DEVELOPER GUIDE | PUBLIC
Document Version: 1.4.0 – 2018-09-24

SAP Tax Declaration Framework for Brazil 1.0 - Developers Guide

THE BEST RUN SAP

Content

1 Document History

The following table provides an overview of the most important document changes.

Version	Date	Description
1.00	2016-12-22	First version.
1.10	2017-03-27	The *Central Tax Repository* and the *Customer Extension* sections were migrated from the Configuration guide to the Developers' guide.
1.20	2017-07-07	The *Promote Report Run* and the *Rectify Report Run* topics were created and added to *REST OData API* under *Tax Obligation Monitor API* section.

2 About This Guide

You use this guide to find technical information about TDF implementation. This guide includes information about the REST OData and ABAP of Tax Obligation Monitor API.

This guide is the central source of information specific to Brazilian localization for developers who need to implement *SAP Tax Declaration Framework for Brazil 1.0* in their systems and need information about customer extensions, API extensions, extensions for CTR views and details of storage provider for big files.

3 Tax Obligation Monitor API

`Tax Obligation Monitor API` is the central point to keep tracking of all legal obligations within TDF solution. This API allows you to create and to manage report runs as well as to store report files.

Overview

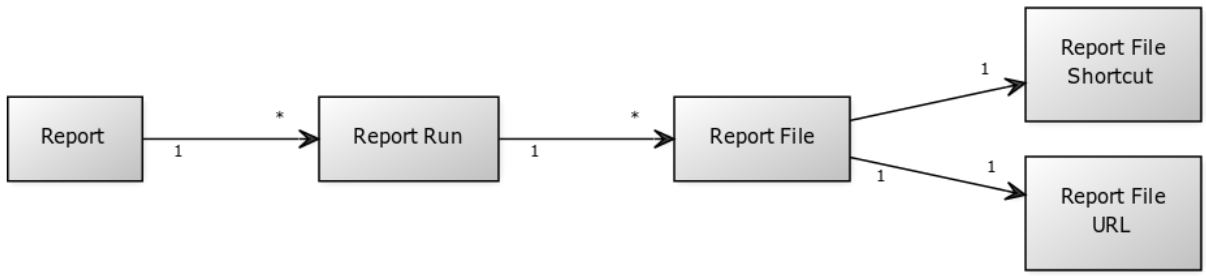
OData Version: 2.0

Support of OData Features

Feature	Support
Query options	The current implementation supports query options that are considered as query or as path parameters, as following: • $select • $count • $expand • $format • $filter • $orderby

Entity Data Model

You can visualize the Entity Data Model of TOM API service in the graphic below:



Service Metadata URI: `https://%3Ctdfhost%3E/sap/opu/odata/TMF/TOM_API_SRV/$metadata`

Resources

Resource	Description	Path
Report [page 6]	Report	`/reports`
Report Run [page 9]	Report Run managed by TOM	`/report_runs`
Rectify Report Run [page 21]	Rectification of Report Run	`/report_historys`
Report Files [page 24]	Report File that belongs to a Report Run	`/report_files`
Report File URL [page 30]	URL in which you can download a Report File	`/report_file_urls`
Report File Shortcut [page 39]	Shortcut to download a Report File using SAP GUI	`/report_file_shortcuts`

3.1 REST OData API

Tax Obligation Monitor API is available as a REST Odata service to give access and permissions for the creation of TOM entities, such as `Report Runs` and `Report Files`.

REST Tax Obligation Monitor API implements OData protocol in order to enable flexible query options.

3.1.1 Report

Report is a part of the Tax Obligation Monitor API that represents a fiscal report managed by TDF.

Resource Path:`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/reports`

Operations

CRUD Operations

HTTP Method	Operation	URI
GET	Get Reports [page 7]	`/reports`
GET	Get Report Entity [page 8]	`/reports({report ID})`

3.1.1.1 Get Reports

In this section, you find information about how to retrieve fiscal reports.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/reports`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
report_id	No	Int64	Report ID	Query string, $filter
report_name	No	String	Report name	Query string, $filter
report_level	No	String	Report level	Query string, $filter
report_acronym	No	String	Report acronym	Query string, $filter

Request Example

```
[GET] https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/reports?$filter=report_id
le 2
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON array containing reports
500	General error	Payload containing an error message

Response Payload Example

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs",
          "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs",
          "type": "/TMF/TOM_API_SRV.report"
```

```
        },
        "report_id": "1",
        "report_name": "SPED EFD ICMS IPI (EFD)",
        "report_level": "State",
        "report_acronym": "EFD"
      },
      {
        "__metadata": {
          "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs",
          "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs",
          "type": "/TMF/TOM_API_SRV.report"
        },
        "report_id": "2",
        "report_name": "SPED Escrituração Contábil Digital (ECD)",
        "report_level": "Federal",
        "report_acronym": "ECD"
      }
    ]
  }
}
```


3.1.1.2 Get Report Entity

In this section, you find information about how to retrieve details about a fiscal report.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/REPORTS({Report ID})`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`report_id`	Yes	Int64	Report ID	OData unique key

Request Example

```
[GET] https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/reports(2)
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON containing a report
400	Not found	No payload response
500	General error	Payload containing an error message

Response Payload Example

```
{
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs",
      "type": "/TMF/TOM_API_SRV.report"
    },
    "report_id": "2",
    "report_name": "SPED Escrituração Contábil Digital (ECD)",
    "report_level": "Federal",
    "report_acronym": "ECD"
  }
}
```

3.1.2 Report Run

In this section, you find information about how to create, delete, get and update report run entities.

Resource Path: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs`

Operations

CRUD Operations

HTTP Method	Operation	URI
GET	Get Report Runs [page 10]	`/reports_runs`
GET	Get Report Run Entity [page 13]	`/reports_runs({Run ID})`
POST	Create Report Run Entity [page 14]	`/reports_runs`
PUT	Update Report Run Entity [page 17]	`/reports_runs({Run ID})`
PUT	Promote Report Run [page 19]	`/report_historys`
DELETE	Delete Report Run Entity [page 20]	`/reports_runs({Run ID})`

3.1.2.1 Get Report Runs

In this section, you find information about how to get report runs.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
run_id	No	String[32]	Run ID	Query string, $filter
report_id	No	String[10]	Report ID	Query string, $filter
report_key	No	String[255]	Report Key	Query string, $filter
company_code	No	String[4]	Company Code	Query string, $filter
branch	No	String[4]	Branch	Query string, $filter
report_status	No	String[1]	Report Status	Query string, $filter
report_status_text	No	String[30]	Report Status Text	Query string, $filter
initial_date	No	DateTime	Initial Date	Query string, $filter
final_date	No	DateTime	Final Date	Query string, $filter
submitted_date	No	DateTime	Submitted Date	Query string, $filter
run_type	No	String[1]	Run Type	Query string, $filter
run_type_text	No	String[30]	Run Type Text	Query string, $filter
created_by	No	String[12]	Created By	Query string, $filter
created_on	No	DateTime	Created On	Query string, $filter
orgstr_key	No	String[5]	Orgstr Key	Query string, $filter
ie	No	String[18]	IE	Query string, $filter
cnpj_root	No	String[8]	CNPJ Root	Query string, $filter

Parameter	Required	Data Type	Description	Parameter Type
central_efd	No	String[20]	Central EFD	Query string, $filter
changed_on	No	DateTime	Changed On	Query string, $filter
description	No	String[255]	Description	Query string, $filter
rectification_run_id	No	String[32]	Rectification Run ID	Query string, $filter
source_run_id	No	String[32]	Source Run ID	Query string, $filter

Request Example

```
[GET] https://<uri>/report_runs?$filter=company_code eq '7035' and branch eq
'0001'
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON array containing report runs
500	General error	Payload containing an error message

