



Informatica® Data Validation  
July 2025

# REST API Reference

Informatica Data Validation REST API Reference  
July 2025

© Copyright Informatica LLC 2023, 2025

Publication Date: 2025-07-10

# Table of Contents

<b>Preface .....</b>	<b>5</b>
Informatica Resources. ....	5
Informatica Documentation. ....	5
Informatica Intelligent Cloud Services web site. ....	5
Informatica Intelligent Cloud Services Communities. ....	5
Informatica Intelligent Cloud Services Marketplace. ....	5
Data Integration connector documentation. ....	6
Informatica Knowledge Base. ....	6
Informatica Intelligent Cloud Services Trust Center. ....	6
Informatica Global Customer Support. ....	6
 <b>Chapter 1: Introduction to Data Validation REST APIs .....</b>	 <b>7</b>
Login. ....	7
Time zone codes. ....	8
 <b>Chapter 2: Test case REST APIs.....</b>	 <b>12</b>
Creating a test case. ....	12
Updating a test case. ....	22
Running a test case. ....	32
Viewing a test case. ....	33
Viewing run sequences for a test case. ....	36
Viewing history of a test case run. ....	38
 <b>Chapter 3: Test case report REST APIs.....</b>	 <b>41</b>
Viewing a summary report for a test case run. ....	41
Viewing a detailed report for a test case run. ....	44
Downloading the report for a test case run. ....	45
 <b>Chapter 4: Test suite REST APIs.....</b>	 <b>48</b>
Creating a test suite. ....	48
Updating a test suite. ....	50
Running test cases in a test suite. ....	51
Viewing a test suite. ....	52
 <b>Chapter 5: Saved SQL query REST APIs.....</b>	 <b>54</b>
Creating a saved SQL query. ....	54
Updating a saved SQL query. ....	56
Viewing a saved SQL query. ....	59
Viewing the supported database types for a saved SQL query. ....	61

**Index..... 63**

# Preface

Read the *REST API Reference* to learn how you can use Data Validation REST APIs to work with test cases, reports, test suites, and saved SQL queries.

## Informatica Resources

Informatica provides you with a range of product resources through the Informatica Network and other online portals. Use the resources to get the most from your Informatica products and solutions and to learn from other Informatica users and subject matter experts.

### Informatica Documentation

Use the Informatica Documentation Portal to explore an extensive library of documentation for current and recent product releases. To explore the Documentation Portal, visit <https://docs.informatica.com>.

If you have questions, comments, or ideas about the product documentation, contact the Informatica Documentation team at [infa\\_documentation@informatica.com](mailto:infa_documentation@informatica.com).

### Informatica Intelligent Cloud Services web site

You can access the Informatica Intelligent Cloud Services web site at <http://www.informatica.com/cloud>. This site contains information about Informatica Cloud integration services.

### Informatica Intelligent Cloud Services Communities

Use the Informatica Intelligent Cloud Services Community to discuss and resolve technical issues. You can also find technical tips, documentation updates, and answers to frequently asked questions.

Access the Informatica Intelligent Cloud Services Community at:

<https://network.informatica.com/community/informatica-network/products/cloud-integration>

Developers can learn more and share tips at the Cloud Developer community:

<https://network.informatica.com/community/informatica-network/products/cloud-integration/cloud-developers>

### Informatica Intelligent Cloud Services Marketplace

Visit the Informatica Marketplace to try and buy Data Integration Connectors, templates, and mapplets:

<https://marketplace.informatica.com/>

## Data Integration connector documentation

You can access documentation for Data Integration Connectors at the Documentation Portal. To explore the Documentation Portal, visit <https://docs.informatica.com>.

## Informatica Knowledge Base

Use the Informatica Knowledge Base to find product resources such as how-to articles, best practices, video tutorials, and answers to frequently asked questions.

To search the Knowledge Base, visit <https://search.informatica.com>. If you have questions, comments, or ideas about the Knowledge Base, contact the Informatica Knowledge Base team at [KB\\_Feedback@informatica.com](mailto:KB_Feedback@informatica.com).

## Informatica Intelligent Cloud Services Trust Center

The Informatica Intelligent Cloud Services Trust Center provides information about Informatica security policies and real-time system availability.

You can access the trust center at <https://www.informatica.com/trust-center.html>.

Subscribe to the Informatica Intelligent Cloud Services Trust Center to receive upgrade, maintenance, and incident notifications. The [Informatica Intelligent Cloud Services Status](#) page displays the production status of all the Informatica cloud products. All maintenance updates are posted to this page, and during an outage, it will have the most current information. To ensure you are notified of updates and outages, you can subscribe to receive updates for a single component or all Informatica Intelligent Cloud Services components. Subscribing to all components is the best way to be certain you never miss an update.

To subscribe, on the [Informatica Intelligent Cloud Services Status](#) page, click **SUBSCRIBE TO UPDATES**. You can choose to receive notifications sent as emails, SMS text messages, webhooks, RSS feeds, or any combination of the four.

## Informatica Global Customer Support

You can contact a Global Support Center through the Informatica Network or by telephone.

To find online support resources on the Informatica Network, click **Contact Support** in the Informatica Intelligent Cloud Services Help menu to go to the **Cloud Support** page. The **Cloud Support** page includes system status information and community discussions. Log in to Informatica Network and click **Need Help** to find additional resources and to contact Informatica Global Customer Support through email.

The telephone numbers for Informatica Global Customer Support are available from the Informatica web site at <https://www.informatica.com/services-and-training/support-services/contact-us.html>.

## CHAPTER 1

# Introduction to Data Validation REST APIs

Use the Data Validation REST APIs to work with test cases, reports, test suites, and saved SQL queries.

To use the REST APIs, you need a valid Informatica Intelligent Cloud Services login and an understanding of REST API guidelines.

To configure a request using a REST API, use the appropriate resource and method, along with the applicable objects. Data Validation returns the requested information, performs the requested task, or returns an error and related messages.

## Login

Use this resource to log into Informatica Intelligent Cloud Services when you use resources that require the IDS-SESSION-ID in the call header. The IDS-SESSION-ID is included in a successful login response.

### POST request

Use the following URL:

```
<login URL>/identity-service/api/v1/Login
```

The login URL includes the region where your organization is located and the Informatica Intelligent Cloud Services domain, informaticacloud.com. You can find your organization's login region by opening the Informatica Intelligent Cloud Services log in page. The regional login URL is located in the browser's address bar before you log in to Informatica Intelligent Cloud Services.

In the following example, <https://dm-us.informaticacloud.com>, is the region URL:

```
https://dm-us.informaticacloud.com/identity-service/home
```

The following table describes the fields to include in the request:

Field	Type	Required	Description
username	String	Yes	Informatica Intelligent Cloud Services user name. Maximum length is 255 characters.
password	String	Yes	Informatica Intelligent Cloud Services password. Maximum length is 255 characters.

## POST response

Returns the user object if the request is successful. Returns the error object if errors occur.

Use the session ID returned in the response for subsequent requests.

The user object includes the following attributes:

Field	Type	Description
sessionId	String	REST API session ID for the current session. Use in most REST API request headers.
sessionExpireTime	String	Time the session expires.
id	String	User ID.
name	String	Informatica Intelligent Cloud Services user name.
currentOrgId	String	Current organization ID.
currentOrgName	String	Name of the current organization.
parentOrgId	String	ID of the parent organization.
orgId	String	ID of the organization the user belongs to.
orgName	String	Name of the organization the user belongs to.
groups	String	User group.
effectiveRoles	String	Roles assigned to the user.
effectivePrivileges	String	Privileges assigned to the user.
status	String	Status of the user.
timeZoneId	String	Time zone of the user. Time zone honors Daylight Saving Time. For more information, see <a href="#">"Time zone codes" on page 8</a> .
authenticator	String	User authentication method.

