Parámetros de consulta global

La mayoría de las operaciones de punto de conexión de la API de Catalogo se pueden manipular con los siguientes parámetros. Es importante entenderlos para sacar el máximo partido a la plataforma.

Campos

Elija los campos que se devuelven en el conjunto de datos actual. Este parámetro admite la notación de puntos para solicitar campos relacionales anidados. También puede utilizar un comodín (*) para incluir todos los campos a una profundidad específica.

```
GET /items/services
  ?fields=name,description,status_id.name
```

Ejemplos

Campo	Descripción
`name,description`	Devuelve solo los campos `name` y `description`.
`name,status_id.name`	Devuelve `name` y el elemento `name` relacionado con `status_id`
`*`	Devuelve todos los campos.
`*.*`	Devuelve todos los campos de primer y segundo nivel relacionado.
`*,institution_id.*`	Devuelve todos los campos y todos los campos de `institution_id`.

> **Rendimiento y tamaño**
>
> Aunque el comodín `fields` es muy útil para fines de depuración, se recomienda solicitar solo campos específicos para su uso en producción. Al solicitar solo los campos que realmente necesita, puede acelerar la solicitud y reducir el tamaño total de la salida.

Relaciones

En el **Catálogo de Servicios** algunos campos de la colección `services` almacenan datos relacionados con múltiples colecciones mediante campos relacionales o tablas intermedias.

Para solicitar **campos específicos** de cada relación, utiliza la sintaxis:

```
?fields=<relacion>.<subrelacion>.<campo>
```

Esto permite que, al consultar servicios, podamos incluir campos de colecciones relacionadas sin traer datos innecesarios.

Escenario

Sabiendo que tenemos la colección `services`, que tiene relaciones con otras colecciones:

- **status_id** → colección `status`
- **service_type_id** → colección `service_type`
- **addressed_to_id** → colección `addressed_to`

Queremos traer:

- Campos básicos del servicio
- Estado y tipo
- Público objetivo

Quedaría de esta manera:

```
GET /items/services
?fields[]=name
  &fields[]=status_id.name
  &fields[]=service_type_id.type
  &fields[]=addressed_to_id.other_entity
```

Filtro

Se utiliza para buscar elementos en una colección que coincida con las condiciones del filtro. El parámetro filter sigue la especificación Reglas de filtro, que incluye información adicional sobre operadores lógicos (AND/OR), filtrado relacional anidado y variables dinámicas.

Hay dos sintaxis disponibles:

```
GET /items/services
    ?filter[name][_eq]=Renovación pasaporte

// or

GET /items/services
    ?filter={ "name": { "_eq": "Renovación pasaporte" }}
```

Ejemplos

Recupera todos los elementos donde es igual a "Renovación pasaporte" **name**

```
{
  "name": {
    "_eq": "Renovación pasaporte"
  }
}
```

Recupere todos los elementos de uno de los siguientes estados: "Borrador", "Publicado"

```
{
  "status_id": {
    "_in": [1, 2]
  }
}
```

Recuperar todos los elementos que se publican entre dos fechas

```
{
  "date_created": {
    "_between": ["2021-01-24", "2021-02-23"]
  }
}
```

Recupere todos los elementos en los que la marca "**default_variation**" de una variación es verdadera

```
{
  "variation_ids": {
```

```
  "default_variation": {
    "_eq": true
  }
}
}
```

> **Filtros anidados**
>
> En el ejemplo anterior se filtrarán los elementos de nivel superior en función de una condición del elemento relacionado. Si quieres filtrar los elementos relacionados, ¡echa un vistazo al parámetro deep!

Buscar

El parámetro de búsqueda permite realizar una búsqueda en todos los campos de tipo cadena y texto de una colección. Es una manera fácil de buscar un artículo sin crear filtros de campo complejos, aunque está mucho menos optimizado. Solo busca en los campos del elemento raíz, no se incluyen los campos de elementos relacionados.

Ejemplo

Buscar todos los servicios que mencionan `Pasaporte`

```
GET /items/services
  ?search=Pasaporte
```

Ordenar

La ordenación predeterminada es ascendente, pero se puede usar un signo menos `-` para revertir esto a un orden descendente. Los campos se priorizan según el orden del parámetro. La notación de puntos debe usarse cuando se ordena con valores de campos anidados.

Ejemplos

Campo	Descripción
`date_created`	Ordenar por fecha de creación descendente.

sort,-date_updated	Ordene por un campo de "ordenación", seguido de la fecha de actualización descendente.
sort,-variation_ids.name	Ordenar por un campo de "ordenación", seguido del nombre de una variación anidada.

```
GET /items/services
    ?sort=sort,-date_updated,author.name

// or

GET /items/services
    ?sort[]=sort
    &sort[]=-date_updated
    &sort[]=-variation_ids.name
```

Límite

Establezca el número máximo de artículos que se devolverán. El límite predeterminado se establece en `100`.

Ejemplos

- Consigue los primeros 200 artículos: `200`
- Obtener el número máximo permitido de artículos: `-1`

```
GET /items/services
    ?limit=200
```

> **Límites elevados y rendimiento**
>
> Dependiendo del tamaño de la colección, obtener la cantidad máxima de elementos puede provocar una disminución del rendimiento o tiempos de espera más largos. Úselo con precaución.