Response Payload Example

```
 {
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED69D8610B4D2AFC67F')",
          "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED69D8610B4D2AFC67F')",
          "type": "/TMF/TOM_API_SRV.report_run"
        },
        "run_id": "5CF3FCDD605A1ED69D8610B4D2AFC67F",
        "report_id": "0000000003",
        "report_key": "",
        "company_code": "7035",
        "branch": "0001",
        "report_status": "N",
        "report_status_text": "Created",
        "initial_date": "/Date(1420070400000)/",
        "final_date": "/Date(1451520000000)/",
        "submitted_date": null,
        "run_type": "0",
        "run_type_text": "Original",
        "created_by": "C5228554",
        "created_on": "/Date(1473166243000)/",
```

```
        "orgstr_key": "ROOT",
        "ie": "",
        "cnpj_root": "07084117",
        "central_efd": "",
        "changed_on": "/Date(1473166347000)/",
        "description": "SPED Escrituração Contábil Fiscal (ECF)  - 06.09.2016
12:50",
        "rectification_run_id": "",
        "source_run_id": "",
        "files": {
          "__deferred": {
            "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED69D8610B4D2AFC67F')/files"
          }
        }
      },
      {
        "__metadata": {
          "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')",
          "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')",
          "type": "/TMF/TOM_API_SRV.report_run"
        },
        "run_id": "5CF3FCDD605A1ED59F9AE267FEF6F5CC",
        "report_id": "0000000002",
        "report_key": "",
        "company_code": "7035",
        "branch": "0001",
        "report_status": "E",
        "report_status_text": "Error",
        "initial_date": "/Date(1391212800000)/",
        "final_date": "/Date(1393545600000)/",
        "submitted_date": null,
        "run_type": "2",
        "run_type_text": "Draft",
        "created_by": "C5196653",
        "created_on": "/Date(1445969844000)/",
        "orgstr_key": "CC",
        "ie": "",
        "cnpj_root": "",
        "central_efd": "",
        "changed_on": "/Date(1445969961000)/",
        "description": "SPED Contábil (ECD) 7035 02-02/2014 - 27.10.2015 18:17",
        "rectification_run_id": "",
        "source_run_id": "",
        "files": {
          "__deferred": {
            "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')/files"
          }
        }
      }]
 }
}
```

3.1.2.2 Get Report Run Entity

In this section, you find information about how to get a report run entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs({Run ID})`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
report_id	Yes	String[32]	Report ID	[Path | Query string | Request body | ...]

Request Example

```
[GET] https://<uri>/report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON containing a report
404	Not found	No response payload
500	General error	No response payload

Response Payload Example

```
{
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')",
      "type": "/TMF/TOM_API_SRV.report_run"
    },
    "run_id": "5CF3FCDD605A1ED59F9AE267FEF6F5CC",
    "report_id": "0000000002",
    "report_key": "",
    "company_code": "7035",
    "branch": "0001",
    "report_status": "E",
    "report_status_text": "Error",
```

```
    "initial_date": "/Date(1391212800000)/",
    "final_date": "/Date(1393545600000)/",
    "submitted_date": null,
    "run_type": "2",
    "run_type_text": "Draft",
    "created_by": "C5196653",
    "created_on": "/Date(1445969844000)/",
    "orgstr_key": "CC",
    "ie": "",
    "cnpj_root": "",
    "central_efd": "",
    "changed_on": "/Date(1445969961000)/",
    "description": "SPED Contábil (ECD) 7035 02-02/2014 - 27.10.2015 18:17",
    "rectification_run_id": "",
    "source_run_id": "",
    "files": {
      "__deferred": {
        "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1ED59F9AE267FEF6F5CC')/files"
      }
    }
  }
}
```


3.1.2.3 Create Report Run Entity

In this section, you find information about how to create a report run entity.

Request

URI: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs`**

Operation Type: *CRUD*

HTTP Method: *POST*

Request Parameters

Parameter	Required	Data Type	Description/Values	Parameter Type
`run_id`	No	String[32]	Run ID	Request Body
`report_id`	Yes	String[10]	Report ID	Request Body
`report_key`	Yes	String[255]	Report Key	Request Body
`company_code`	Yes	String[4]	Company Code	Request Body
`branch`	Yes	String[4]	Branch	Request Body
`report_status`	Yes	String[1]	Report Status	Request Body

Parameter	Required	Data Type	Description/Values	Parameter Type
report_status_text	Yes	String[30]	Report Status Text	Request Body
initial_date	No	DateTime	Initial Date	Request Body
final_date	No	DateTime	Final Date	Request Body
submitted_date	No	DateTime	Submitted Date	Request Body
run_type	Yes	String[1]	Run Type	Request Body
run_type_text	Yes	String[30]	Run Type Text	Request Body
created_by	Yes	String[12]	Created By	Request Body
created_on	No	DateTime	Created On	Request Body
orgstr_key	Yes	String[5]	Orgstr Key	Request Body
ie	No	String[18]	IE	Request Body
cnpj_root	No	String[8]	CNPJ Root	Request Body
central_efd	No	String[20]	Central EFD	Request Body
changed_on	No	DateTime	Changed On	Request Body
description	Yes	String[255]	Description	Request Body
rectification_run_id	Yes	String[32]	Rectification Run ID	Request Body
source_run_id	Yes	String[32]	Source Run ID	Request Body

Request Example

```
[POST] https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs
{
    "report_id": "0000000003",
    "report_key": "",
    "company_code": "BR10",
    "branch": "SCR",
    "report_status": "H",
    "report_status_text": "Completed",
    "initial_date": "/Date(1412121600000)/",
    "final_date": "/Date(1414627200000)/",
    "submitted_date": "/Date(1478822400000)/",
    "run_type": "2",
    "run_type_text": "Draft",
    "created_on": "/Date(1421261994000)/",
    "orgstr_key": "ROOT",
    "ie": "",
    "cnpj_root": "07358761",
    "central_efd": "",
    "changed_on": "/Date(1478881182000)/",
    "description": "CAUS",
    "rectification_run_id": "",
```

```
    "source_run_id": ""
}
```

Response

Response Status and Error Codes

Code	Reason	Description
201	Success	A JSON containing a report
500	General Error	Payload containing an error message

Response Payload Example

```
 {
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1EE6ACE66040070D7246')",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1EE6ACE66040070D7246')",
      "type": "/TMF/TOM_API_SRV.report_run"
    },
    "run_id": "5CF3FCDD605A1EE6ACE66040070D7246",
    "report_id": "0000000003",
    "report_key": "|ECF|07358761|1||2014|D|20161125171713|",
    "company_code": "BR10",
    "branch": "SCR",
    "report_status": "H",
    "report_status_text": "Scheduled",
    "initial_date": "/Date(1412121600000)/",
    "final_date": "/Date(1414627200000)/",
    "submitted_date": "/Date(1478822400000)/",
    "run_type": "2",
    "run_type_text": "Draft",
    "created_by": "BICAM",
    "created_on": "/Date(1480094233000)/",
    "orgstr_key": "ROOT",
    "ie": "",
    "cnpj_root": "07358761",
    "central_efd": "",
    "changed_on": "/Date(1480094233000)/",
    "description": "CAUS",
    "rectification_run_id": "",
    "source_run_id": "",
    "files": {
      "__deferred": {
        "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_runs('5CF3FCDD605A1EE6ACE66040070D7246')/files"
      }
    }
  }
}
```

3.1.2.4 Update Report Run Entity

In this section, you find information about how to update a report run entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_run({Run ID})`

Operation Type: *CRUD*

HTTP Method: *PUT*

Request Parameters

Parameter	Required	Data Type	Description/Values	Parameter Type
run_id	Yes	String[32]	Run ID	OData Unique key
report_id	Yes	String[10]	Report ID	Request Body
report_key	Yes	String[255]	Report Key	Request Body
company_code	Yes	String[4]	Company Code	Request Body
branch	Yes	String[4]	Branch	Request Body
report_status	Yes	String[1]	Report Status	Request Body
report_status_text	Yes	String[30]	Report Status Text	Request Body
initial_date	Yes	DateTime	Initial Date	Request Body
final_date	Yes	DateTime	Final Date	Request Body
submitted_date	Yes	DateTime	Submitted Date	Request Body
run_type	Yes	String[1]	Run Type	Request Body
run_type_text	Yes	String[30]	Run Type Text	Request Body
created_by	Yes	String[12]	Created By	Request Body
created_on	Yes	DateTime	Created On	Request Body
orgstr_key	Yes	String[5]	Orgstr Key	Request Body
ie	Yes	String[18]	IE	Request Body
cnpj_root	Yes	String[8]	CNPJ Root	Request Body

Parameter	Required	Data Type	Description/Values	Parameter Type
central_efd	Yes	String[20]	Central EFD	Request Body
changed_on	No	DateTime	Changed On	Request Body
description	Yes	String[255]	Description	Request Body
rectification_run_id	Yes	String[32]	Rectification Run ID	Request Body
source_run_id	Yes	String[32]	Source Run ID	Request Body

Request Example

```
[PUT] https://<uri>/report_runs('5CF3FCDD605A1ED692A877161974FD25')
{
    "run_id": "5CF3FCDD605A1EE6ACE66040070D7246",
    "report_id": "0000000003",
    "report_key": "",
    "company_code": "BR10",
    "branch": "SCR",
    "report_status": "H",
    "report_status_text": "Completed",
    "initial_date": "/Date(1412121600000)/",
    "final_date": "/Date(1414627200000)/",
    "submitted_date": "/Date(1478822400000)/",
    "run_type": "2",
    "run_type_text": "Draft",
    "created_on": "/Date(1421261994000)/",
    "orgstr_key": "ROOT",
    "ie": "",
    "cnpj_root": "07358761",
    "central_efd": "",
    "description": "CAUS",
    "rectification_run_id": "",
    "source_run_id": ""
}
```

Response

Response Status and Error Codes

Code	Reason	Description
204	Success	Nothing is returned
500	General error	Payload containing an error message

Response Payload Example

