Querying events

There are two different endpoints that work about the same but will return EPCIS events in a different format: JSON or XML. Please note that only EPCIS events of the last 18 months can be retrieved trough the v3 endpoints. Older events will be made available through a separate endpoint with limited query functionality. Until that time, use the v2 endpoint to retrieve events older than 18 months.

JSON endpoint

The `/epcis/v3/query` endpoint is an endpoint that returns EPCIS events in JSON format. Our advise is to use this endpoint when user defined extensions are not needed as it is easier to use and has more features.

XML endpoint

The `/epcis/v3/soap` endpoint is an XML SOAP endpoint that implements (parts of) the EPCIS Query Control Interface that is defined by the EPCIS specification 1.2 (section 8.2.5).

The `poll` operation is supported, and the `SimpleEventQuery` is the only (predefined) query that is partially supported. The query parameters that are supported for the JSON endpoint at `/epcis/v3/query` are also supported, see the list of supported parameters below. The SOAP interface doesn't support cursors though. Also, this endpoint follows the naming of parameters as defined by the EPCIS standard.

The SOAP endpoint also doesn't support subscribing to EPCIS queries.

Query

A query is expressed as a list of zero or more parameters. A query without any parameters will match all events. You can add parameters to further restrict the set of events that match. An event must match all provided parameters, so if there is more than one parameter, these will be combined with AND logic. Some parameters allow to provide a list of values (e.g. when using `EQ_bizLocation`). These are combined with OR, meaning that events will match if at least one of the values provided matches.

Supported parameters

This query endpoint implements `SimpleEventQuery` as defined by section 8.2.7.1 of the EPCIS standard, version 1.2. Such a query consists of a number or query parameters and one or more query values. All parameters are optional: an empty query simply matches all events.
Below is a list of query parameters that are supported. Many of the query parameters are prefixed with `EQ_`, `GE_` or `LT_`. `EQ_` means that the field should be equal to the provided value. `GE_` means that the value should be greater than or equal to the provided value. `LT_` means that the field should be less than (and thus not equal to) the provided value. Parameters prefixed with `MATCH_` deal with matching EPC URIs. More details about this are described below.

Name	Value type	Description
`eventType`	One or more strings	If specified, the result will only include events whose type matches one of the types specified in the parameter value. Each element of the parameter value may be one of the following strings: `ObjectEvent`, `AggregationEvent`, `QuantityEvent`, `TransactionEvent`, or `TransformationEvent`.
`GE_eventTime`	ISO 8601 date/time	If specified, **only** events with `eventTime` greater than or equal to the specified value will be included in the result.

Location identifiers explained

Time zones explained

Name	Value type	Description
`LT_eventTime`	ISO 8601 date/time	If specified, only events with `eventTime` less than the specified value will be included in the result.
`GE_recordTime`	ISO 8601 date/time	If provided, only events with `recordTime` greater than or equal to the specified value will be returned.
`LT_recordTime`	ISO 8601 date/time	If provided, only events with `recordTime` less than the specified value will be returned.
`EQ_action`	One or more strings	If specified, the result will only include events that (a) have an action field; and where (b) the value of the action field matches one of the specified values. The elements of the value of this parameter each must be one of the strings `ADD`, `OBSERVE`, or `DELETE`.
`EQ_bizStep`	One or more strings	If specified, the result will only include events that (a) have a non-null `bizStep` field; and where (b) the value of the `bizStep` field matches one of the specified values.
`EQ_disposition`	One or more strings	Like the `EQ_bizStep` parameter, but for the `disposition` field.
`EQ_readPoint`	One or more strings	If specified, the result will only include events that (a) have a non-null `readPoint` field; and where (b) the value of the `readPoint` field matches one of the specified values.
`EQ_bizLocation`	One or more strings	Like the `EQ_readPoint` parameter, but for the `bizLocation` field.
`EQ_transformationID`	One or more strings	If this parameter is specified, the result will only include events that (a) have a `transformationID` field (that is, `TransformationEvents`); and where (b) the `transformationID` field is equal to one of the values specified in this parameter.
`EQ_eventID`	One or more strings	Like the `EQ_readPoint` parameter, but for the `eventId` field.
`GE_errorDeclarationTime`	ISO 8601 date/time	If this parameter is specified, the result will only include events that (a) contain an `ErrorDeclaration`; and where (b) the value of the `errorDeclarationTime` field is greater than or equal to the specified value.
`LT_errorDeclarationTime`	ISO 8601 date/time	If this parameter is specified, the result will only include events that (a) contain an `ErrorDeclaration`; and where (b) the value of the `errorDeclarationTime` field is less than to the specified value.
`EQ_errorReason`	One or more strings	If this parameter is specified, the result will only include events that (a) contain an `ErrorDeclaration`; and where (b) the error declaration contains a non-null reason field; and where (c) the reason field is equal to one of the values specified in this parameter.
`EQ_bizTransaction_[type]`	One or more strings	This is not a single parameter, but a family of parameters. If a parameter of this form is specified, the result

