas correspondientes dentro de la sección de Informes de la Administración y creará un nuevo permiso asociado a la visualización del informe llamado *Consulta: NOMBRE DEL INFORME*. El perfil de usuario *Administrador* tendrá asignado automáticamente ese permiso, pero deberá asignarlo manualmente al resto de perfiles que desee que puedan ejecutar la consulta.

Ejemplo Completo

Ágora utiliza el sistema de informes personalizadas para algunos informes incluidos con la versión estándar del producto. Puede encontrar los ficheros de definición en la misma carpeta dónde puede desplegar sus propios informes:

```
[Carpeta de instalación de Ágora]\custom-queries
```

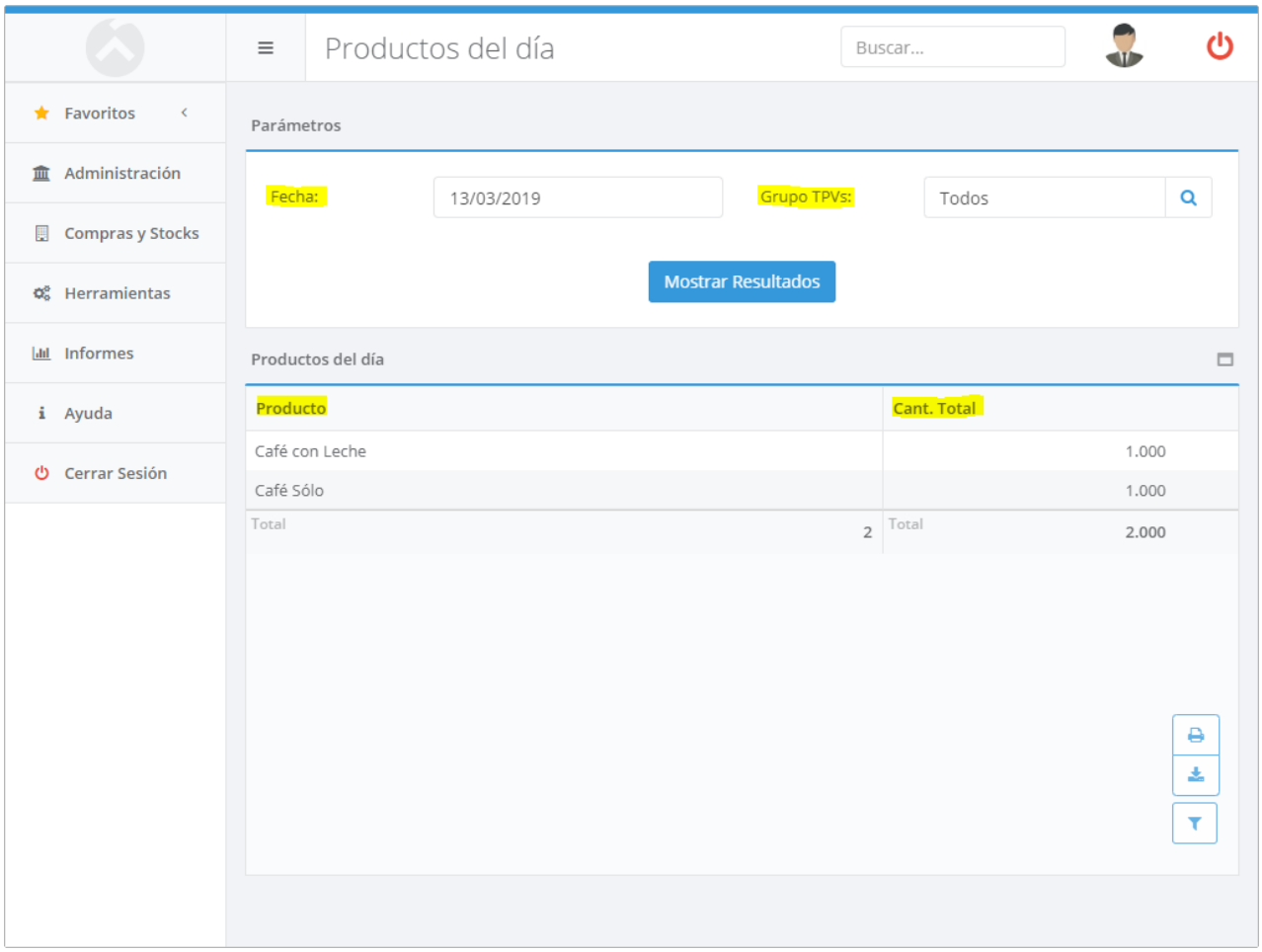
Estos ejemplos le ayudarán a comprender mejor cómo funciona el sistema.