```
Nothing is returned.
```

3.1.2.5 Promote Report Run

In this section, you find information about how you can promote your report from Draft to Official.

> **i** Note
>
> You can only promote a report run with status *Created* and in which the run type is set as *Draft*.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_run({Run ID})`

Operation Type: *CRUD*

HTTP Method: *PUT*

Request Parameters

Header	Required	Data Type	Values	Parameter Type
`run_id`	Yes	String[32]	Run ID	oData unique key
`run_type_text`	Yes	String[30]	Promote	oData unique key

Request Example

```
String[32[PUT]https://<uri>/report_runs('5CF3FCDD605A1ED788EE33C4FE48FDBF')
{
    "run_id":"5CF3FCDD605A1ED788EE33C4FE48FDBF",
    "run_type_text":"Promote"
}
```

Response

Response Status and Error Codes

Code	Reason	Description
204	Success	No content

Code	Reason	Description
400	Error	Examples of possible error messages related to Tax Obligation Monitor API: • If the current report status is Promoting, you get the following message: "This report is already being promoted. Enter a different report key." • If the report status is different from *Created* and the run type is different from *Draft*, you get the following message: "You can only promote report runs from Draft to Official."
500	General error	Payload containing an error message

3.1.2.6 Delete Report Run Entity

In this section, you find information about how to delete a report run entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_runs({Run ID})`

Operation Type: *CRUD*

HTTP Method: *DELETE*

Request Parameters

Parameter	Required	Data Type	Description/Values	Parameter Type
`run_id`	Yes	String[32]	Report run ID	Query string

Request Example

```
[DELETE] https://<uri>/report_runs('5CF3FCDD605A1ED692A877161974FD25')
```

Response

Response Status and Error Codes

Code	Reason	Description
204	Success	Nothing is returned
500	General Error	Payload containing an error message

Response Payload Example

```
Nothing is returned.
```

3.1.3 Rectify Report Run

Report runs can be rectified after you have executed a SPED report, with data saved in the SPED Historical Master Data (`/TMF/D_SPED_HIST`) table, in which status is set as *Completed*.

Overview

You rectify a report run when you have to edit an official run with status set as *Completed* in Tax Obligation Monitor, which has the same parameters as the original execution. The new execution uses the data stored in the SPED Historical Master Data (`/TMF/D_SPED_HIST`) table to generate the new official run.

OData Version: 2.0

Root URI: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_historys`**

Permissions: You must have permissions in the following authorization objects: /TMF/ECD01, /TMF/ECF01, /TMF/EFD01 and /TMF/PCO01.

Support of OData Features

Feature	Support
Query options	The current implementation supports the *$filter* query option, that is considered as query or path parameter.

3.1.3.1 Get History of Report Run

In this section, you find information about how to get the history data of your official report run.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_historys`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`run_id`	Yes	String[32]	Run ID that you want to get the history data	Query string, $filter
`row_id`	No	String[10]	Row number of the SPED file	Query string, $filter
`reg_name`	No	String[4]	Register name	Query string, $filter

Request Example

```
[GET]https://<uri>/ report_historys?$filter=run_id eq
'5CF3FCDD605A1ED791AE5A48A981AB72' and reg_name eq '0305'
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success result	A JSON array containing report history data.
400	Bad bequest	Payload containing an error message.
500	General error	Payload containing an error message.

Response Payload Example

```
{
  "d" : {
    "results" : [
      {
        "__metadata" : {
          "id" : "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/TMF/
TOM_API_SRV/
```

```
report_historys(run_id='5CF3FCDD605A1ED791AE5A48A981AB72',row_id='%20%20%20%20%
%20%2013')",
          "uri" : "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/TMF/
TOM_API_SRV/
report_historys(run_id='5CF3FCDD605A1ED791AE5A48A981AB72',row_id='%20%20%20%20%
%20%2013')",
          "type" : "/TMF/TOM_API_SRV.report_history"
        },
        "run_id" : "5CF3FCDD605A1ED791AE5A48A981AB72",
        "row_id" : "       13",
        "reg" : "|0305||DESCR|120|",
        "reg_name" : "0305"
      },
      {
        "__metadata" : {
          "id" : "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/TMF/
TOM_API_SRV/
report_historys(run_id='5CF3FCDD605A1ED791AE5A48A981AB72',row_id='%20%20%20%20%
%20%2045')",
          "uri" : "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/TMF/
TOM_API_SRV/
report_historys(run_id='5CF3FCDD605A1ED791AE5A48A981AB72',row_id='%20%20%20%20%
%20%2045')",
          "type" : "/TMF/TOM_API_SRV.report_history"
        },
        "run_id" : "5CF3FCDD605A1ED791AE5A48A981AB72",
        "row_id" : "       45",
        "reg" : "|0305|DESCR 2|11001|110|",
        "reg_name" : "0305"
      }
    ]
  }
}
```


3.1.3.2 Change Report Master Data

In this section, you find information about how to change the master data saved in the SPED Historical Master Data (`/TMF/D_SPED_HIST`) table, to be able to rectify a report run.

Request

URI: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_rectifys`**

Operation Type: *CRUD*

HTTP Method: *PUT*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`run_id`	Yes	String[32]	Run ID that you want to get the history data	OData Unique key
`row_id`	Yes	String[10]	Row Number of the SPED file	Request Body

Parameter	Required	Data Type	Description	Parameter Type
new_line	Yes	String[1333]	Register	Request Body

Request Example

```
[PUT]https://<uri>/sap/opu/odata/TMF/TOM_API_SRV/
report_rectifys(run_id='5CF3FCDD605A1ED791AE5A48A981AB72')
{
"run_id": "5CF3FCDD605A1ED791AE5A48A981AB72",
"row_id": "13",
"new_line": "|0305||DESCR 2|122|"
}
```

Response

Response Status and Error Codes

Code	Reason	Description
204	Success	No content.
400	Bad request	Payload containing an error message.
500	General error	Payload containing an error message.

Response Payload Example

Nothing is returned.

3.1.4 Report Files

In this section, you find information about how to create, delete and get report files.

Resource Path: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files`

Operations

CRUD Operations

HTTP Method	Operation	URI
GET	Get Report Files [page 25]	/report_files
GET	Get Report File Entity [page 27]	/report_files ({File ID})
POST	Create Report File Entity [page 28]	/report_files

HTTP Method	Operation	URI
DELETE	Delete Report File Entity [page 30]	`/report_files ({File ID})`

3.1.4.1 Get Report Files

In this section, you find information about how to get a report file.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`file_id`	No	String[32]	File ID	OData unique key
`run_id`	No	String[32]	Run ID	Query string, $filter
`file_type`	No	String[1]	Fiscal report file type	Query string, $filter
`file_type_text`	No	String[30]	File type text	Query string, $filter
`file_description`	No	String[100]	File description	Query string, $filter
`file_name`	No	String[100]	File name	Query string, $filter
`file_created_on`	No	DateTime	File creation date	Query string, $filter
`file_manual`	No	String[1]	File manually uploaded	Query string, $filter

Request Example

```
[GET] https://<uri>/report_files
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON array containing report files
500	General error	Payload containing an error message

Response Payload Example


```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1EE6A0B6D54093EBA34A')",
          "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1EE6A0B6D54093EBA34A')",
          "type": "/TMF/TOM_API_SRV.report_file"
        },
        "file_id": "5CF3FCDD605A1EE6A0B6D54093EBA34A",
        "run_id": "5CF3FCDD605A1EE6A0B6D536C8E3634A",
        "file_type": "A",
        "file_type_text": "Application file",
        "file_description": "File generated automatically",
        "file_name": "SPED EFD ICMS IPI (EFD) - 23/09/2016 - 17:53.txt",
        "file_created_on": "/Date(1474653222000)/",
        "file_manual": "",
        "mime_type": "",
        "url": {
          "__deferred": {
            "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1EE6A0B6D54093EBA34A')/url"
          }
        }
      },
      {
        "__metadata": {
          "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('005056912D7F1EE3BDC44EB16170891C')",
          "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('005056912D7F1EE3BDC44EB16170891C')",
          "type": "/TMF/TOM_API_SRV.report_file"
        },
        "file_id": "005056912D7F1EE3BDC44EB16170891C",
        "run_id": "005056912D7F1EE3989EBC39ABF02DDD",
        "file_type": "X",
        "file_type_text": "Auxiliary file",
        "file_description": "teste4",
        "file_name": "debug.txt",
        "file_created_on": "/Date(1403018561000)/",
        "file_manual": "X",
        "mime_type": "",
        "url": {
          "__deferred": {
            "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('005056912D7F1EE3BDC44EB16170891C')/url"
          }
        }
      }
    ]
  }
```

