CGR SAP Connector

Calling APIs

Endpoints, parameters and payloads for both product versions. All streams return UTF-8 JSON, but SICF and OData differ in URL, authentication, request body and response shape.

TECHNICAL DOCUMENTATION

Version 1.0 · September 2026

Request Conventions

Deploy the endpoint family matching your installed product version.

Area	CGR Connector (SICF)	**CGR Connector (OData)**
Base path	`/cgr/api`	`/sap/opu/odata/CGR/CONN_SRV`
Authentication	SICF service logon configuration	Typically `Authorization: Bearer <access_token>`
Table GET	`/table`	`/table('1')/$value`
Function GET	`/func`	`/func('1')/$value`
Function POST body	Array of `NAME` / `VALUE` entries	JSON object mapping parameters to values
Function POST response	Function result returned directly as JSON	OData entity with serialized JSON in `d.CONTENT`
Report GET	`/report`	`/report('1')/$value`

BEFORE YOU COPY ANYTHING

Substitute your actual host, port, client, system alias, object names and credentials. URL-encode any query value containing spaces, quotation marks or reserved characters.

Obtain An OAuth Access Token (OData)

Submit a Client Credentials request to the SAP Gateway token endpoint. Supply the OAuth client ID and secret via HTTP Basic authentication, and request the scope assigned to `/CGR/CONN_SRV`.

```
POST /sap/bc/sec/oauth2/token HTTP/1.1
Host: sap-gateway.example.com:8443
Content-Type: application/x-www-form-urlencoded
Authorization: Basic <base64(client_id:client_secret)>

grant_type=client_credentials&scope=%2FCGR%2FCONN_SRV_0001
```

A successful response includes the access token, its type, validity period and granted scope.

```
{
  "access_token": "eyJhbGciOi...",
  "token_type": "Bearer",
  "expires_in": 3600,
  "scope": "/CGR/CONN_SRV_0001"
}
```

Include the token with every OData request.

```
Authorization: Bearer <access_token>
Accept: application/json
```

SICF: Reading Tables

Base path `/cgr/api`. The connector returns the requested data directly as JSON, with no OData wrapper.

Read tables or database views

Endpoint: `GET /cgr/api/table`

Parameter	Required	**Purpose**
`id`	Yes	SAP table or database-view name, for example `VBAP` or `T001W`
`select`	No	Comma-separated list of fields to return
`filter`	No	Selection expression using comparison, string and logical operators
`top`	No	Maximum row count to return
`skip`	No	Number of rows to skip; combine with `top` for pagination
`sap-client`	No	Target SAP client
`sap-system`	No	Gateway system alias, for backend routing

```
GET https://<server>:<port>/cgr/api/table?id=VBAP&select=VBELN,POSNR,NETWR
```

SICF: Filtering And Paging

FILTER OPERATORS

Operator	Meaning	Example
`eq` , `ne`	Equals / not equals	`LAND1 eq 'US'`
`gt` , `ge`	Greater than / or equal	`ERSDA ge '20200101'`
`lt` , `le`	Less than / or equal	`ERSDA le '20231231'`
`startswith`	Text begins with value	`startswith(MAKTX, 'Plastic')`
`contains`	Text contains value	`contains(MAKTX, 'lastic')`
`AND` , `OR`	Combine expressions; parentheses group	`(MATKL eq '120' OR MATKL eq '130')`

A complex filter combines them with parentheses:

```
GET https://<server>:<port>/cgr/api/table?id=MARA&filter=(ERSDA ge '20200101' AND ERSDA le '20231231') AND (MATKL eq '120' OR MATKL eq '130')
```

Pagination combines `skip` and `top`:

```
GET https://<server>:<port>/cgr/api/table?id=MARA&select=MATNR,MTART&skip=100&top=100
```

SICF: Functions And BAPIs

Endpoint: `GET or POST /cgr/api/func`. Always include the function name in `id`. Optionally pass `sap-client`, `sap-system` and `commit`.

USE COMMIT DELIBERATELY

Set `commit=X` only when the function performs updates that require commitment.