## Time zone codes

The Informatica Intelligent Cloud Services REST API uses the following time zone codes:

- ACT
- AET
- Africa/Cairo
- Africa/Casablanca
- Africa/Johannesburg
- Africa/Nairobi
- America/Barbados
- America/Bogota



- America/Buenos\_Aires
- America/Caracas
- America/Chicago
- America/Costa\_Rica
- America/Dawson\_Creek
- America/Denver
- America/Dominica
- America/El\_Salvador
- America/Guadeloupe
- America/Halifax
- America/Havana
- America/Jamaica
- America/La\_Paz
- America/Los\_Angeles
- America/Mexico\_City
- America/Montreal
- America/New\_York
- America/Panama
- America/Phoenix
- America/Puerto\_Rico
- America/Tijuana
- America/Vancouver
- Asia/Baghdad
- Asia/Bahrain
- Asia/Dubai
- Asia/Hong\_Kong
- Asia/Jerusalem
- Asia/Karachi
- Asia/Katmandu
- Asia/Kuala\_Lumpur
- Asia/Kuwait
- Asia/Magadan
- Asia/Muscat
- Asia/Qatar
- Asia/Rangoon
- Asia/Riyadh
- Asia/Seoul
- Asia/Singapore
- AST

- Atlantic/Cape\_Verde
- Atlantic/South\_Georgia
- Australia/Lord\_Howe
- Australia/Perth
- Brazil/Acre
- Brazil/DeNoronha
- Brazil/East
- Brazil/West
- BST
- CNT
- CTT
- Europe/Amsterdam
- Europe/Athens
- Europe/Belgrade
- Europe/Berlin
- Europe/Brussels
- Europe/Bucharest
- Europe/Budapest
- Europe/Copenhagen
- Europe/Istanbul
- Europe/London
- Europe/Luxembourg
- Europe/Madrid
- Europe/Moscow
- Europe/Paris
- Europe/Prague
- Europe/Rome
- Europe/Stockholm
- Europe/Vienna
- Europe/Warsaw
- Europe/Zurich
- GMT
- HST
- Indian/Mauritius
- IST
- JST
- Pacific/Apia
- Pacific/Auckland
- Pacific/Chatham

- Pacific/Enderbury
- Pacific/Fiji
- Pacific/Gambier
- Pacific/Kiritimati
- Pacific/Norfolk
- Pacific/Tahiti
- UTC
- VST

## CHAPTER 2

# Test case REST APIs

You can use REST APIs to perform the following tasks:

- Create a test case.
- Update a test case.
- Run a test case.
- View a test case.
- View run sequences for a test case.
- View history of a test case run.

## Creating a test case

Use the following URI to create a test case:

`POST /datavalidation-service/api/v1/testcase`

Use the IDS-SESSION-ID header in the POST request.

### POST request

Use the following fields in the POST request:

Field	Required?	Description
testCaseName	Yes	Name of the test case.
description	No	Description of the test case.
leftObjectType	Yes	Type of the data source. Specify one of the following values: <ul style="list-style-type: none"><li>- <b>OBJECT</b>. Indicates that the data source is a table or view.</li><li>- <b>SAVED_SQL_QUERY</b>. Indicates that the data source is a saved SQL query.</li></ul>

Field	Required?	Description
leftConnId	Yes	<p>Connection ID of the first connection.</p> <p>When you open the connection in Administrator, the numeric value that you see in the URL is the connection ID.</p> <p>For example, if the URL is &lt;Informatica Intelligent Cloud Services URL&gt;/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000898/read, the connection ID is 014N8B0B000000000898.</p>
leftSavedSqlQuery	No	<p>ID of the saved SQL query of the first connection.</p> <p>Use the following format:</p> <pre>"leftSavedSqlQuery" : { "id" : &lt;value&gt; }</pre>
leftTableName	No	Table name of the first connection.
leftTablePk	Yes	<p>Primary key of the table in the first connection.</p> <p>You can also enter multiple keys.</p>
rightObjectType	Yes	<p>Type of the data source.</p> <p>Specify one of the following values:</p> <ul style="list-style-type: none"> <li>- <b>OBJECT</b>. Indicates that the data source is a table or view.</li> <li>- <b>SAVED_SQL_QUERY</b>. Indicates that the data source is a saved SQL query.</li> </ul>
rightConnId	Yes	<p>Connection ID of the second connection.</p> <p>When you open the connection in Administrator, the numeric value that you see in the URL is the connection ID.</p> <p>For example, if the URL is &lt;Informatica Intelligent Cloud Services URL&gt;/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000898/read, the connection ID is 014N8B0B000000000898.</p>
rightSavedSqlQuery	No	<p>ID of the saved SQL query of the second connection.</p> <p>Use the following format:</p> <pre>"rightSavedSqlQuery" : { "id" : &lt;value&gt; }</pre>
rightTableName	No	Table name of the second connection.
rightTablePk	Yes	<p>Primary key of the table in the second connection.</p> <p>You can also enter multiple keys.</p>
runtimeEnvId	Yes	Runtime environment where the test case runs.

Field	Required?	Description
stagingConnectionId	Yes	<p>Connection ID of the flat file connection that Data Validation uses to stores reports.</p> <p>When you open the flat file connection in Administrator, the numeric value that you see in the URL is the connection ID.</p> <p>For example, if the URL is &lt;Informatica Intelligent Cloud Services URL&gt;/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000ATW/read, the connection ID is 014N8B0B000000000ATW.</p>
frsDocLocation	Yes	<p>The location to save the test case.</p> <p>To save the test case in a project, use the following syntax:</p> <pre>{   "type": "Project",   "id": "&lt;project_ID&gt;",   "path": "&lt;project_name&gt;" }</pre> <p>To save the test case in a folder, use the following syntax:</p> <pre>{   "type": "Folder",   "id": "&lt;folder_ID&gt;",   "path": "&lt;folder_name&gt;" }</pre> <p>When you open a project, the numeric value that you see in the URL is the project ID. Similarly, when you open a folder, the numeric value that you see in the URL is the folder ID.</p> <p>By default, the test case is saved under the <b>Default</b> project.</p>
emailNotificationStrategy	Yes	<p>Specify one of the following values:</p> <ul style="list-style-type: none"> <li>- ORG. Data Validation uses the default email notification options that are configured in Administrator.</li> <li>- CUSTOM. Specify the email addresses that Data Validation must use for the success and failure email notifications. Separate multiple email addresses with a comma.</li> <li>- NONE. Data Validation doesn't send email notifications.</li> </ul>
successEmails	Required if emailNotificationStrategy is set to CUSTOM	Specify the email addresses that Data Validation must use for the success email notifications. Separate multiple email addresses with a comma.
errorEmails	Required if emailNotificationStrategy is set to CUSTOM	Specify the email addresses that Data Validation must use for the failure email notifications. Separate multiple email addresses with a comma.
leftConnPath	Required for Amazon Redshift v2 and Amazon S3 v2 connections	If you use an Amazon Redshift v2 or an Amazon S3 v2 connection, enter a valid path for the first connection.