```
}
```

3.1.4.2 Get Report File Entity

In this section, you find information about how to get a report file entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_file({File ID})`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`file_id`	Yes	String[32]	File ID	OData unique key

Request Example

```
[GET] https://<uri>/report_files('5CF3FCDD605A1ED6ACB2918CB232990D')
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON containing a report file
500	General error	Payload containing an error message

Response Payload Example

```
{
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1ED6ACB2918CB232990D')",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1ED6ACB2918CB232990D')",
      "type": "/TMF/TOM_API_SRV.report_file"
    },
    "file_id": "5CF3FCDD605A1ED6ACB2918CB232990D",
    "run_id": "5CF3FCDD605A1ED6ACB29115C171790D",
    "file_type": "A",
    "file_type_text": "Application file",
```

```
    "file_description": "File generated automatically",
    "file_name": "SPED EFD ICMS IPI (EFD) - 23/11/2016 - 15:50.txt",
    "file_created_on": "/Date(1479916224000)/",
    "file_manual": "",
    "mime_type": "",
    "url": {
      "__deferred": {
        "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1ED6ACB2918CB232990D')/url"
      }
    }
  }
}
```

3.1.4.3 Create Report File Entity

In this section, you find information about how to create a report file entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files`

Operation Type: *CRUD*

HTTP Method: *POST*

Request Parameters

Parameter	Required	Data Type	Description/ Values	Parameter Type
`file_id`	Yes	String[32]	File ID	Request body
`run_id`	Yes	String[32]	Run ID	Request body
`file_type`	Yes	String[1]	Fiscal report file type	Request body
`file_description`	Yes	String[100]	File description	Request body
`file_name`	No	String[100]	File name	Request body
`file_manual`	No	String[1]	File uploaded manually	Request body

Request Example

```
[POST] https://<uri>/report_files
{
    "file_id": "",
```

```
    "run_id": "5CF3FCDD605A1ED6ACB29115C171790D",
    "file_type": "D",
    "file_description": "File generated automatically",
    "file_name": "SPED EFD ICMS IPI (EFD) - 23/11/2016 - 15:50.txt",
    "file_manual": "",
    "mime_type": ""
}
```

Response

Response Status and Error Codes

Code	Reason	Description
201	Success	A JSON containing a report file
500	General error	Payload containing an error message

Response Payload Example

```
{
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1ED6ACC818CE5CB63977')",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1ED6ACC818CE5CB63977')",
      "type": "/TMF/TOM_API_SRV.report_file"
    },
    "file_id": "5CF3FCDD605A1ED6ACC818CE5CB63977",
    "run_id": "5CF3FCDD605A1ED6ACB29115C171790D",
    "file_type": "D",
    "file_type_text": "",
    "file_description": " File generated automatically",
    "file_name": "SPED EFD ICMS IPI (EFD) - 23/11/2016 - 15:50.txt",
    "file_created_on": "/Date(1479990195000)/",
    "file_manual": "",
    "mime_type": "",
    "url": {
      "__deferred": {
        "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files('5CF3FCDD605A1ED6ACC818CE5CB63977')/url"
      }
    }
  }
}
```

3.1.4.4 Delete Report File Entity

In this section, you find information about how to delete a report file entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files({File ID})`

Operation Type: *CRUD*

HTTP Method: *DELETE*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`file_id`	Yes	String[32]	Fiscal report file type	OData unique key

Request Example

```
[DELETE] https://<uri>/report_files ('5CF3FCDD605A1EE6ACCB77CA3E442374')
}
```

Response

Response Status and Error Codes

Code	Reason	Description
204	Success	Nothing is returned
500	General error	Payload containing an error message.

Response Payload Example

```
 Nothing is returned.
```

3.1.5 Report File URL

In this section, you find information about how to represents a report file URL.

Resource Path:`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_urls`

Operations

CRUD Operations

HTTP Method	Operation	URI
GET	Get Report Files URL Entity [page 31]	`/report_files_urls ({File ID})`
GET	Download Report File [page 32]	`/report_files_urls ({File ID})/$value`
GET	Download Report File in Chunks [page 33]	`/report_files_urls ({File ID})/$value`
POST	Upload Report Files [page 34]	`/report_files_urls`
POST	Upload Report File in Chunks [page 35]	`/report_files_urls`

3.1.5.1 Get Report Files URL Entity

In this section, you find information about how to get a report file URL entity.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_urls({File ID})`

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`file_id`	Yes	String[32]	File ID	OData unique key

Request Example

```
[GET] https://<uri>/report_files_urls('5CF3FCDD605A1ED6AACCBDA4A94A1D1A')
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON containing a file URL
500	General Error	Payload containing an error message

Response Payload Example

```
{
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('')",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('')",
      "type": "/TMF/TOM_API_SRV.report_file_url",
      "content_type": "application/octet-stream",
      "media_src": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('')/$value"
    },
    "file_id": "",
    "url": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1ED6AACCBDA4A94A1D1A')/$value",
    "mime_type": ""
  }
}
```

3.1.5.2 Download Report File

In this section, you find information about how to download a report file.

Request

URI: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_urls({File ID})/$value`**

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`file_id`	Yes	String[32]	File ID	OData unique key

Request Example

```
[GET] https://<uri>/report_files_urls('5CF3FCDD605A1ED6AACCBDA4A94A1D1A')/$value
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	Return the requested file
404	Not found	File ID not found
500	General error	Payload containing error message

3.1.5.3 Download Report File in Chunks

In this section, you find information about how to download a report file in chunks, in case of big files.

Request

URI: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_urls({File ID})/$value`**

Operation Type: *CRUD*

HTTP Method: *GET*

Request Headers

Header	Required	Values
`download-chunk`	Yes	Number of file chunks to be retrieved from Tax Obligation Monitor. E.g. 1

Request Parameters

Parameter	Required	Data Type	Description	Parameter Type
`file_id`	Yes	String[32]	File ID	OData unique key

Request Example

```
[GET] https://<uri>/report_files_urls('5CF3FCDD605A1ED6AACCBDA4A94A1D1A')/$value
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	Return of file chunk requested
400	Bad Request	Download chunk doesn't exist
404	Not Found	File ID not found
500	General Error	Payload containing an error message

3.1.5.4 Upload Report Files

In this section, you find information about how to upload a report file.

Request

URI: `https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_urls`

Operation Type: *CRUD*

HTTP Method: *POST*

Request Headers

Header	Required	Values
`slug`	Yes	Header with metadata information about the file that is being uploaded

Slug Fields

The slug header contains metadata about the file uploaded to TOM. The slug fields should be separated by pipes.

Field	Required	Data Type	Description/Values
Report Run ID	Yes	String[32]	File ID

Field	Required	Data Type	Description/Values
File Type	Yes	String[1]	A chart representing the file type. The possible values are the following: • A - Application file • S - Submitted file • X - Auxiliary file
File Description	Yes	String[100]	File description
File Name	Yes	String[100]	File name

Slug Example

```
5CF3FCDD605A1EE6AEF5B901C0A3EC00Z|X|auxiliar file|auxiliar.txt
```

Request Example

```
[POST] https://<uri>/report_files_urls
```

Response

Response Status and Error Codes

Code	Reason	Description
201	Created	Payload containing a success message
400	Bad Request	One of the parameters in slug has an invalid value
404	Not Found	Report Run informed in Slug not found
500	General error	Payload containing an error message

3.1.5.5 Upload Report File in Chunks

In this section, you find information about how to upload a report file in parts, in case of big files.

Request

URI: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_urls`**

Operation Type: *CRUD*

HTTP Method: *POST*

Request Headers

Header	Required	Values
`slug`	Yes	When the upload session starts, you should send this header with the metadata information about the file that is being uploaded. After that, the `slug` should contain the `file_id` that is generated at the beggining of the upload session.
`upload-session`	Yes	This header manages the upload session. The following values are valid : • Start: Starts the upload session. • Append: The body is attached to an existing file. • Finish: Ends the upload session.

Slug Field for Upload Start

Field	Required	Data Type	Description/Values
`Report Run ID`	Yes	String[32]	Report Run ID
`File Type`	Yes	String[1]	A chart representing the file type. The possible values are the following: • A - Application file • S - Submitted file • X - Auxiliary file
`File Description`	Yes	String[100]	File Description
`File Name`	Yes	String[100]	File Name

Slug Example:

```
5CF3FCDD605A1EE6AEF5B901C0A3EC00Z|X|auxiliar file|auxiliar.txt
```

Slug Fields for Appending and Finishing Upload

To append data to a file or to finish the upload session, the slug header must contain the `File ID` generated when you started the upload session.

Field	Required	Data Type	Description/Values
`File ID`	Yes	String[32]	File ID generated in the start session

Slug Example:

```
5CF3FCDD605A1EE6AF8FFF69990E3A8F
```

Request Example:

```
[POST] https://<uri>/report_files_urls
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Created	Payload containing a success message
400	Bad Request	One of the parameters in slug has an invalid value
404	Not Found	Report Run informed in Slug not found
500	General Error	Payload containing an error message

Example of a Complete Upload

You can upload a file in chunks using three http requests. In the example below, the file content is sent as plain text in the request body. However, you can algo send a file content as binary data, such as an image for example.