Name	Value type	Description
		will only include events that (a) include a `bizTransactionList`; (b) where the business transaction list includes an entry whose `type` subfield is equal to `[type]` extracted from the name of this parameter; and (c) where the `bizTransaction` subfield of that entry is equal to one of the values specified in this parameter.
`EQ_source_[type]`	One or more strings	This is not a single parameter, but a family of parameters. If a parameter of this form is specified, the result will only include events that (a) include a `sourceList`; (b) where the source list includes an entry whose `type` subfield is equal to `[type]` extracted from the name of this parameter; and (c) where the `source` subfield of that entry is equal to one of the values specified in this parameter.
`EQ_destination_[type]`	One or more strings	This is not a single parameter, but a family of parameters. If a parameter of this form is specified, the result will only include events that (a) include a `destinationList`; (b) where the destination list includes an entry whose `type` subfield is equal to `[type]` extracted from the name of this parameter; and (c) where the `destination` subfield of that entry is equal to one of the values specified in this parameter.
`EQ_correctiveEventID`	One or more strings	If this parameter is specified, the result will only include events that (a) contain an `ErrorDeclaration`; and where (b) one of the elements of the `correctiveEventIDs` list is equal to one of the values specified in this parameter.
`EXISTS_errorDeclaration`	Void	If this parameter is specified, the result will only include events that contain an `ErrorDeclaration`.
`MATCH_epc`	One or more strings	If this parameter is specified, the result will only include events that (a) have an `epcList` or a `childEPCs` field (that is, `ObjectEvent`, `AggregationEvent`, `TransactionEvent`); and where (b) one of the EPCs listed in the `epcList` or `childEPCs` field (depending on event type) matches one of the EPC patterns or URIs specified in this parameter.
`MATCH_parentID`	One or more strings	Like `MATCH_epc`, but matches the `parentID` field of `AggregationEvent` or the `parentID` field of `TransactionEvent`.
`MATCH_inputEPC`	One or more strings	If this parameter is specified, the result will only include events that (a) have an `inputEPCList` (that is, `TransformationEvent`); and where (b) one of the EPCs listed in the `inputEPCList` field matches one of the EPC patterns or URIs specified in this parameter.
`MATCH_outputEPC`	One or more strings	If this parameter is specified, the result will only include events that (a) have an `outputEPCList` (that is, `TransformationEvent`); and where (b) one of the EPCs listed in the `outputEPCList` field matches one of the EPC patterns or URIs specified in this parameter.

Name	Value type	Description
`MATCH_anyEPC`	One or more strings	If this parameter is specified, the result will only include events that (a) have an `epcList` field, a `childEPCs` field, a `parentID` field, an `inputEPCList` field, or an `outputEPCList` field (that is, `ObjectEvent`, `AggregationEvent`, `TransactionEvent` or `TransformationEvent`); and where (b) the `parentID` field or one of the EPCs listed in the `epcList`, `childEPCs`, `inputEPCList`, or `outputEPCList` field (depending on event type) matches one of the EPC patterns or URIs specified in this parameter
`MATCH_epcClass`	One or more strings	If this parameter is specified, the result will only include events that (a) have a `quantityList` or a `childQuantityList` field (that is, `ObjectEvent`, `AggregationEvent` or `TransactionEvent`); and where (b) one of the EPC classes listed in the `quantityList` or `childQuantityList` field (depending on event type) matches one of the EPC patterns or URIs specified in this parameter. The result will also include `QuantityEvents` whose `epcClass` field matches one of the EPC patterns or URIs specified in this parameter.
`MATCH_inputEPCClass`	One or more strings	If this parameter is specified, the result will only include events that (a) have an `inputQuantityList` field (that is, `TransformationEvent`); and where (b) one of the EPC classes listed in the `inputQuantityList` field (depending on event type) matches one of the EPC patterns or URIs specified in this parameter.
`MATCH_outputEPCClass`	One or more strings	If this parameter is specified, the result will only include events that (a) have an `outputQuantityList` field (that is, `TransformationEvent`); and where (b) one of the EPC classes listed in the `outputQuantityList` field (depending on event type) matches one of the EPC patterns or URIs specified in this parameter.
`MATCH_anyEPCClass`	One or more strings	If this parameter is specified, the result will only include events that (a) have a `quantityList`, `childQuantityList`, `inputQuantityList`, or `outputQuantityList` field (that is, `ObjectEvent`, `AggregationEvent`, `TransactionEvent` or `TransformationEvent`); and where (b) one of the EPC classes listed in any of those fields matches one of the EPC patterns or URIs specified in this parameter. The result will also include `QuantityEvents` whose `epcClass` field matches one of the EPC patterns or URIs specified in this parameter.