Field	Required?	Description
leftConnAdvancedParams	No	<p>If you use an Amazon Redshift v2, Amazon S3 v2, or flat file connection, specify the advanced parameters to be used for the first connection.</p> <p><b>Amazon Redshift v2</b></p> <p>For Amazon Redshift v2 connections, specify the S3 bucket name in the advanced parameters. For example, enter:</p> <pre>{     "S3BucketName": "&lt;value&gt;" }</pre> <p><b>Amazon S3 v2</b></p> <p>For Amazon S3 v2 connections, Data Validation supports the Parquet file and flat file formats.</p> <p>For Parquet file format, specify the following advanced parameters:</p> <pre>{ "formatId": "Parquet", "@type": "dataFormat", "dataFormatAttributes": {} }</pre> <p>For flat file format, specify the following default advanced parameters:</p> <pre>{     "formatId": "Flat",     "@type": "dataFormat",     "dataFormatAttributes": {         "escapeChar": "&lt;value&gt;",         "delimiter": "&lt;value&gt;",         "headerLineNumber": &lt;value&gt;,         "qualifier": "&lt;value&gt;",         "firstDataRow": &lt;value&gt;     } }</pre> <p>For example:</p> <pre>{     "formatId": "Flat",     "@type": "dataFormat",     "dataFormatAttributes": {         "escapeChar": "",         "delimiter": ",",         "headerLineNumber": 1,         "qualifier": "\"",         "firstDataRow": 2     } }</pre> <p><b>Flat file</b></p> <p>For flat file connections, specify the advanced parameters to determine how the flat file must be parsed. For example, enter:</p> <pre>{ "srcFFAttrs" : { "delimiter" : "&lt;value&gt;", "textQualifier" : "&lt;value&gt;", "escapeChar" : "&lt;value&gt;", "headerLineNo" : &lt;value&gt;, "firstDataRow" : &lt;value&gt; } }</pre>

Field	Required?	Description
rightConnPath	Required for Amazon Redshift v2 and Amazon S3 v2 connections	If you use an Amazon Redshift v2 or an Amazon S3 v2 connection, enter a valid path for the second connection.



Field	Required?	Description
rightConnAdvancedParams	No	<p>If you use an Amazon Redshift v2, Amazon S3 v2, or flat file connection, specify the advanced parameters to be used for the second connection.</p> <p><b>Amazon Redshift v2</b></p> <p>For Amazon Redshift v2 connections, specify the S3 bucket name in the advanced parameters. For example, enter:</p> <pre>{     "S3BucketName": "&lt;value&gt;" }</pre> <p><b>Amazon S3 v2</b></p> <p>For Amazon S3 v2 connections, Data Validation supports the Parquet file and flat file formats.</p> <p>For Parquet file format, specify the following advanced parameters:</p> <pre>{     "formatId": "Parquet",     "@type": "dataFormat",     "dataFormatAttributes": {} }</pre> <p>For flat file format, specify the following default advanced parameters:</p> <pre>{     "formatId": "Flat",     "@type": "dataFormat",     "dataFormatAttributes": {         "escapeChar": "&lt;value&gt;",         "delimiter": "&lt;value&gt;",         "headerLineNumber": &lt;value&gt;,         "qualifier": "&lt;value&gt;",         "firstDataRow": &lt;value&gt;     } }</pre> <p>For example:</p> <pre>{     "formatId": "Flat",     "@type": "dataFormat",     "dataFormatAttributes": {         "escapeChar": "",         "delimiter": ",",         "headerLineNumber": 1,         "qualifier": "\"",         "firstDataRow": 2     } }</pre> <p><b>Flat file</b></p> <p>For flat file connections, specify the advanced parameters to determine how the flat file must be parsed. For example, enter:</p> <pre>{ "srcFFAttrs" : { "delimiter" :     "&lt;value&gt;" , "textQualifier" :</pre>

Field	Required?	Description
		<pre>"&lt;value&gt;" , "escapeChar" : "&lt;value&gt;" , "headerLineNo" : &lt;value&gt; , "firstDataRow" : &lt;value&gt; }}</pre>
leftConnWhereClause	No	<p>WHERE clause to be used for the first connection.</p> <p>If you use a WHERE clause, Data Validation selects those rows for sampling that meet the condition specified in the WHERE clause.</p>
rightConnWhereClause	No	<p>WHERE clause to be used for the second connection.</p> <p>If you use a WHERE clause, Data Validation selects those rows for sampling that meet the condition specified in the WHERE clause.</p>
verboseMode	No	<p>Defines the logging level.</p> <p>Enter <b>true</b> to use the verbose mode. The log includes messages of all logging levels. This option is useful for debugging.</p> <p>Enter <b>false</b> to use the standard mode. The log includes only error messages.</p>
keepDIAssets	No	<p>Defines whether you want to save the Data Integration mappings and tasks that Data Validation creates when it runs the test case.</p> <p>Enter <b>true</b> to save the Data Integration assets after the test case run.</p> <p>Enter <b>false</b> if you do not want to save the Data Integration assets after the test case run.</p>
ignoreCase	No	<p>Defines whether you want to ignore casing differences in the data.</p> <p>Enter <b>true</b> to ignore casing differences in the data.</p> <p>Enter <b>false</b> to consider the casing differences in the data as a mismatch.</p>
trimString	No	<p>Defines whether you want to trim leading and trailing white spaces in string values.</p> <p>Enter <b>true</b> to trim leading and trailing white spaces in string values.</p> <p>Enter <b>false</b> to retain leading and trailing white spaces in string values.</p> <p>Default is <b>false</b>.</p>
badRecordLimit	No	<p>Defines the maximum number of unmatched, extra, and missing records to show in the detailed test results. Enter the value as <b>100</b>, <b>500</b>, or <b>1000</b>.</p> <p>Default is <b>100</b>.</p>

Field	Required?	Description
colMappings	Yes	<p>Defines the name, type, precision, and scale for each column mapping. Use the following syntax to map columns in the first connection and second connection:</p> <pre> "colMappings": [   {     "aggrFunc": "&lt;function1,function2,functionn&gt;",     "leftColName": "&lt;value&gt;",     "leftColType": "&lt;value&gt;",     "rightColName": "&lt;value&gt;",     "rightColType": "&lt;value&gt;",     "rightColPrecision": &lt;value&gt;,     "rightColScale": &lt;value&gt;,     "leftColPrecision": &lt;value&gt;,     "leftColScale": &lt;value&gt;,   },   {     "leftColName": "&lt;value&gt;",     "leftColType": "&lt;value&gt;",     "rightColName": "&lt;value&gt;",     "rightColType": "&lt;value&gt;",     "rightColPrecision": &lt;value&gt;,     "rightColScale": &lt;value&gt;,     "leftColPrecision": &lt;value&gt;,     "leftColScale": &lt;value&gt;,     "aggrFunc": "min"   } ], </pre> <p>Use the <b>aggrFunc</b> field to define the aggregation functions that Data Validation must use to compare the table. Separate multiple aggregation functions with a comma. The aggregation function names must be in uppercase.</p> <p>An aggregation functions test retrieves summarized information about the data contained in the data sources. Use aggregation to verify whether all records were moved or to identify incorrect logic in WHERE clauses.</p> <p>Based on the data type of the columns, you can use one or more of the following aggregation functions:</p> <ul style="list-style-type: none"> <li>- <b>COUNT</b>. Counts the number of rows that contain non-null values for a string or numeric column.</li> <li>- <b>COUNT_ROWS</b>. Counts the number of rows for a string or numeric column. Includes rows that contain nulls.</li> <li>- <b>MIN</b>. Calculates the minimum value for a numeric column.</li> <li>- <b>MAX</b>. Calculates the maximum value for a numeric column.</li> <li>- <b>AVG</b>. Calculates the average value of a numeric column.</li> <li>- <b>SUM</b>. Calculates the total value of a numeric column.</li> </ul>
sampling	Yes	<p>Configure data sampling for Data Validation to compare between sample source data and sample target data. The less data that Data Validation samples, the faster the test runs.</p>