1. Start the file upload.
   `[POST] https://<uri>/report_files_urls`

   Headers

Header	Values
`download-chunk`	Start
`slug`	Get report run entity

   Request Body
   `Chunk 1`

Sample Code

Response


```
{
  "d": {
    "__metadata": {
      "id": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')",
      "uri": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')",
      "type": "/TMF/TOM_API_SRV.report_file_url",
      "content_type": "application/octet-stream",
      "media_src": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')/$value"
    },
    "file_id": "5CF3FCDD605A1EE6AF912C8DDDE2BE02",
    "url": "https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')/$value",
    "mime_type": ""
  }
}
```


2. Append data.
   `[POST] https://<uri>/report_files_urls`

   Headers

Header	Values
upload-session	Append
slug	5CF3FCDD605A1EE6AF912C8DDDE2BE02

Request Body

`Chunk 2`

Sample Code

Response


```
{
  "d": {
    "__metadata": {
      "id": "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/tmf/
TOM_API_SRV/report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')",
      "uri": "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/tmf/
TOM_API_SRV/report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')",
      "type": "/TMF/TOM_API_SRV.report_file_url",
      "content_type": "application/octet-stream",
      "media_src": "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/tmf/
TOM_API_SRV/report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')/$value"
    },
    "file_id": "5CF3FCDD605A1EE6AF912C8DDDE2BE02",
    "url": "/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')/$value",
    "mime_type": ""
  }
}
```

3. Finish the file upload.
   `[POST] https://<uri>/report_files_urls`

   Headers

Header	Values
`upload-session`	Finish
`slug`	`5CF3FCDD605A1EE6AF912C8DDDE2BE02`

Request Body

`Chunk 3`

Sample Code

Response

```
{
  "d": {
    "__metadata": {
      "id": "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/tmf/
TOM_API_SRV/report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')",
      "uri": "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/tmf/
TOM_API_SRV/report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')",
      "type": "/TMF/TOM_API_SRV.report_file_url",
      "content_type": "application/octet-stream",
      "media_src": "https://ldcix2m.wdf.sap.corp:44314/sap/opu/odata/tmf/
TOM_API_SRV/report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')/$value"
    },
    "file_id": "5CF3FCDD605A1EE6AF912C8DDDE2BE02",
    "url": "/sap/opu/odata/TMF/TOM_API_SRV/
report_files_urls('5CF3FCDD605A1EE6AF912C8DDDE2BE02')/$value",
    "mime_type": ""
  }
}
```

3.1.6 Report File Shortcut

In this section, you find information about how to represents a report file shortcut.

Resource Path: **`https://<tdfhost>/sap/opu/odata/TMF/TOM_API_SRV/report_files_shortcuts`**

Operations

CRUD Operations

HTTP Method	Operation	URI
GET	Get Report Files Shortcut Entity [page 40]	`/report_files_shortcuts ({File ID})`

3.1.6.1 Get Report Files Shortcut Entity

In this section, you find information about how to get a report file shortcut entity.

Request

URI: **`https://<tdf_host>/sap/opu/odata/TMF/TOM_API_SRV/report_files_shortcuts({File ID})/$value`**

Operation Type: *CRUD*

HTTP Method: *GET*

Request Parameters

Parameter	Required	Data Type	Description/Values	Parameter Type
`file_id`	Yes	String[32]	File ID	OData unique key

Request Example

```
[GET] https://<uri>/report_files_shortcuts('5CF3FCDD605A1ED6AACCBDA4A94A1D1A')/
$value
```

Response

Response Status and Error Codes

Code	Reason	Description
200	Success	A JSON array containing a report file
500	General error	Payload containing an error message

Response Payload Example

```
A file is returned with a shortcut to SAP Gui.
```

3.2 ABAP API

Tax Obligation Monitor API is also available for ABAP programs, to give access and permissions for the creation of TOM entities, such as `Report Runs` and `Report Files`.

ABAP Tax Obligation Monitor API provides classes and methods to custom your application with Tax Obligation Monitor functionalities.

3.2.1 Report Key: Customer Extension

Report Key is a unique identifier that validates the creation of the SPED reports for an official run, which can be extended to custom the reports.

You can custom the reports by following the steps below:

1. Create a report key class that inherits from the `/TMF/CL_REPORT_KEY` class.
2. Redefine the `MOUNT_REPORT_KEY_HOOK` method with the custom definition, according to the following rule:
   - Enter `|` as the first and the last characters.
   - Separate all fields with `|`.
     Example:
     Final Report Key: *|Report Acronym|Field 1|Field 2|Field 3|Field 4|Field 5|Field 6|Field 7|D|20161111130740|*

   > **i** Note
   >
   > When you generate the report key, the first field is automatically generated with the *Report Acronym*, and the last two fields are filled in with the *Run Type* and its date and time (if the Run Type it is filled in as Draft or as Rectification).
3. Redefine the `VALIDATE_CREATE_OFFICIAL_RUN` method, if necessary. This method checks if there is a valid report key within the database that is equal to the `GET_REPORT_KEY` method.
4. Enter the report key class that has been created within the *Report Key Generation Class* field, in the `/TMF/D_REP_CUST` table (`SM30` transaction).

How to Consume

We recommend you to use the Report Key with the class `/TMF/CL_TOM_METADATA`.

Example

Selection by Report Key:

```
DATA(lt_report_runs) = /tmf/cl_tom_metadata=>get_report_runs_by_range(
                                                ir_report_key    = VALUE #( ( sign = 'I' option = 'EQ' low = mv_report_key ) )
                                                ir_report_status = VALUE #( ( sign = 'E' option = 'EQ' low = /tmf/cl_constants=>mo_tom->report_status-error )
                                                                            ( sign = 'E' option = 'EQ' low = /tmf/cl_constants=>mo_tom->report_status-canceled ) ) ).
```

Selection by Run Type:

```
DATA(lo_report_key) = /tmf/cl_report_key_factory=>get_instance( mo_report_run ).

"Get Original and Rectified reports with the same key
rt_offic_rectif_runs = /tmf/cl_tom_metadata=>get_report_runs_by_range(
                                          ir_report_key    = VALUE #( ( sign = mc_include option = mc_covers_pattern low = lo_report_key->get_repkey_by_run_type_for_sel( ) ) )
                                          ir_run_type      = VALUE #( ( sign = mc_include option = mc_not_equal      low = /tmf/cl_constants=>mo_tom->run_usage-draft ) )
                                          ir_report_status = VALUE #( ( sign = mc_exclude option = mc_equal          low = /tmf/cl_constants=>mo_tom->report_status-error )
                                                                      ( sign = mc_exclude option = mc_equal          low = /tmf/cl_constants=>mo_tom->report_status-canceled ) ) ).
```

4 Storage Provider

Storage provider is a TDF solution that allows you to store SPED files.

This solution is based on the `Template Method Design Pattern`, in which algorithm steps can be overwritten by subclasses. The base class of the template method define which are the hook methods that you need to overwrite by subclasses, to differ the system behaviors while ensuring that the algorithm is still followed.

The TDF default storage providers are `File System Storage Provider` and `Database Storage Provider`, which instances are created through the `/TMF/CL_STORAGE_FACTORY` class.

You can custom a storage provider and use it to store the SPED files by implementing the BAdI: Storage Provider (`/TMF/BADI_STORAGE_PROV_FACTORY`).

GET_PROVIDER ( ) - Gets a report run entity and returns an instance of the storage provider.