Simple importing parameters, via GET

For functions with individual field inputs, append each input parameter to the query string.

```
GET https://<server>/cgr/api/func?id=BAPI_PO_GETDETAIL1&PURCHASEORDER=3000000010
```

SICF: Structures And Tables

The SICF POST body is a JSON array in which each element carries a parameter `NAME` and its `VALUE`. Use an object for a structure, and an array for a table parameter.

```
POST https://<server>/cgr/api/func?id=BAPISDORDER_GETDETAILEDLIST
Content-Type: application/json

[
  {
    "NAME": "I_BAPI_VIEW",
    "VALUE": { "HEADER": "X", "ITEM": "X" }
  },
  {
    "NAME": "SALES_DOCUMENTS",
    "VALUE": [
      { "VBELN": "0000004980" },
      { "VBELN": "0000004981" }
    ]
  }
]
```

READING THE SICF RESPONSE

Parse the HTTP response body once. The returned JSON is the actual function result — there is no wrapper to unpack.

SICF: ALV Reports And Queries

Endpoint: `GET /cgr/api/report`

Parameter	Required	**Purpose**
`id`	Yes	ABAP report or SAP Query program name
`variant`	Yes	Predefined variant containing the report selection values
`sap-client`	No	Target SAP client
`sap-system`	No	Target gateway system alias

```
GET https://<server>/cgr/api/report?id=RM07DOCS&variant=VAR1&sap-client=010
```

TWO CONDITIONS

The report or query must produce ALV-compatible output, and the named variant must already exist in the target SAP system.

OData: Reading Tables

Base path `/sap/opu/odata/CGR/CONN_SRV`. Add the bearer token and `Accept: application/json` to every request. GET media streams return raw JSON.

Endpoint: `GET /sap/opu/odata/CGR/CONN_SRV/table('1')/$value`

Parameter	Required	**Purpose**
`id`	Yes	SAP table or database-view name
`select`	No	Comma-separated fields to return
`filter`	No	Dynamic selection expression
`top` / `skip`	No	Page size and offset
`orderby`	No	Sorting expression (beta)
Dynamic field names	No	Additional query parameters act as equality filters, for example `LAND1=DE`

```
GET /sap/opu/odata/CGR/CONN_SRV/table('1')/$value?
id=T001W&select=WERKS,NAME1,LAND1&top=5&LAND1=DE HTTP/1.1
Host: sap-gateway.example.com:8443
Authorization: Bearer <access_token>
Accept: application/json
```

```
[
  { "WERKS": "1000", "NAME1": "Frankfurt Main Plant", "LAND1": "DE" },
  { "WERKS": "1100", "NAME1": "Munich Assembly Center", "LAND1": "DE" }
]
```

OData: Functions And BAPIs

Read-only function, via GET

Endpoint: `GET /func('1')/$value`. Pass the function name in `id` and simple importing values as additional query parameters. The `/$value` media stream returns the raw function output directly in the HTTP body.

```
GET /sap/opu/odata/CGR/CONN_SRV/func('1')/$value?id=BAPI_USER_GET_DETAIL&USERNAME=YANRB
HTTP/1.1
Host: sap-gateway.example.com:8443
Authorization: Bearer <access_token>
Accept: application/json
```

Write operations, via POST

Unlike the SICF version, the OData POST body is an object mapping function parameter names to values.

```
POST /sap/opu/odata/CGR/CONN_SRV/func?id=BAPI_USER_GET_DETAIL HTTP/1.1
Authorization: Bearer <access_token>
Content-Type: application/json

{
  "USERNAME": "YANRB",
  "CACHE_RESULTS": "X"
}
```

For updating functions requiring commitment, add `commit=X` to the query string and ensure the technical user is authorized for the business operation.

OData: Parsing Responses

ODATA POST RESPONSES REQUIRE TWO JSON PARSES

The outer document is an OData entity. Its `d.CONTENT` property is a *string* containing another JSON document — the actual function result.

```
{
  "d": {
    "ID": "1",
    "MIMETYPE": "application/json",
    "CONTENT": "{\"ADDRESS\":{\"FIRSTNAME\":\"Arthur\"},\"RETURN\":[...]}"
  }
}
```

```
// 1. Parse the Gateway OData response.
const odataResult = JSON.parse(httpResponseBody);

// 2. Extract the serialized function result.
const contentString = odataResult.d.CONTENT;

// 3. Parse it again to obtain the real function data.
const functionData = JSON.parse(contentString);

console.log(functionData.ADDRESS.FIRSTNAME);
```

Calling an ABAP report

```
GET /sap/opu/odata/CGR/CONN_SRV/report('1')/$value?
id=RSPARAM&variant=DEFAULT&skip=0&top=100 HTTP/1.1
Authorization: Bearer <access_token>
Accept: application/json
```