Field	Required?	Description
		<p>Use the following syntax to specify the sampling type and value:</p> <pre>"sampling": {   "type": "&lt;value&gt;",   "val": &lt;value&gt; }</pre> <p>Use one of the following options:</p> <ul style="list-style-type: none"> <li>- Enter the <b>type</b> as <b>percentage</b> to sample a percentage of the rows. The test report shows records that are missing or unmatched in the Connection 2 table, but doesn't show extra records if any exist. Enter the <b>val</b> as <b>1, 5, 10, 25, 50, or 75</b>. For example, enter:  <pre>"sampling": { "type": "percentage",   "val": 25 } }</pre> </li> <li>- Enter the <b>type</b> as <b>first</b> to sample a defined number of rows at the beginning of the table. Enter the <b>val</b> as <b>100, 1000, or 10000</b>. For example, enter:  <pre>"sampling": { "type": "first", "val": 100 } }</pre> </li> <li>- Enter the <b>type</b> as <b>last</b> to sample a defined number of rows at the end of the table. Enter the <b>val</b> as <b>100, 1000, or 10000</b>. For example, enter:  <pre>"sampling": { "type": "last", "val": 1000 } }</pre> </li> <li>- Enter the <b>type</b> as <b>where_clause</b> to sample those rows that meet the condition specified in the WHERE clause. If you use the <b>where_clause</b> option, you must specify a valid where clause in at least one of the following attributes: <ul style="list-style-type: none"> <li>- leftConnWhereClause</li> <li>- rightConnWhereClause</li> </ul> </li> </ul>

#### POST request sample

The following snippet shows a POST request sample to create a test case:

```
{
  "testCaseName": "PCToCDITestCase",
  "frsDocLocation": "{ \"type\": \"Project\", \"id\": \"436TU7N2RsicOmYQGVpWAl\", \"path
\": \"SG\" }",
  "description": "",
  "leftConnId": "014N8B0B000000000AGX",
  "leftConnType": "Oracle",
  "leftTableName": "EMPS_SNOW",
  "leftTablePk": ["EMPID"],
  "rightConnId": "014N8B0B000000000AGX",
  "rightConnType": "Oracle",
  "leftConnPath": "",
  "rightConnPath": "",
  "leftConnAdvancedParams": {},
  "rightConnAdvancedParams": {},
  "rightTableName": "EMPS_UPPERLOWERCASEFIELD",
  "rightTablePk": ["emp id"],
  "runtimeEnvId": "014N8B25000000000006",
  "stagingConnectionId": "014N8B0B00000000003HI",
  "colMappings": [{
    "aggrFunc": "",
    "leftColName": "NICKNAME",
    "leftColType": "string",
```

```

        "leftColPrecision": 100,
        "leftColScale": 0,
        "rightColName": "NICKNAME",
        "rightColType": "string",
        "rightColPrecision": 100,
        "rightColScale": 0
    }, {
        "aggrFunc": "",
        "leftColName": "CITY",
        "leftColType": "string",
        "leftColPrecision": 50,
        "leftColScale": 0,
        "rightColName": "CITY",
        "rightColType": "string",
        "rightColPrecision": 50,
        "rightColScale": 0
    }, {
        "aggrFunc": "",
        "leftColName": "GENDER",
        "leftColType": "string",
        "leftColPrecision": 20,
        "leftColScale": 0,
        "rightColName": "GENDER",
        "rightColType": "string",
        "rightColPrecision": 20,
        "rightColScale": 0
    }, {
        "aggrFunc": "",
        "leftColName": "EMPID",
        "leftColType": "decimal",
        "leftColPrecision": 38,
        "leftColScale": 0,
        "rightColName": "emp id",
        "rightColType": "decimal",
        "rightColPrecision": 38,
        "rightColScale": 0
    }, {
        "aggrFunc": "",
        "leftColName": "NAME",
        "leftColType": "string",
        "leftColPrecision": 200,
        "leftColScale": 0,
        "rightColName": "Full Name",
        "rightColType": "string",
        "rightColPrecision": 200,
        "rightColScale": 0
    }, {
        "aggrFunc": "",
        "leftColName": "SALARY",
        "leftColType": "decimal",
        "leftColPrecision": 9,
        "leftColScale": 1,
        "rightColName": "SALARY1",
        "rightColType": "decimal",
        "rightColPrecision": 9,
        "rightColScale": 1
    }, {
        "aggrFunc": "",
        "leftColName": "ORACLESTATUS",
        "leftColType": "string",
        "leftColPrecision": 70,
        "leftColScale": 0,
        "rightColName": "oracle status",
        "rightColType": "string",
        "rightColPrecision": 70,
        "rightColScale": 0
    }, {
        "aggrFunc": "",
        "leftColName": "ORACLECOMMENTS",
        "leftColType": "string",

```

```

        "leftColPrecision": 80,
        "leftColScale": 0,
        "rightColName": "Oracle Comments",
        "rightColType": "string",
        "rightColPrecision": 80,
        "rightColScale": 0
    }
},
"verboseMode": false,
"keepDIAssets": false,
"ignoreCase": false,
"trimString": false,
"leftConnWhereClause": "",
"rightConnWhereClause": "",
"badRecordLimit": 100,
"successEmails": "",
"errorEmails": "",
"emailNotificationStrategy": "NONE"
}

```

### POST response

If the test case was created successfully, the POST request returns a 200 `Successful operation` response and returns the following response fields:

Field	Type	Description
testCaseId	String	ID of the test case.
frsDocId	String	ID of the test case in the Informatica repository.
testCaseName	String	Name of the test case.

If the test case creation failed, the POST request returns a 400 `Bad request` response or a 500 `Internal Server Error` response. It also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Updating a test case

Use the following URI to update a test case:

```
PUT /datavalidation-service/api/v1/testcase/{testCaseId}
```

Use the `IDS-SESSION-ID` header in the PUT request.

## PUT request

Use the following fields in the PUT request:

Field	Required?	Description
testCaseName	Yes	Name of the test case.
description	No	Description of the test case.
leftObjectType	Yes	Type of the data source. Specify one of the following values: - <b>OBJECT</b> . Indicates that the data source is a table or view. - <b>SAVED_SQL_QUERY</b> . Indicates that the data source is a saved SQL query.
leftConnId	Yes	Connection ID of the first connection. When you open the connection in Administrator, the numeric value that you see in the URL is the connection ID. For example, if the URL is <Informatica Intelligent Cloud Services URL>/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000898/read, the connection ID is 014N8B0B000000000898.
leftSavedSqlQuery	No	ID of the saved SQL query of the first connection. Use the following format: "leftSavedSqlQuery" : { "id" : <value> }
leftTableName	No	Table name of the first connection.
leftTablePk	Yes	Primary key of the table in the first connection. You can also enter multiple keys.
rightObjectType	Yes	Type of the data source. Specify one of the following values: - <b>OBJECT</b> . Indicates that the data source is a table or view. - <b>SAVED_SQL_QUERY</b> . Indicates that the data source is a saved SQL query.
rightConnId	Yes	Connection ID of the second connection. When you open the connection in Administrator, the numeric value that you see in the URL is the connection ID. For example, if the URL is <Informatica Intelligent Cloud Services URL>/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000898/read, the connection ID is 014N8B0B000000000898.
rightSavedSqlQuery	No	ID of the saved SQL query of the second connection. Use the following format: "rightSavedSqlQuery" : { "id" : <value> }

Field	Required?	Description
rightTableName	No	Table name of the second connection.
rightTablePk	Yes	Primary key of the table in the second connection. You can also enter multiple keys.
runtimeEnvId	Yes	Runtime environment where the test case runs.
stagingConnectionId	Yes	<p>Connection ID of the flat file connection that Data Validation uses to stores reports.</p> <p>When you open the flat file connection in Administrator, the numeric value that you see in the URL is the connection ID.</p> <p>For example, if the URL is &lt;Informatica Intelligent Cloud Services URL&gt;/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000ATW/read, the connection ID is 014N8B0B000000000ATW.</p>
frsDocLocation	Yes	<p>The location to save the test case.</p> <p>To save the test case in a project, use the following syntax:</p> <pre>{\"type\": \"Project\", \"id\": \"&lt;project_ID&gt;\", \"path\": \"&lt;project_name&gt;\"}</pre> <p>To save the test case in a folder, use the following syntax:</p> <pre>{\"type\": \"Folder\", \"id\": \"&lt;folder_ID&gt;\", \"path\": \"&lt;folder_name&gt;\"}</pre> <p>When you open a project, the numeric value that you see in the URL is the project ID. Similarly, when you open a folder, the numeric value that you see in the URL is the folder ID.</p> <p>By default, the test case is saved under the <b>Default</b> project.</p>
emailNotificationStrategy	Yes	<p>Specify one of the following values:</p> <ul style="list-style-type: none"> <li>- ORG. Data Validation uses the default email notification options that are configured in Administrator.</li> <li>- CUSTOM. Specify the email addresses that Data Validation must use for the success and failure email notifications. Separate multiple email addresses with a comma.</li> <li>- NONE. Data Validation doesn't send email notifications.</li> </ul>
successEmails	Required if emailNotificationStrategy is set to CUSTOM	Specify the email addresses that Data Validation must use for the success email notifications. Separate multiple email addresses with a comma.
errorEmails	Required if emailNotificationStrategy is set to CUSTOM	Specify the email addresses that Data Validation must use for the failure email notifications. Separate multiple email addresses with a comma.