Parameter	Type	Associated Type	Description	Required
`IO_REPORT_RUN`	Importing	`/TMF/CL_ROM_REPORT_RUN`	TOM API: Report Run	X
`RO_SOTRAGE_PROVIDER`	Returning	`/TMF/CL_STORAGE_BASE`	Base Class For Storage Provider	

Example

```
DATA(lo_storage_provider) = /tmf/
cl_storage_factory=>get_provider( lo_report_run ).
```

The `/TMF/CL_STORAGE_BASE` class is the base for all storage providers. If you custom your own storage provider, it must inhere from this class.

Consumables Methods

To execute the applications, you call some methods already defined in the base class, which are the following:

DELETE_FILE( ): Get a report file and a report run entities and delete a file from TOM. This action is logged on TOM. This method calls the DELETE_FILE_HOOK( ) extendible method.

Parameter	Type	Associated Type	Description	Required
`IO_REPORT_FILE`	Importing	`/TMF/CL_TOM_REPORT_FILE`	TOM API: Report files	X

Parameter	Type	Associated Type	Description	Required
IO_REPORT_RUN	Importing	/TMF/CL_TOM_REPORT_RUN	TOM API: Report Run	

Sample Code

```
lo_storage_provider->delete_file( io_report_file = lo_report_file
                                  io_report_run  = lo_report_run ).
```

GET_FILE( ): Get a file ID and a repot file entity. Return the file as XSTRING format. The action is validated by the /TMF/CL_TOM_VALIDATOR class . This method calls the GET_FILE_HOOK( ) extendible method.

Parameter	Type	Associated Type	Description	Required
IV_FILE_ID	Importing	/TMF/DE_FILE_ID	File ID	X
IO_REPORT_FILE	Importing	/TMF/CL_TOM_REPORT_FILE	TOM API: Report Files	
RV_XSTRING	Returning	XSTRING	File Content	

Sample Code

```
DATA(lv_file) = lo_storage_provider->get_file( iv_file_id = lv_file_id
io_report_file = lo_report_file ).
```

SAVE_FILE_FROM_SPED( ): Get a table with a SPED output table, the report file and thereport run entities and convert the table into a SPED file. Then, call the SAVE_FILE( ) method to save the file into TOM API.

Parameter	Type	Associated Type	Description	Required
IT_SPED_FILE	Importing	/TMF/T_SPED_OUTPUT	Table to output data on SPED	X
IO_REPORT_FILE	Importing	/TMF/CL_TOM_REPORT_FILE	TOM API: Report Files	X
IO_REPORT_RUN	Importing	/TMF/CL_TOM_REPORT_RUN	TOM API: Report Run	

Sample Code

```
lo_storage_provider->save_file_from_sped( io_report_file = mo_report_file
                                          it_sped_file   = mt_sped_file
```

```
                                          io_report_run  = mo_report_run ).
```

SAVE_FILE( ): Get the file content in the XSTRING format, the report file and report run entities to create the file in TOM. The action is logged on TOM and it is validated by the /TMF/CL_TOM_VALIDATOR class . This method calls the SAVE_FILE_HOOK( )extendible method to actually save the file in the specific storage provider.

Parameter	Type	Associated Type	Description	Required
IO_REPORT_FILE	Importing	/TMF/CL_TOM_REPORT_FILE	TOM API: Report Files	X
IV_FILE_CONTENT	Importing	XSTRING	File content as XString	X
IO_REPORT_RUN	Importing	/TMF/CL_TOM_REPORT_RUN	TOM API: Report Run	X

Sample Code

```
save_file( io_report_file  = lo_report_file
           iv_file_content = lv_binary_file
           io_report_run   = lo_report_run ).
```

GET_STORAGE_PROVIDER_NAME( ): Returns the storage provider's name. This method calls the GET_STORAGE_PROVIDER_NAME_HOOK( ) extendible method to get the name of the storage provider.

Parameter	Type	Associated Type	Description	Required
RV_PROVIDER_NAME	Returning	STRING	Storage Provider name	

Example

```
DATA(lv_storage_provider_name) = lo_storage_provider-
>get_storage_provider_name( ).
```

Hook Methods

You use these methods to allow a specific behavior within your specific classes and extensions, which are the following:

DELETE_FILE_HOOK( ): Get the report file entity and delete the file from a specific storage provider. This method is called by the DELETE_FILE( ) method.

Parameter	Type	Associated Type	Description	Required
IO_REPORT_FILE	Importing	/TMF/ CL_TOM_REPORT_F ILE	TOM API: Report Files	X

SAVE_FILE_HOOK( ):Get the report file entity and the file content in the XSTRING format and persists in the specific storage provider. This method is called by the SAVE_FILE( ) method.

Parameter	Type	Associated Type	Description	Required
IO_REPORT_FILE	Importing	/TMF/ CL_TOM_REPORT_F ILE	TOM API: Report Files	X
IV_FILE_CONTENT	Importing	XSTRING	File content as XString	X

GET_FILE_HOOK( ): Get the report file entity and return the file from a specific storage provider in XSTRING format. This method is called by the GET_FILE( ) method.

Parameter	Type	Asociated Type	Description	Required
IO_REPORT_FILE	Importing	/TMF/ CL_TOM_REPORT_F ILE	TOM API: Report file	X
RV_XSTRING	Returning	XSTRING	File content	

SAVE_SESSION_APPEND_HOOK( ): Get the session instance created in the GET_FILE_SESSION_START( ) method and the file chunk content in XSTRING format and persist it in the specific storage provider. This method is called by the SAVE_SESSION_APPEND( ) method.

Parameter	Type	Type	Description	Required
IO_SESSION	Importing	/TMF/ CL_STORAGE_SESS ION_SAVE	File streaming session	X
IV_FILE_CHUNK	Importing	XSTRING	File chunk	X

GET_FILE_SESSION_DOWNLOAD_HOOK( ): Get the session instance created in the GET_FILE_SESSION_START( ) method and the number of the chunk to be downloaded and return the file chunk from the specific storage provider in XSTRING format. This method is called by the GET_FILE_SESSION_DOWNLOAD( ) method.

Parameter	Type	Associated Type	Description	Required
IO_SESSION	Importing	/TMF/CL_STORAGE_SESSION_READ	File streaming read session	X
IV_CHUNK_NUMBER	Importing	INT4	Number of the chunk to be downloaded	X
RV_FILE_CHUNK	Returning	XSTRING	File chunk	

CALCULATE_NR_OF_CHUNKS_HOOK( ): Get the report file and report run entities, and with the size of the chunk, calculate and return the number of chunks of a file from a specific storage provider. This result is set in the session created by GET_FILE_SESSION_START ( ) method.

Parameter	Type	Associated Type	Description	Required
IO_REPORT_RUN	Importing	/TMF/CL_TOM_REPORT_RUN	TOM API: Report Run	X
IO_REPORT_FILE	Importing		TOM API: Report Files	X
IV_CHUNK_SIZE	Importing	DF_BYTES	Chunk size	X
RV_NUMBER_OF_CHUNKS	Returning	INT4	Number of chunks of a file	

GET_STORAGE_PROVIDER_NAME_HOOK( ): Returns the name of a specific storage provider.

Parameter	Type	Associated Type	Description	Required
RV_PROVIDER_NAME	Returning	STRING	Storage Provider name	

4.1 Storage Provider for Big Files

Storage provider for big files is a TDF solution that allows you to store big SPED files, by saving a file in chunks.

To save big files in TOM, the file in chunks must be as XSTRING type. To save a file in chunks, proceed as follows:

1. Create a *Save Session* using the SAVE_SESSION_START( ) method from a specific storage provider.
2. Append the file chunks using the SAVE_SESSION_APPEND( ) method, to keep the file chunk within a specific storage provider.
3. Finish the *Save Session* using the SAVE_SESSION_FINISH( ) from a specific storage provider.

Consumable Methods

SAVE_SESSION_START( ): Get the report file and report run entities and create an instance of the save session. Then, call the SAVE_SESSION_APPEND( ) and the SAVE_SESSION_FINISH( ) methods. The action is logged in TOM.

Parameter	Type	Associated Type	Description	Required
IO_REPORT_FILE	Importing	/TMF/CL_TOM_REPORT_FILE	TOM API: Report Files	X
IO_REPORT_RUN	Importing	/TMF/CL_TOM_REPORT_RUN	TOM API: Report Run	X
RO_SESSION	Returning	/TMF/CL_STORAGE_SESSION_SAVE	File ID	

SAVE_SESSION_APPEND( ): Get the session instance created in the SAVE_SESSION_START( ) method and a chunk of the file and call the SAVE_SESSION_APPEND_HOOK( ) method to save the file chunk in the specific storage provider. This method must be called after the SAVE_SESSION_START( ) method and before the SAVE_SESSION_FINISH( ) method. The action is logged in TOM.