Omitir

Omita los primeros `n` elementos de la respuesta. Este parámetro se puede utilizar para la paginación.

Ejemplos

- Consigue los objetos 101—200

```
GET /items/services
  ?offset=100
```

Página

Una alternativa a `offset`. La página es una forma de establecer `offset` bajo el capó mediante el cálculo de **`limit * page`**. La primera página que está indexada es `1`.

Ejemplos

- Obtener artículos 101-200

```
GET /items/services
  ?page=2
```

Agregación y agrupación

Agregación

Las funciones de agregado o `aggregate` le permiten realizar cálculos en un conjunto de valores, devolviendo un único resultado.

Las siguientes funciones de agregación están disponibles en Catalogo:

Nombre	Descripción
`count`	Cuenta cuántos elementos hay
`countDistinct`	Cuenta cuántos objetos únicos hay
`sum`	Suma los valores del campo dado
`sumDistinct`	Suma los valores únicos en el campo dado
`avg`	Obtener el valor promedio del campo dado
`avgDistinct`	Obtener el valor promedio de los valores únicos en el campo dado

Nombre	Descripción
`min`	Devuelve el valor más bajo del campo
`max`	Devuelve el valor más alto del campo

```
GET /items/services
    ?aggregate[count]=*
```

Agrupación

De forma predeterminada, las funciones de agregación anteriores se ejecutan en todo el conjunto de datos. Para permitir informes más flexibles, puede combinar la agregación anterior con la agrupación. La agrupación permite ejecutar las funciones de agregación en función de un valor compartido. Esto permite cosas como "Calificación promedio por mes" o "Ventas totales de artículos en la categoría de jeans".

La consulta permite agrupar varios campos simultáneamente. En combinación con las funciones, esto permite la generación de informes agregados por año-mes-fecha.groupBy

```
GET /items/services
    ?aggregate[count]=name,description
    &groupBy[]=institution_id
    &groupBy[]=date_updated
```

Profundo

`deep` le permite establecer cualquiera de los otros parámetros de consulta (excepto `fields` y `deep` en sí) en un conjunto de datos relacional anidado.

Ejemplos

- Limite las variaciones relacionadas anidados a 3

```
{
  "variation_ids": {
    "_limit": 3
  }
}
```

- Solo obtenga 3 artículos relacionados, con solo el comentario mejor calificado anidado

```
{
  "variation_ids": {
    "_limit": 3,
    "result_ids": {
      "_sort": "name",
      "_limit": 1
    }
  }
}
```

Hay dos sintaxis disponibles:

```
GET /items/services
  ?deep[institution_id][_filter][acronym][_eq]=OGTIC

// or

GET /items/services
  ?deep={ "institution_id": { "_filter": { "acronym": { "_eq": "OGTIC" }}}}
```

Alias

Los `alias` le permiten cambiar el nombre de los campos sobre la marcha y solicitar el mismo conjunto de datos anidados varias veces utilizando diferentes filtros.

> **Campos anidados**
>
> Solo es posible asignar alias a los campos del mismo nivel. El alias de los campos anidados, por ejemplo, `field.nested`, no funcionará.

```
GET /items/services
  ?alias[name]=nombre
  &alias[description]=descripcion
  &deep[institution_id][_filter][acronym][_eq]=OGTIC
```

Exportar

Guarde la respuesta del API actual en un archivo. Acepta un tipo de los siguientes: `csv`, `json`, `xml`, `yaml`.

```
GET /items/services
  ?export=type
```

Ejemplos:

```
?export=csv
?export=json
?export=xml
?export=yaml
```

Funciones

Las funciones permiten la modificación "en vivo" de los valores almacenados en un campo. Las funciones se pueden usar en cualquier parámetro de consulta que normalmente proporcionaría una clave de campo, incluidos los campos, la agregación y el filtro.

Las funciones se pueden usar envolviendo la clave de campo en una sintaxis similar a JavaScript, por ejemplo:

- `timestamp` -> `year(timestamp)`

Funciones de fecha y hora

Filtro	Descripción
`year`	Extraer el año de un campo de fecha y hora/fecha/marca de tiempo
`month`	Extraer el mes de un campo datetime/date/timestamp
`week`	Extraer la semana de un campo datetime/date/timestamp
`day`	Extraer el día de un campo de fecha y hora/fecha/marca de tiempo
`weekday`	Extraer el día de la semana de un campo datetime/date/timestamp

Filtro	Descripción
hour	Extraer la hora de un campo datetime/date/timestamp
minute	Extraer el minuto de un campo de fecha y hora/fecha/marca de tiempo
second	Extraiga el segundo de un campo de fecha y hora/fecha/marca de tiempo

Funciones de matriz

Filtro	Descripción
count	Extraer el número de elementos de una matriz JSON o un campo relacional

```
GET /items/services
  ?fields=id,name,weekday(date_created)
  &filter[year(date_created)][_eq]=2024
```

Revisión #10
Creado 23 julio 2025 02:09:20 por Luis Matos
Actualizado 28 agosto 2025 14:20:25 por Luis Matos