Field	Required?	Description
leftConnPath	Required for Amazon Redshift v2 and Amazon S3 v2 connections	If you use an Amazon Redshift v2 or an Amazon S3 v2 connection, enter a valid path for the first connection.

Field	Required?	Description
leftConnAdvancedParams	No	<p>If you use an Amazon Redshift v2, Amazon S3 v2, or flat file connection, specify the advanced parameters to be used for the first connection.</p> <p><b>Amazon Redshift v2</b></p> <p>For Amazon Redshift v2 connections, specify the S3 bucket name in the advanced parameters. For example, enter:</p> <pre>{   "S3BucketName": "&lt;value&gt;" }</pre> <p><b>Amazon S3 v2</b></p> <p>For Amazon S3 v2 connections, Data Validation supports the Parquet file and flat file formats.</p> <p>For Parquet file format, specify the following advanced parameters:</p> <pre>{ "formatId": "Parquet", "@type": "dataFormat", "dataFormatAttributes": {} }</pre> <p>For flat file format, specify the following default advanced parameters:</p> <pre>{   "formatId": "Flat",   "@type": "dataFormat",   "dataFormatAttributes": {     "escapeChar": "&lt;value&gt;",     "delimiter": "&lt;value&gt;",     "headerLineNumber": &lt;value&gt;,     "qualifier": "&lt;value&gt;",     "firstDataRow": &lt;value&gt;   } }</pre> <p>For example:</p> <pre>{   "formatId": "Flat",   "@type": "dataFormat",   "dataFormatAttributes": {     "escapeChar": "",     "delimiter": ",",     "headerLineNumber": 1,     "qualifier": "\"",     "firstDataRow": 2   } }</pre> <p><b>Flat file</b></p> <p>For flat file connections, specify the advanced parameters to determine how the flat file must be parsed. For example, enter:</p> <pre>{ "srcFFAttrs" : { "delimiter" : "&lt;value&gt;", "textQualifier" : "&lt;value&gt;", "escapeChar" : "&lt;value&gt;", "headerLineNo" : &lt;value&gt;, "firstDataRow" : &lt;value&gt; } }</pre>

Field	Required?	Description
rightConnPath	Required for Amazon Redshift v2 and Amazon S3 v2 connections	If you use an Amazon Redshift v2 or an Amazon S3 v2 connection, enter a valid path for the second connection.

Field	Required?	Description
rightConnAdvancedParams	No	<p>If you use an Amazon Redshift v2, Amazon S3 v2, or flat file connection, specify the advanced parameters to be used for the second connection.</p> <p><b>Amazon Redshift v2</b></p> <p>For Amazon Redshift v2 connections, specify the S3 bucket name in the advanced parameters. For example, enter:</p> <pre>{     "S3BucketName": "&lt;value&gt;" }</pre> <p><b>Amazon S3 v2</b></p> <p>For Amazon S3 v2 connections, Data Validation supports the Parquet file and flat file formats.</p> <p>For Parquet file format, specify the following advanced parameters:</p> <pre>{     "formatId": "Parquet",     "@type": "dataFormat",     "dataFormatAttributes": {} }</pre> <p>For flat file format, specify the following default advanced parameters:</p> <pre>{     "formatId": "Flat",     "@type": "dataFormat",     "dataFormatAttributes": {         "escapeChar": "&lt;value&gt;",         "delimiter": "&lt;value&gt;",         "headerLineNumber": &lt;value&gt;,         "qualifier": "&lt;value&gt;",         "firstDataRow": &lt;value&gt;     } }</pre> <p>For example:</p> <pre>{     "formatId": "Flat",     "@type": "dataFormat",     "dataFormatAttributes": {         "escapeChar": "",         "delimiter": ",",         "headerLineNumber": 1,         "qualifier": "\"",         "firstDataRow": 2     } }</pre> <p><b>Flat file</b></p> <p>For flat file connections, specify the advanced parameters to determine how the flat file must be parsed. For example, enter:</p> <pre>{ "srcFFAttrs" : { "delimiter" :     "&lt;value&gt;" , "textQualifier" :</pre>

Field	Required?	Description
		<pre>"&lt;value&gt;" , "escapeChar" : "&lt;value&gt;" , "headerLineNo" : &lt;value&gt; , "firstDataRow" : &lt;value&gt; }}</pre>
leftConnWhereClause	No	<p>WHERE clause to be used for the first connection.</p> <p>If you use a WHERE clause, Data Validation selects those rows for sampling that meet the condition specified in the WHERE clause.</p>
rightConnWhereClause	No	<p>WHERE clause to be used for the second connection.</p> <p>If you use a WHERE clause, Data Validation selects those rows for sampling that meet the condition specified in the WHERE clause.</p>
verboseMode	No	<p>Defines the logging level.</p> <p>Enter <b>true</b> to use the verbose mode. The log includes messages of all logging levels. This option is useful for debugging.</p> <p>Enter <b>false</b> to use the standard mode. The log includes only error messages.</p>
keepDIAssets	No	<p>Defines whether you want to save the Data Integration mappings and tasks that Data Validation creates when it runs the test case.</p> <p>Enter <b>true</b> to save the Data Integration assets after the test case run.</p> <p>Enter <b>false</b> if you do not want to save the Data Integration assets after the test case run.</p>
ignoreCase	No	<p>Defines whether you want to ignore casing differences in the data.</p> <p>Enter <b>true</b> to ignore casing differences in the data.</p> <p>Enter <b>false</b> to consider the casing differences in the data as a mismatch.</p>
trimString	No	<p>Defines whether you want to trim leading and trailing white spaces in string values.</p> <p>Enter <b>true</b> to trim leading and trailing white spaces in string values.</p> <p>Enter <b>false</b> to retain leading and trailing white spaces in string values.</p> <p>Default is <b>false</b>.</p>
badRecordLimit	No	<p>Defines the maximum number of unmatched, extra, and missing records to show in the detailed test results. Enter the value as <b>100</b>, <b>500</b>, or <b>1000</b>.</p> <p>Default is <b>100</b>.</p>