Parameter	Type	Associated Type	Description	Required
IO_SESSION	Importing	/TMF/CL_STORAGE_SESSION_SAVE	File ID	X
IV_FILE_CHUNK	Importing	XSTRING	File Content	X

SAVE_SESSION_FINISH( ): Get the session instance created in the SAVE_SESSION_START( ) method and finish the process of saving the file in chunks. This method must be called after the SAVE_SESSION_START( ) and the SAVE_SESSION_APPEND( ) methods. The action is logged in TOM.

Parameter	Type	Associated Type	Description	Required
IO_SESSION	Importing	/TMF/CL_STORAGE_SESSION_SAVE	File ID	X

Sample Code

Example of Consumption

```
"Create save session
DATA(lo_session) = lo_storage_provider->save_session_start( io_report_file =
lo_report_file
                                                            io_report_run  =
lo_report_run ).
LOOP AT lt_content_binary INTO DATA(lv_binary_chunk).
  "Append file chunk
  lo_storage_provider->save_session_append( io_session      = lo_session
```

```
                                                iv_file_chunk    =
lv_binary_chunk ).
ENDLOOP.
"Finish save session
lo_storage_provider->save_session_finish( io_session = lo_session ).
```

Download File in Chunks

You can download big files in chunks for the standard storage providers or for storage providers that you have defined, proceeding as follows:

1. Create a *Download Session* using the `GET_FILE_SESSION_START( )` method from a specific storage provider.
2. Download the file chunks using the `GET_FILE_SESSION_DOWNLOAD( )` method from a specific storage provider.

GET_FILE_SESSION_START( ): Get the report file and report run entities and create an instance for the download session. Call the GET_FILE_SESSION_DOWNLOAD( ) method after that. The action is validated by the /TMF/CL_TOM_VALIDATOR class.

Parameter	Type	Associated Type	Description	Required
IO_REPORT_FILE	Importing	/TMF/CL_TOM_REPORT_FILE	TOM API: Report Files	X
IO_REPORT_RUN	Importing	/TMF/CL_TOM_REPORT_RUN	TOM API: Report Run	X
RO_SESSION	Returning	/TMF/CL_STORAGE_SESSION_READ	File streaming session	

GET_FILE_SESSION_DOWNLOAD( ): Get the session instance created in the GET_FILE_SESSION_START( ) method and the number of the chunk to be downloaded and then call the GET_FILE_SESSION_DOWNLOAD_HOOK( ) extendible method to download the chunk from a specific storage provider.

Parameter	Type	Associated Type	Description	Required
IO_SESSION	Importing	/TMF/CL_STORAGE_SESSION_READ	File streaming read session	X
IV_CHUNK_NUMBER	Importing	INT4	Number of the chunk to be downloaded	X
RV_FILE_CHUNK	Returning	XSTRING	File chunk	

Sample Code

Example of downloading a file in chunks

```
DATA lv_binary TYPE xstring.
"Create download session
DATA(lo_session) = lo_storage_provider-
>get_file_session_start( io_report_file = lo_report_file

io_report_run  = lo_report_run ).
DO lo_session->get_number_of_chunks( ) TIMES.
    "Download file chunk
    DATA(lv_xstring) = lo_storage_provider-
>get_file_session_download( io_session      = lo_session

iv_chunk_number = sy-index ) .
    lv_binary = lv_binary && lv_xstring.
ENDDO.
```

5 CTR Data Structures

In the Central Tax Repository (CTR), two types of data structure are used in data building for fiscal documents: tables and calculation views. These are divided into subtypes, as described in the following sections.

Tables

Tables are structures that store information from SAP ERP or another source. For the CTR, some types of tables have been created in order to organize and configure data. These tables are shown in the entity relationship (ER) model as a blue square or rectangle with no lines inside (see section *Modeling Structure*).

SAP ERP Standard Tables

These tables contain exactly the same data as the SAP ERP system, for example, `T001K`.

> **i** Note
>
> Developers must not consume SAP ERP standard tables in programs, that is, front-end applications.

CPL Tables

Complement tables deliver information that does not exist in SAP ERP. The names of these tables have the suffix CPL, for example, `INVENTARIO_CPL`.

The following table lists all of the complement tables and the private views that consume them.

Complement Table	Consumed By
/TMF/D_EMP_CPL	PRV_EMPRESA
/TMF/D_ICMSSTCPL	PRV_APURACAO_ICMS_ST_UF_ANTERIOR
/TMF/D_ICMS_CPL	PRV_CMS_ANTERIOR
/TMF/D_ICTER_CPL	PRV_INSUMOS_CONSUMIDOS_TERCEIROS
/TMF/D_IMPH_CPL	PRV_IMPRESSAO_CABECALHO
/TMF/D_IMPI_CPL	PRV_IMPRESSAO_ITEM
/TMF/D_INVEN_CPL	PRV_INVENTARIO
/TMF/D_ITEM_CPL	PRV_ITEM
/TMF/D_LCONT_CPL	PRV_LANCAMENTO_CONTABIL
/TMF/D_NFDOC_CPL	PRV_NF_DOCUMENTO

Complement Table	Consumed By
/TMF/D_NF_IT_CPL	PRV_NF_ITEM
/TMF/D_OP_MD_CPL	PRV_OPER_MEDICAMENTO
/TMF/D_PLCTA_CPL	PRV_PLANO_CONTAS
/TMF/D_PROD_CPL	PRV_PRODUCAO
/TMF/D_SCONT_CPL	PRV_SALDO_CONTABIL

Shadow Tables

Shadow tables have the same structure as the reuse views. They are used when data from SAP ERP requires changes or corrections that, for any reason, cannot be done in SAP ERP itself, but are legal obligations in Brazil.

In the views that use shadow tables, if any register is retrieved from SAP ERP with the same key in the shadow table, the information is replaced with data from the shadow table. That is, shadow tables have priority over data formed from SAP ERP tables.

Also, it is possible to insert data that is not available in the SAP ERP directly into the shadow tables.

Shadow tables are represented with a name similar to the scenarios required for the fiscal legislation in Brazil.

Calculation Views

Calculation views are entities that get information from tables in SAP ERP, the CTR or legacy systems as input, execute several required calculations, and give the result in a proper consumable format .

The CTR splits views into different types in order to differentiate between views that belong to standard SAP ERP and views that can be manipulated by users to get any kind of information that does not exist in the standard system.

These views are shown in the ER model as a purple square or rectangle with two lines inside (see section *Modeling Structure*).

Private Views

Private views are used to isolate the SAP ERP content from the CTR. These views are the foundation for CTR reuse views and are the only entity that access SAP ERP data. Private views cannot consume a reuse view.

The name of these views have the prefix PRV, for example, `PRV_LANCAMENTO_CONTABIL`.

> **i Note**
>
> Developers must not consume private views in programs, that is, front-end applications. It is not recommended the consume of views.

Reuse Views

Reuse views aggregate, in a single place, data from SAP ERP, legacy systems, CPL tables, shadow tables PRV views or other reuse views. They never access any standard tables; they contain all the necessary data to create reports.

The names of these tables usually follow the name of the process to which they refer, for example, `NF_ITEM`.

All reuse views are available for consumption directly in an ABAP program. This is possible because the HANA View is synchronized with NetWeaver. For more information on how to use refer to the ABAP program /TMF/CTR_DEMO.

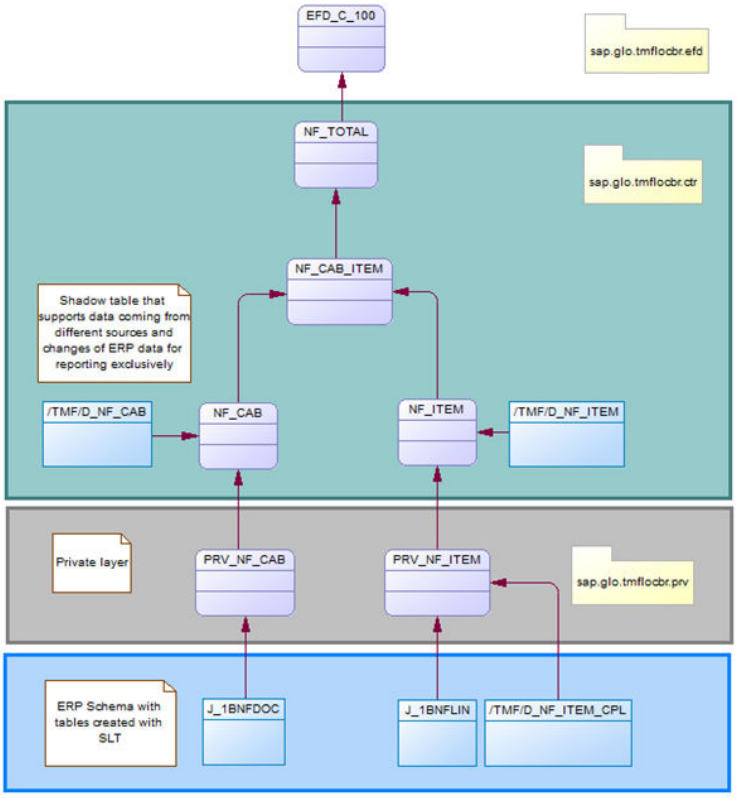
TDF Analytics Views

Analytics views are optimized in a consumable format, to facilitate the consume in analytics tools, such as Lumira and BOC. The analytics views are the following:

- `sap.glo.tmflocbr.ana.fiscal/SALDO_ICMS`
- `sap.glo.tmflocbr.ana.fiscal/VALORES_IMPOSTOS_REGIAO`
- `sap.glo.tmflocbr.ana.ciap/FICHAS`
- `sap.glo.tmflocbr.ana.ciap/PARCELAS`

Modeling Structure

The figure below illustrates how data is modeled in the CTR:



The bottom (blue) box shows SAP ERP tables. These tables feed the private views shown in the next (gray) box. Complement tables are also shown in the gray box, if they exist. The next (green) box shows the reuse views, which will be consumed by other views, for example, to fill SPED EFD.

Entity Relationship Model

Definition

The entity relationship model, also known as the ER model or ER diagram, is a diagrammatic model that describes the system data model with a high level of abstraction. It is a static view of the system. Its main application is to visualize the relationship between database tables, which are built through the association of table attributes.

6 CTR Demo

The program `/TMF/CTR_DEMO` demonstrates how to consume and insert data into the CTR. The program shows how to apply, using ABAP code, the main concepts presented in this document.

Access the *Verify Shadow Tables Use* report (`/TMF/CTR_TECH_INFO`) to check which tables are used in each CTR.

7 CTR Extensions

It is possible that information is identified as required in the CTR but not yet supported there. In this case, the CTR must be enhanced to support this data.

Depending on the source and category of the data, there are different strategies to support CTR enhancements.

Extension Scenarios

- **Field exists in SAP ERP but not in the CTR**
  The requirement must be communicated to SAP, which will analyze it and possibly add it to the CTR upper views and reports depending on them.
- **Field does not exist in SAP ERP but is required in the CTR**
  CPL tables (see section *CPL Tables* in CTR Data Structures [page 51]) have been created to support this situation. Check whether the field is not already present in the respective table. If it is not, the requirement must be communicated to SAP, which will analyze it and possibly add it to the CPL table and consequently to all necessary CTR views. Once the field is available, the data must be inserted accordingly so it is consumed by the CTR.
- **Information is not available as part of CTR scope**
  New views must be created. The CTR views should be added to the new views as a separate projection. Copying the complete CTR view instead of reusing it causes the view to not receive updates and corrections, and, therefore, this is not recommended. It is recommended to first apply for the previously mentioned options because, in that case, the data is communicated through several CTR views, facilitating the data consumption.
  For example, a new table is created with an additional field `NEW_FIELD`. To join this field with the existing nota fiscal CTR view, a new view is created. This new view projects the new table and joins a CTR view with data already aggregated, which enables the new view to be ready for output.
- **SPED-related tables are not properly populated**
  To ensure that the `/TMF/D_REP_FISC`, `/TMF/D_VIEW_VERS`, `/TMF/D_REP_VERS`, `/TMF/D_REG_PROFI`, and `/TMF/D_REG_PRIO` tables are populated correctly, with all necessary entries from the EFD and ECD reports, you check SAP Note 2045948 for the required information.

8 Customer Extension

> → Recommendation
>
> In general for TDF, SAP recommends to fix data inconsistencies at origin, which means correcting data in the ERP system connected to the TDF system. This is the best approach as it avoids introducing new errors in subsequent reporting periods. In cases where it is not possible to correct data in ERP, filling shadow tables is the second option. In **exceptional** cases, where it is not possible to correct data either at origin or in the shadow tables, proceed as described below.

In TDF, there are two ways to extend the standard content and logic, in addition to the usage of shadow tables:

- Create a new HANA view to generate the result of a given register.
- Create a specific class to handle or retrieve the data from a register.

Important

- Customer extensions are considered to be non-standard development. Views created by customers or logic implemented in the ABAP are not supported by SAP.
- During incident processing, customers might be asked to activate the TDF trace report in order to identify customer extensions in TDF reports. Customers might also be asked to deactivate customer extensions. For details of the trace report, see *View Report Execution Log* in the application help for TDF.
- It is the customer's responsibility to ensure the compatibility of customer extensions after installing support packages or implementing SAP notes.

Create Views with Your Own Logic

If you have a specific scenario not covered by the standard logic available in TDF views or that cannot be covered through populating shadow tables, you can create a specific view for your requirements.

To create a view, proceed as follows:

1. Create a package in the same level as package "sap", with the customer's or partner's name.
2. In this package, you can organize the objects and sub-packages freely according to your requirements.
3. In that structure, implement your HANA view, with the logic you need. It is important to pay attention to the following items:
    - Use input parameters only if they are available in the parameter class of the SPED report you are extending. For example, for SPED EFD-Contribuições, you can only use parameters available in class /TMF/CL_SPED_PARAMETERS_PCO.
    - The name of these parameters must be the same used in the GET_* and SET_* methods. For example, for a parameter to filter company code, use P_EMPRESA as its name, since the method GET_EMPRESA is available in class /TMF/CL_SPED_PARAMETERS_PCO and this method is responsible for returning the company code entered in the selection screen of the report.
    - Two fields are mandatory in the view's output:
        - REG (register name)
          This field marks the beginning of the register's content

- SORTER (unique code that is used to sort the report's final result)
  This field marks the end of the register's content and must be the last one in the output list.
- You cannot have measures as part of the view's output, only attributes.
- If you need any additional fields, for example, to act as filters in a child register, you can insert them before the field REG, so it that they are not in the report line. The fields MANDT, EMPRESA and FILIAL must be available in the output this way whenever is possible

4. Create this view's mapping to NetWeaver, generating an ABAP view.
5. Map the ABAP to the register. You do this in Customizing for *Accounting* under *Tax Declaration Framework for Brazil* > *Reporting* > *Map View Version to Legal Report Version*.

Modify the ABAP Code Responsible for Handling a HANA View

There may be cases when it is necessary to create specific logic to handle register data in ABAP. For example, this could be to create a specific query for a register, to create post-processing logic to treat the data retrieved by a view, or to create a specific format for a given field

> ⚠ Caution
>
> Modifying data outside HANA using ABAP is not considered good practice as other tools can consume data from HANA and give different results from SPED reports. Use this alternative **only** when really necessary and in cases where fixing data at origin, shadow tables and customer-specific views is not possible.

To modify the ABAP code:

1. Create a specific class for your needs. This class must have class /TMF/CL_SPED_REG or classes /TMF/CL_SPED_REG_<report> as the super class. For example, to create a class for register 0200 of SPED EFD-Contribuições, you can create the class ZCL_SPED_REG_PCO_0200 as a child of /TMF/CL_SPED_REG_PCO.
   Note: You should **not** redefine the method GET_FORMATTED_DATA in class /TMF/CL_SPED_REG.
2. Implement your specific logic in the created class.
3. Create an implementation for BAdI /TMF/BADI_SPED_REG_FACTORY, available in enhancement spot /TMF/ES_SPED_REG_FACTORY.
   The BAdI is available in Customizing for *Accounting* under *Tax Declaration Framework for Brazil* > *Reporting* > *BAdI: Generate Register Instance*.
4. Implement method /TMF/IF_SPED_REG_FACTORY~REGISTER_FACTORY in this BAdI implementation. In this method you have the following available as input data:
   - Details of the register to be executed as provided by HANA view sap.glo.tmflocbr.ctr/REGISTROS_POR_VERSAO_REPORT (IS_REGISTER)
   - Report selection screen parameters and values (IO_PARAMETERS)

   Use this data to generate the register instance to be returned in parameter RO_REGISTER.
   In the example above, when the field is_register-report_id has the value "3" (SPED EFD-Contribuições) and the field is_register-view_register has the value "0200", you create the instance of class ZCL_SPED_REG_PCO_0200.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.
About the icons:

- Links with the icon : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
  - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
  - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon : You are leaving the documentation for that particular SAP product or service and are entering a SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.
The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Gender-Related Language

We try not to use gender-specific word forms and formulations. As appropriate for context and readability, SAP may use masculine word forms to refer to all genders.

www.sap.com/contactsap

THE BEST RUN SAP