Besides criteria to match events, we support the following parameters:

- `orderBy`: specifies the ordering of the results. Can be either `eventTime` or `recordTime`.
- `orderDirection`: the direction of ordering. Either `asc` for ascending order, or `desc` for descending order.

- `eventCountLimit` : an integer value, that specifies how many events should be returned. If more events match, then only the first matching events are returned.
- `maxEventCount` : like `eventCountLimit` , this specifies the number of events to return. However, if more events match, this will return an error.

Limiting query results

By default, there is a limit on the number of events that is returned, and the total size of the events. Events can vary in size (e.g. a count can include thousands of EPCs, while a commissioning event only has one EPC), and to ensure that queries are executed predictably fast and are not using excessive resources on the backend, we limit the response size (currently about 4MB). If the total size of events is larger than 4MB, even if the number of events is less than 1000, the EPCIS repository will still return an error.

There is also a limit on the number of events that can be matched, but this limit can be modified by query parameters. The default is 1000. If a query matches more than 1000 events, it will return an error. Using the parameter `maxEventCount` will increase this limit, up to a maximum of 5000 events. Like `maxEventCount` , you can use `eventCountLimit` to increase (or decrease) the limit, with the difference that eventCountLimit will not return an error. Instead, it will return the first matching events. The `has_more` field will then specify if there are more events that match (so you know that the response is not the complete set of events that matches the criteria).

Using `eventCountLimit` , in combination with `orderBy` can be used to find the first n or last n events that match criteria, e.g. find the last 10 counting events for a particular store. For example, to find the 10 most recent EPCIS events related to RFID counts for an organization (example for JSON query endpoint):

Example EPCIS query

```
{
        "parameters": [
                {
                        "name": "EQ_bizStep",
                        "value": "urn:epcglobal:cbv:bizstep:cycle_counting"
                },
                {
                        "name": "eventCountLimit",
                        "value": 10
                },
                {
                        "name": "orderBy",
                        "value": "eventTime"
                },
                {
                        "name": "orderDirection",
                        "value": "desc"
                }
        ]
}
```

When a query matches too many events, the EPCIS repository will return an error response (this behaviour is defined by the EPCIS specification). To retrieving a large number of events easier, the EPCIS repository supports cursors.

Matching EPCs

The criteria to match EPCs (e.g. `MATCH_epc`) supports matching on pure identity URIs, and pattern URIs. The following are valid examples:

- `urn:epc:id:sgtin:0193153.078298.202632890553` (a pure identity EPC URI)
- `urn:epc:idpat:sgtin:0193153.078298.*` (matches all URIs that have the same GTIN)
- `urn:epc:idpat:sgtin:0193153.*.*` (matches all URIs that have the same company prefix)
- `urn:epc:idpat:sgtin:*.*.*` (matches all URIs that are of the sgtin scheme)

Note that patterns start with `urn:epc:idpat:` , where pure identity URIs start with `urn:epc:id:` .

Most of the URI schemes defined by the TDS standard are supported (`urn:epc:id:sgtin` , `urn:epc:id:sscc` , `urn:epc:id:sgln` , `urn:epc:id:grai` , `urn:epc:id:giai` , `urn:epc:id:gsrn` , `urn:epc:id:gsrnp` , `urn:epc:id:gdti` , `urn:epc:id:cpi` , `urn:epc:id:sgcn` , `urn:epc:id:gid`).

Use the parameter `MATCH_epc` to find EPCIS events with serialized information (in the `epcList` or `childEPCs` field), or use parameter `MATCH_epcClass` to find EPCIS events with non-serialized information (in the `quantityList` or `childQuantityList` field). It is not possible to query for both in one query.