Field	Required?	Description
colMappings	Yes	<p>Defines the name, type, precision, and scale for each column mapping. Use the following syntax to map columns in the first connection and second connection:</p> <pre> "colMappings": [   {     "aggrFunc": "&lt;function1,function2,functionn&gt;",     "leftColName": "&lt;value&gt;",     "leftColType": "&lt;value&gt;",     "rightColName": "&lt;value&gt;",     "rightColType": "&lt;value&gt;",     "rightColPrecision": &lt;value&gt;,     "rightColScale": &lt;value&gt;,     "leftColPrecision": &lt;value&gt;,     "leftColScale": &lt;value&gt;,   },   {     "leftColName": "&lt;value&gt;",     "leftColType": "&lt;value&gt;",     "rightColName": "&lt;value&gt;",     "rightColType": "&lt;value&gt;",     "rightColPrecision": &lt;value&gt;,     "rightColScale": &lt;value&gt;,     "leftColPrecision": &lt;value&gt;,     "leftColScale": &lt;value&gt;,     "aggrFunc": "min"   } ], </pre> <p>Use the <b>aggrFunc</b> field to define the aggregation functions that Data Validation must use to compare the table. Separate multiple aggregation functions with a comma. The aggregation function names must be in uppercase.</p> <p>An aggregation functions test retrieves summarized information about the data contained in the data sources. Use aggregation to verify whether all records were moved or to identify incorrect logic in WHERE clauses.</p> <p>Based on the data type of the columns, you can use one or more of the following aggregation functions:</p> <ul style="list-style-type: none"> <li>- <b>COUNT</b>. Counts the number of rows that contain non-null values for a string or numeric column.</li> <li>- <b>COUNT_ROWS</b>. Counts the number of rows for a string or numeric column. Includes rows that contain nulls.</li> <li>- <b>MIN</b>. Calculates the minimum value for a numeric column.</li> <li>- <b>MAX</b>. Calculates the maximum value for a numeric column.</li> <li>- <b>AVG</b>. Calculates the average value of a numeric column.</li> <li>- <b>SUM</b>. Calculates the total value of a numeric column.</li> </ul>
sampling	Yes	<p>Configure data sampling for Data Validation to compare between sample source data and sample target data. The less data that Data Validation samples, the faster the test runs.</p>

Field	Required?	Description
		<p>Use the following syntax to specify the sampling type and value:</p> <pre>"sampling": {   "type": "&lt;value&gt;",   "val": &lt;value&gt; }</pre> <p>Use one of the following options:</p> <ul style="list-style-type: none"> <li>- Enter the <b>type</b> as <b>percentage</b> to sample a percentage of the rows. The test report shows records that are missing or unmatched in the Connection 2 table, but doesn't show extra records if any exist. Enter the <b>val</b> as <b>1, 5, 10, 25, 50, or 75</b>. For example, enter:  <pre>"sampling": { "type": "percentage",   "val": 25 } }</pre> </li> <li>- Enter the <b>type</b> as <b>first</b> to sample a defined number of rows at the beginning of the table. Enter the <b>val</b> as <b>100, 1000, or 10000</b>. For example, enter:  <pre>"sampling": { "type": "first", "val": 100 } }</pre> </li> <li>- Enter the <b>type</b> as <b>last</b> to sample a defined number of rows at the end of the table. Enter the <b>val</b> as <b>100, 1000, or 10000</b>. For example, enter:  <pre>"sampling": { "type": "last", "val": 1000 } }</pre> </li> <li>- Enter the <b>type</b> as <b>where_clause</b> to sample those rows that meet the condition specified in the WHERE clause. If you use the <b>where_clause</b> option, you must specify a valid where clause in at least one of the following attributes: <ul style="list-style-type: none"> <li>- leftConnWhereClause</li> <li>- rightConnWhereClause</li> </ul> </li> </ul>

#### PUT response

If the test case was updated successfully, the PUT request returns a 200 `Successful operation` response and returns the following response fields:

Field	Type	Description
testCaseId	String	ID of the test case.
frsDocId	String	ID of the test case in the Informatica repository.
testCaseName	String	Name of the test case.

If the test case update failed, the PUT request returns a 400 `Bad request` response. If the test case ID was not found, the PUT request returns a 404 `Test Case Not Found` response. If there was a server issue, the PUT request returns a 500 `Internal Server` response.

The PUT request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Running a test case

Use the following URI to run a test case:

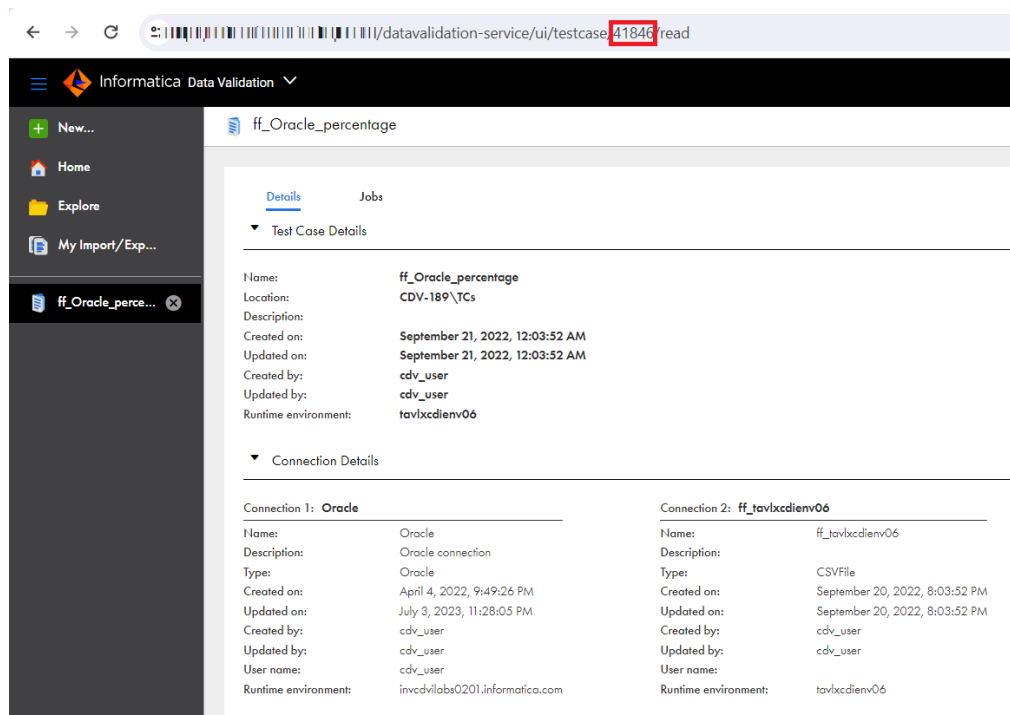
POST /datavalidation-service/api/v1/testcase/run/{testCaseId}

Use the IDS-SESSION-ID header in the POST request.

### POST request

Use the test case ID to run a test case.

When you open a test case, the numeric value that you see in the URI is the test case ID. For example, in the following image, the test case ID is 41846:





### POST response

If the test case ran successfully, the POST request returns a `200 Successful operation` response and returns the following response fields:

Field	Type	Description
testCaseId	String	ID of the test case.
testCaseRunId	String	Run ID of the test case.
sequenceName	String	Instance name or name of the test case job. The sequence name uses the following format: <code>&lt;test_case_name&gt;__&lt;counter&gt;</code> . The counter is set to <code>001</code> when you run a test case for the first time. The counter is incremented by 1 for each test case run thereafter.
startDateTime	String	Date and time when the test case run started.

If the test case run failed, the POST request returns a `400 Bad request` response. If the test case ID was not found, the POST request returns a `404 Test Case Not Found` response. If the test case run failed, the POST request returns a `500 Internal Server` response.