No obstante, a continuación se muestra un ejemplo completo de informe (muy simple) sobre el que analizaremos los distintos componentes.

El objetivo de este informe será mostrar los productos vendidos en una fecha determinada en los grupos de TPVs seleccionados, totalizando la cantidad vendida de cada producto.

El informe se llamará *Productos por día* y estará disponible dentro del grupo de informes de *Catálogo* de la sección de Informes de la Administración:

Al acceder a él se mostrará el interfaz de usuario necesario para establecer los parámetros del informe, y al ejecutarlo se visualizarán los resultados en el siguiente formato:



El fichero completo de definición del informe es el siguiente:

```
<?xml version="1.0" encoding="utf-8" ?>
<Query name='Productos del día' group='Catálogo'
       guid='{E71A1FD9-218D-4FD3-ADA1-898375B61C2A}'>
  <Params>
    <Param name='businessDay' label='Fecha:'
           type='date' required='true' />
    <Param name='posGroupIds' label='Grupo TPVs:'
           type='posGroup' required='true' multipleValue='true' />
  </Params>
  <Sql><![CDATA[
  select ps.ProductName Product, sum(ps.Quantity) Quantity
  from Productsales ps
  where ps.BusinessDay = @businessDay
    and ps.PosGroupId in (select * from stringToInts(@posGroupIds))
  group by ps.ProductName
  ]]>
  </Sql>
  <Columns>
    <Column name='Product' label='Producto'
            width='400' totalize='count'/>
    <Column name='Quantity' label='Cant. Total'
            width='120' format='asQuantity'
            align='right' totalize='sum'/>
    </Columns>
</Query>
```

Lo primero que encontramos es la definición de la consulta indicando su nombre y posición dentro de la sección de Informes de Ágora. En este caso, al grupo de informes de *Catálogo* se le añadirá una nueva entrada, *Productos del día*:

```
<Query name='Productos del día' group='Catálogo' ...
```

A continuación se definen los parámetros de la consulta. En este caso será la fecha de negocio para la cual queremos obtener el informe y los grupos de TPVs (uno o más) sobre los que queremos obtenerlos:

```
<Params>
  <Param name='businessDay' label='Fecha:'
         type='date' required='true' />
  <Param name='posGroupIds' label='Grupo TPVs:'
         type='posGroup' required='true' multipleValue='true' />
</Params>
```

Con ello, Ágora generará el interfaz de usuario de parametrización que aparece en la captura anterior.

Además, se asegurará de que las opciones disponibles son las permitidas para el usuario actual, evitando que pueda acceder a Grupos de TPVs a los que no está asociado.

Cuando el usuario pulse Mostrar Resultados, se ejecutará la consulta SQL que hemos definido:

```
<Sql><![CDATA[
select ps.ProductName Product, sum(ps.Quantity) Quantity
from Productsales ps
where ps.BusinessDay = @businessDay
  and ps.PosGroupId in (select * from stringToInts(@posGroupIds))
group by ps.ProductName
]]>
</Sql>
```

Esta consulta hace una simple agregación sobre la vista `ProductSales` filtrando por fecha de negocio y grupos de TPVs. En ella se puede apreciar cómo son accesibles los parámetros `@businessDay` y `@posGroupIds` que hemos definido anteriormente y cómo se hace uso de la función `stringToInts` para extraer los IDs de los grupos de TPVs seleccionados.

Por último, se definen las columnas que se mostrarán al usuario, en este caso `Producto` y `Cant. Total`. En ellas se referencian las columnas obtenidas de por la consulta de SQL, `Product` y `Quantity`, y se establecen las opciones de formato necesarias:

```
<Columns>
  <Column name='Product' label='Producto'
          width='400' totalize='count'/>
  <Column name='Quantity' label='Cant. Total'
          width='120' format='asQuantity'
          align='right' totalize='sum'/>
</Columns>
```