Response codes

The query endpoint can return the responses listed below. If the response is other than HTTP 200 OK, the HTTP response body contains reason field with a detailed message what went wrong.

Status code	Reason	Description
200 OK		The query was executed successfully. The response payload contains the results of the query.
400 Bad Request	Cannot parse JSON	The JSON is not syntactically well formed, or does not match the schema.
400 Bad Request	Invalid cursor value:	The provided value for a cursor cannot be parsed (if you provide the cursor value that was returned in the previous response, this should not happen).
400 Bad Request	The specified limit of x exceeds the maximum of y	When specifying a custom limit, it cannot exceed a given maximum value of 1000. Use cursors to iterate over larger result sets.
400 Bad Request	The query result is x bytes, which exceeds the maximum of y bytes	The total size of all EPCIS events that match the query is larger than the allowed maximum (about 6MB). Use cursors to iterate over larger result sets.
400 Bad Request	Your query matches too many events (it matched more than the allowed limit of y)	If the query matches more events than the given limit. The default limit is 100, and can be increased up to 1000. Use cursors to iterate over larger result sets.
400 Bad Request	Parameter in list value should be a (list of) string, but at least one value is not: x	The value provided should be a list of strings, but at least one element in the list is not a string.
400 Bad Request	Parameter value should be a (list of) string, but is x	The value provided should be a list of strings, but it is of a different value.
400 Bad Request	Parameter value should be time, but it is x	The value provided should be a time, but it is of another type.
400 Bad Request	Unsupported event type: x	The provided value is not a valid event type. Supported event types are `ObjectEvent`, `AggregationEvent`, `QuantityEvent`, `TransformationEvent` and `TransactionEvent`.
400 Bad Request	Cannot map parameter x (value: y)	The provided parameter is not supported.
400 Bad Request	Invalid value for parameter x. Expected one of y, but got: z	The value for the given parameter is not valid, because it is of an unexpected type.
400 Bad Request	maxEventCount and eventCountLimit are mutually exclusive, and cannot be provided both.	A query must specify only one of `maxEventCount` and `eventCountLimit`, or none at all. Specifying both is not allowed.
400 Bad Request	The following parameters are defined more than once, which is not allowed: x	A list of parameters that are provided more than once, which is not allowed. Most parameters support a list of values, which will be combined with OR logic and this can be used to match events with more than one value for a parameter.
400 Bad Request	Field ‘x’ is not a valid field to sort on	The provided field is not a valid field to sort on. Currently, sorting is allowed on `eventTime`,

Status code	Reason	Description
		`recordTime` or `quantity` (as defined by the EPCIS specification).
400 Bad Request	Invalid orderDirection: x. Must be either 'ASC' or 'DESC'.	The order direction is invalid. The only allowed options are `ASC` for ascending order and `DESC` for descending order. Note that this is case insensitive.
400 Bad Request	Value x is too large	The provided numerical integer value is too large.
400 Bad Request	Action x is not valid, it should be one of 'ADD', 'OBSERVE' or 'DELETE'.	The value provided when matching on an action must be either `ADD`, `OBSERVE` or `DELETE` (as defined by the EPCIS specification).
400 Bad Request	Cannot match field x on URI y. It is an invalid EPC URI.	The provided URI to match on EPC URI is not valid. It does start with `urn:epc:`, but it is not a valid URL according to the TDS standard.
400 Bad Request	Expired cursor specified	The cursor that was specified has expired (cursors will expire after one week).
400 Bad Request	A limit (x) is not allowed in combination with a cursor.	When using cursors, the number of events that are returned depends on the number of events that match and the size of those events. Therefore it is not allowed to specify a custom limit.
400 Bad Request	An order (x y) is not allowed in combination with a cursor.	When using a cursor, a fixed internal ordering is used to guarantee that all events that match appear exactly once in the total result set. Therefore, specifying a custom ordering (and direction) is not allowed.
402 Payment required	Subscription required	No valid subscription found that allows usage of the EPCIS repository.
429 Too Many Requests	Too many requests	You are doing too many requests within a short timeframe. Please wait a bit before doing the next request.
500 Internal Server Error	Internal server error	Something went wrong on our end, please try again later.

Updated about 2 years ago

READ MORE:

Introduction to EPCIS →

Capturing events →

Using cursors →

EPCIS API →

GS1 EPCIS and CBV standards ↗