The POST request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Viewing a test case

Use the following URI to view the details of a test case:

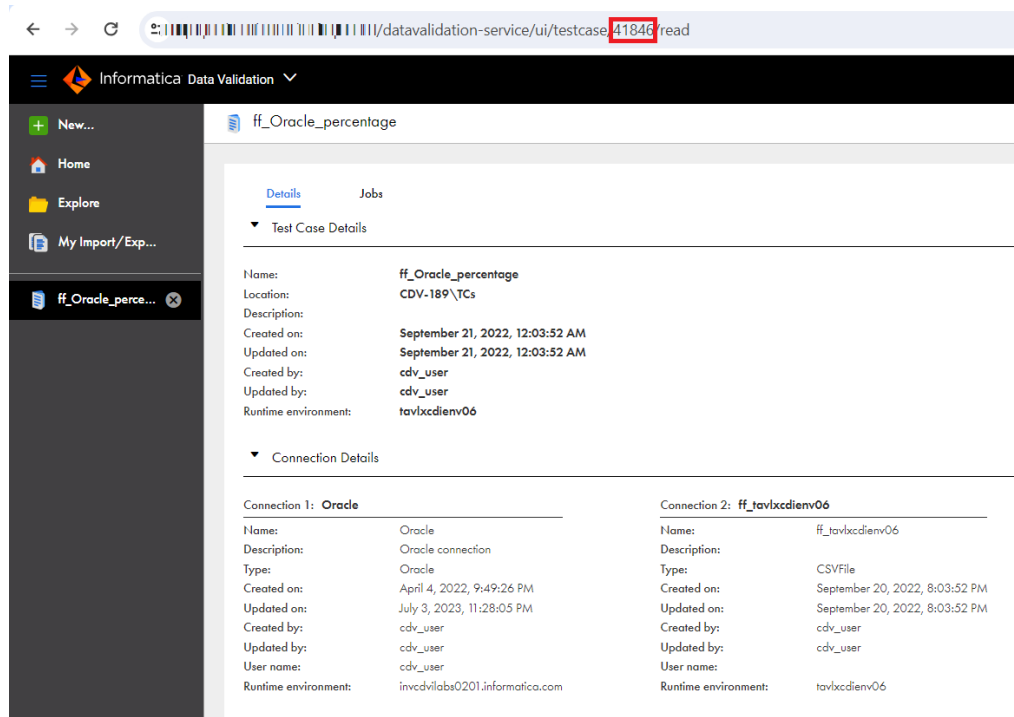
```
GET /datavalidation-service/api/v1/testcase/{testCaseId}
```

Use the `IDS-SESSION-ID` header in the GET request.

### GET request

Use the test case ID to view a test case.

When you open a test case, the numeric value that you see in the URI is the test case ID. For example, in the following image, the test case ID is **41846**:



## GET response

If the test case ID was found, the GET request returns a 200 Successful operation response and returns the following response fields:

Field	Type	Description
testCaseld	String	ID of the test case.
orgId	String	ID of the organization that contains the test case.
testCaseName	String	Name of the test case.
description	String	Description of the test case.
leftConnId	String	Connection ID of the first connection.
leftTableName	String	Table name of the first connection.
leftTablePk	Array	Primary key of the table in the first connection.
rightConnId	String	Connection ID of the second connection.
rightTableName	String	Table name of the second connection.
rightTablePk	Array	Primary key of the table in the second connection.
runtimeEnvId	String	Runtime environment where the test case runs.
stagingConnectionId	String	Connection ID of the flat file connection that Data Validation uses to stores reports.

Field	Type	Description
frsDocId	String	ID of the test case in the Informatica repository.
frsDocLocation	String	The location where the test case is saved.
leftConnPath	String	The path of the first connection. Applies to Amazon Redshift v2 and Amazon S3 v2 connections.
rightConnPath	String	The path of the second connection. Applies to Amazon Redshift v2 and Amazon S3 v2 connections.
leftConnAdvancedParams	String	Advanced parameters for the first connection.
rightConnAdvancedParams	String	Advanced parameters for the second connection.
leftConnWhereClause	String	WHERE clause for the first connection.
rightConnWhereClause	String	WHERE clause for the second connection.
createDatetime	String	The date and time when the test case was created.
updateDatetime	String	The date and time when the test case was last updated.
verboseMode	String	Defines the logging level. The value <b>true</b> indicates that the test case uses the verbose mode. The log includes messages of all logging levels. This option is useful for debugging. The value <b>false</b> indicates that the test case uses the standard mode. The log includes only error messages.
keepDIAssets	String	Defines whether you want to save the Data Integration mappings and tasks that Data Validation creates when it runs the test case. The value <b>true</b> indicates that the Data Integration assets will be saved after the test case run. The value <b>false</b> indicates that the Data Integration assets will not be saved after the test case run.
ignoreCase	String	Defines whether you want to ignore casing differences in the data. The value <b>true</b> indicates that the casing differences in the data will be ignored. The value <b>false</b> indicates that the casing differences in the data will be considered as a mismatch.
trimString	String	Defines whether you want to trim leading and trailing white spaces in string values. The value <b>true</b> indicates that the leading and trailing white spaces in string values will be trimmed. The value <b>false</b> indicates that the leading and trailing white spaces in string values will be retained. Default is <b>false</b> .
badRecordLimit	String	Defines the maximum number of unmatched, extra, and missing records to show in the detailed test results. Default is <b>100</b> .

Field	Type	Description
colMappings	Array	Defines the name, type, precision, and scale for each column mapping.
sampling	Array	Defines the sampling type and value used.

If the test case ID is incorrect, the GET request returns a `400 Bad request` response. If the test case ID was not found, the GET request returns a `404 Test Case Not Found` response. If there was a server issue, the GET request returns a `500 Internal Server` response.

The GET request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Viewing run sequences for a test case

Use the following URI to view the run sequences for a test case:

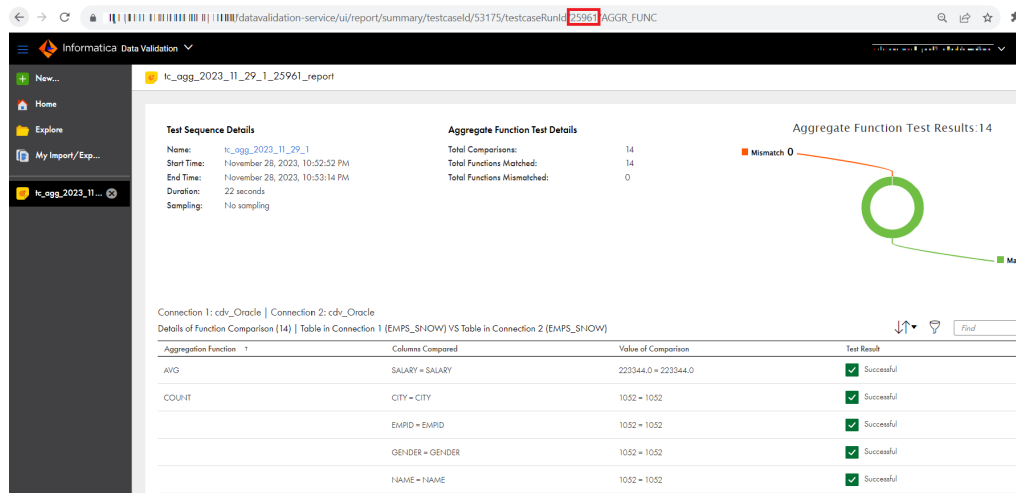
```
GET /datavalidation-service/api/v1/testcase/sequences/{testCaseId}
```

Use the IDS-SESSION-ID header in the GET request.

### GET request

Use the test case ID to view the run sequences for a test case.

When you open a test case, the numeric value that you see in the URI is the test case ID. For example, in the following image, the test case ID is **25961**:



## GET response

If the test case ID was found, the GET request returns a 200 Successful operation response and returns the following response fields:

Field	Type	Description
sequenceName	String	Instance name or name of the test case job. The sequence name uses the following format: <test_case_name>__<counter>. The counter is set to 001 when you run a test case for the first time. The counter is incremented by 1 for each test case run thereafter.
testCaselId	String	ID of the test case.
testCaseRunId	String	Run ID of the test case.
startDateTime	String	Date and time when the test case run started.
finishDateTime	String	Date and time when the test case run completed.
duration	String	Number of milliseconds for which the test case ran.
errorMessage	String	Detailed information about the error message.
samplingStrategy	String	Sampling type and value used.

