Aspire External API

Contents

Overview

The Aspire External API uses Open Data Protocol (OData), or a certain numberof query options from the protocol to allow for easy data access via URL, similarto SQL. The following section will provide all the possible query options that were implemented and the overall syntax while using these query options, accompanied by subsections which present examples of how specific query options can be used to construct clauses.

Query Options

There are five possible query options which users of the Aspire External API can utilize; they are listed below:

1. **filter**, SQL equivalent is WHERE
2. **select**, SQL equivalent is SELECT
3. **orderby**, SQL equivalent is ORDER BY
4. **top**, SQL equivalent is TOP
5. **skip**, SQL equivalent is SKIP

The first three query options work directly with and only with fields/properties, so in order to filter out, select or order by, the user will have to specify the fields the clause/s should be utilized on. The remaining two query options work with integers only. All the query options/clauses can be used separately or in a combination. It is up to the user what data should be used and how it should be presented. The order of the query options is not strict; however, it is worth noting that the same query option cannot be repeated multiple times.

The list of key query symbols is below:

- **?** - marks the beginning of a query; written only once after the endpointname
- **$** - marks the beginning of a query option/clause; written as many times as there are query options in a request; preceded by **?** (first query option) or **&** (the remaining query options in a request)
- **& -** marks each subsequent query option after the first; written everytime a new query option/clause is added; succeeded by **$**

Refer to the following subsections for examples that are specific to each queryoption. The last subsection presents examples of combinations of all query options.

Filter

The filter query option can be used on all fields for a given endpoint (without the nested properties and their parent). That can be done by means of logical and comparison operators and they are as follows:

- Logical operators: **and** (expression on both sides is true), **or** (expressionon either side is true), **contains** (query any match in the value), **startswith** (query value from left to right), **endswith** (query value fromright to left)
- Comparison operators: **eq** (equals), **ne** (does not equal), **gt** (greater than), **lt** (less than), **ge** (greater than or equals), **le** (less than or equals)

Before we get to a couple of examples, through which this will be demonstrated, it is important to say that the filter query option can be utilized so that the user creates as many combinations as possible of the operators as desired, as long as the setting makes sense, and the syntax is respected.

EXAMPLES:

- {API_URL}/Contacts**?$filter=Active eq true** - This should return only theresults which contain active contacts
- {API_URL}/Contacts**?$filter=Email ne 'email@email.com'** - This should return only the results which do not contain the filtered email
- {API_URL}/Opportunities**?$filter=AnticipatedCloseDate gt 2020-11-20** - This should return only the results which contain anticipated close datethat is greater than the filtered one
- {API_URL}/Opportunities**?$filter=EstimatedDollars lt 2000** - This should return only the results which contain estimated dollars valuethat is less than the filtered one
- {API_URL}/ClockTimes**?$filter=ClockStart ge 08:00:00** - This shouldreturn only the results which contain clock start time value that is greater than or equal to the filtered one
- {API_URL}/ClockTimes**?$filter=BreakTime le 1** - This should return only the results which contain break time value that is less than or equal to the filtered one
- {API_URL}/Companies**?$filter=startswith(CompanyName, 'Company')** - This should return companies whose CompanyName starts with 'Company'
- {API_URL}/Companies**?$filter=endswith(CompanyName, 'Name')** - This should return companies whose CompanyName ends with 'Name'
- {API_URL}/Companies**?$filter=contains(CompanyName, 'part of name')** - This should return companies whose CompanyName contains'part of name'
- {API_URL}/Contacts**?$filter=Active eq true and CompanyID eq 1111** -This should return only the results which contain active contacts andthat belong to the specified company
- {API_URL}/Contacts**?$filter=(Active eq true and CompanyID ne 1111) or Email eq 'email@email.com'** - This should return only the results which contain active contacts that do not belong to the specified company, orresults that contain the specific email value

These are some of the ways filtering can be used, it is up to the user to determine how to filter out the data. The value used for filtering is either closed under ' ' (single quotes) or not. The single quotes are used when filtering properties of type string, other types (bool, int, double/decimal, DateTime) do not require single quotes when filtering.

NOTE: If you are using **Swagger UI** you will omit the query symbol/s and queryoption name because it is already predefined within the **$filter** field, you wouldonly enter the filter value e.g., **AnticipatedCloseDate gt 2020-11-20**.

Select

The select query option can be used on all fields for a given endpoint (without the nested properties and their parent). Selecting specific field/s should return only the selected field/s in the response. The number of selected fields/properties is not limited.

EXAMPLES:

- {API_URL}/Contacts**?$select=FirstName -** This should return only the contacts' FirstName for as many results that exist
- {API_URL}/Contacts?**$select=FirstName,LastName,CompanyName,ContactTypeID** - This should return only the contacts' FirstName, LastName, CompanyName and ContactTypeID for as many results thatexist

NOTE: If you are using **Swagger UI** you will omit the query symbol/s and query option name because it is already predefined within the **$select** field, you would only enter the fields you wish to be selected out, e.g., **FirstName,LastName,CompanyName,ContactTypeID**.

Order By

The orderby query option can be used on all fields for a given endpoint (without the nested properties and their parent). This query option sorts the results by the given field either alphabetically, numerically or by date, depending on the field's type. Sort can be either in ascending or descending order.

EXAMPLES:

- {API_URL}/Contacts**?$orderby=ContactID** - This should sort the resultsby ContactID in ascending order (default order since none is specified)
- {API_URL}/Divisions**?$orderby=AccountNumber asc** - This should sortthe resulting Divisions by AccountNumber in ascending order
- {API_URL}/Companies**?$orderby= CreatedDateTime desc** - This shouldsort the resulting Companies by CreatedDateTime in descending order

NOTE: If you are using **Swagger UI** you will omit the query symbol/s and queryoption name because it is already predefined within the **$orderby** field, you would only enter the property by which you wish to order and the sort order, e.g., **AccountNumber asc**.

Top

The top query option allows the user to select the number of records they wish to see. Since the page size is limited to 1000 records, so is the top query option.

EXAMPLES:

- {API_URL}/Contacts**?$top=10** - This should return only the first 10contacts in the result list
- {API_URL}/Contacts**?$top=50** - This should return only the first 50contacts in the result list
- {API_URL}/Contacts**?$top=200** - This should return only the first 200contacts in the result list

NOTE: If you are using **Swagger UI** you will omit the query symbol/s and queryoption name because it is already predefined within the **$top** field, you wouldonly enter the number of records you wish to be returned, e.g., **100**.

Skip

The skip query option allows the user to skip a specified number of records, it acts as a form of pagination.

EXAMPLES:

- {API_URL}/Contacts**?$skip=10 -** This should skip the first 10 contacts inthe result list
- {API_URL}/Contacts**?$skip=50** - This should skip first 50 contacts in theresult list
- {API_URL}/Contacts**?$skip=200 -** This should skip the first 200 contactsin the result list

NOTE: If you are using **Swagger UI** you will omit the query symbol/s and queryoption name because it is already predefined within the **$skip** field, you wouldonly enter the number of records you wish to be skipped, e.g., **500**.

Multiple Query Options

The previous subsections defined the construct of individual clauses and, as previously stated, users can utilize multiple query options in a single request to get a more representable data overview. The following examples cover some of the possible combinations of query options.

EXAMPLES:

- {API_URL}/Properties**?$filter=BranchName ne 'Some branch name' and Active eq true&$select=PropertyID,PropertyName,CountyID** - This should return only PropertyID, PropertyName and CountyID fieldswhere BranchName is not equal to 'Some branch name' and status is active (Active = true)
- {API_URL}/Properties**?$top=50&$skip=100 -** This should skip the first100 results and return the succeeding 50 properties
- {API_URL}/ItemAllocations**?$filter=LastModifiedByUserName eq 'Some username'&$top=100&$orderby=LastModifiedByUserName asc** - This should return the first 100 ItemAllocation records, sorted in ascending order, where LastModifiedByUserName is 'Some username'
- {API_URL}/EmployeeIncidents**?$filter=ContactName eq 'Some contact name'&$select=EmployeeIncidentID&$top=100&$skip=100&$orderby =EmployeeIncidentID asc** - This is a combination of all query optionsand should return only the EmployeeIncidentID for the second 100 results, where ContactName is 'Some contact name', sorted in ascending order by EmployeeIncidentID

NOTE: If you are using **Swagger UI**, it also allows for combination of all the query options possible, simply follow through the notes and syntax in each query option subsection.