Field	Type	Description
mappingStrategyType	String	Displays one of the following values based on the comparison method used: <ul style="list-style-type: none"> <li>- <b>FULL</b>. Uses the <b>Value Test - Compare Entire Table</b> method in which Data Validation compares the entire table based on the actual data values.</li> <li>- <b>AGGR_FUNC</b>. Uses the <b>Aggregation Functions Test</b> method in which Data Validation uses aggregation functions to compare the table.</li> </ul>
report	Array	You see the following details: <ul style="list-style-type: none"> <li>- Name of the test case</li> <li>- Total number of records processed</li> <li>- Total number of records processed in the first table</li> <li>- Total number of records processed in the second table</li> <li>- Total number of missing records in the second table</li> <li>- ID of the test case</li> <li>- Total number of matched records</li> <li>- Matching result</li> <li>- Row count comparison result</li> <li>- Total number of unmatched records</li> <li>- Total number of additional records in the second table</li> <li>- Matching status of columns</li> <li>- Date when the test case was run</li> <li>- Date when the report was generated</li> <li>- Aggregation functions used for comparison</li> <li>- Total number of comparisons made</li> <li>- Total number of failed matches</li> </ul>

If the test case ID is incorrect, the GET request returns a 400 `Bad request` response. If the test case ID was not found, the GET request returns a 404 `Test Case Not Found` response. If there was a server issue, the GET request returns a 500 `Internal Server` response.

The GET request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Viewing history of a test case run

Use the following URI to view the history of a test case run:

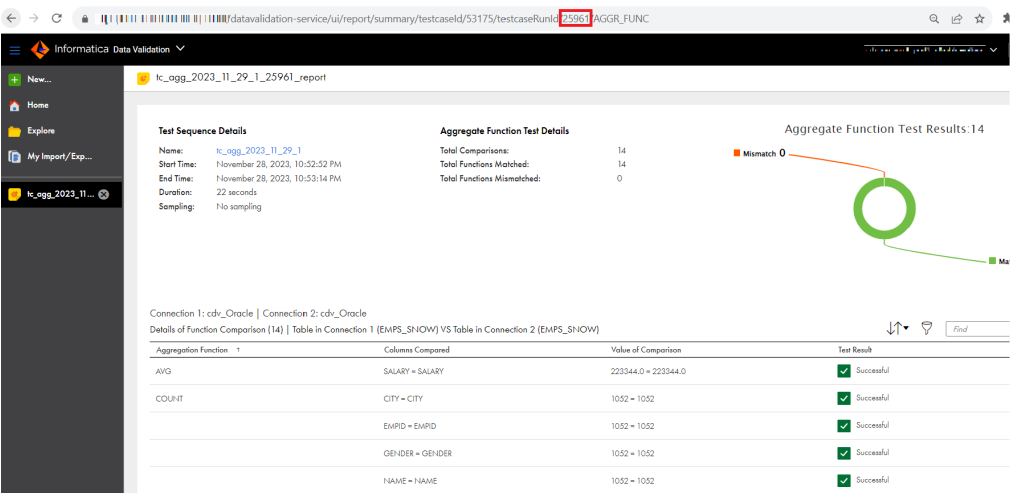
GET /datavalidation-service/api/v1/testcase/history/{testCaseRunId}

Use the IDS-SESSION-ID header in the GET request.

### GET request

Use the test case run ID to view the history of a test case run.

When you open the report of a test case, the numeric value that you see in the URI after the **testCaseRunID** parameter is the test case run ID. For example, in the following image, the test case run ID is **25961**:



**GET response**

If the test case ID was found, the GET request returns a 200 Successful operation response and returns the following response fields:

Field	Type	Description
sequenceName	String	Instance name or name of the test case job. The sequence name uses the following format: <test_case_name>__<counter>. The counter is set to 001 when you run a test case for the first time. The counter is incremented by 1 for each test case run thereafter.
testCaselId	String	ID of the test case.
testCaseRunId	String	Run ID of the test case.
startDateTime	String	Date and time when the test case run started.
finishDateTime	String	Date and time when the test case run completed.
duration	String	Number of milliseconds for which the test case ran.
errorMessage	String	Detailed information about the error message.
samplingStrategy	String	Defines the sampling type and value used.

Field	Type	Description
mappingStrategyType	String	Displays one of the following values based on the comparison method used: <ul style="list-style-type: none"> <li>- <b>FULL</b>. Uses the <b>Value Test - Compare Entire Table</b> method in which Data Validation compares the entire table based on the actual data values.</li> <li>- <b>AGGR_FUNC</b>. Uses the <b>Aggregation Functions Test</b> method in which Data Validation uses aggregation functions to compare the table.</li> </ul>
report	Array	Array that gives the following details: <ul style="list-style-type: none"> <li>- Name of the test case</li> <li>- Total number of records processed</li> <li>- Total number of records processed in the first table</li> <li>- Total number of records processed in the second table</li> <li>- Total number of missing records in the second table</li> <li>- ID of the test case</li> <li>- Total number of matched records</li> <li>- Matching result</li> <li>- Row count comparison result</li> <li>- Total number of unmatched records</li> <li>- Total number of additional records in the second table</li> <li>- Matching status of columns</li> <li>- Date when the test case was run</li> <li>- Date when the report was generated</li> <li>- Aggregation functions used for comparison</li> <li>- Total number of comparisons made</li> <li>- Total number of failed matches</li> </ul>

If the test case run ID is incorrect, the GET request returns a 400 `Bad request` response. If the test case run ID was not found, the GET request returns a 404 `Test Case Not Found` response. If there was a server issue, the GET request returns a 500 `Internal Server` response.

The GET request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.



## CHAPTER 3

# Test case report REST APIs

You can use REST APIs to perform the following tasks:

- View a summary report for a test case run.
- View a detailed report for a test case run.
- Download the report for a test case run.

## Viewing a summary report for a test case run

Use the following URI to download a summary report for a test case run:

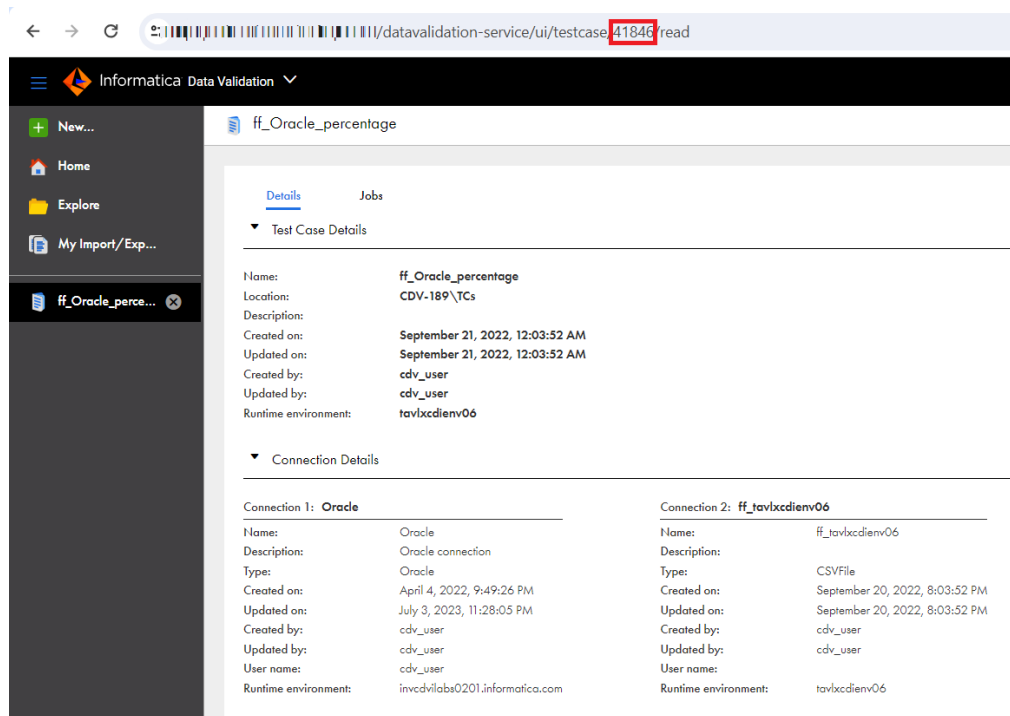
```
GET /datavalidation-service/api/v1/report/summary/testcase/{testCaseId}/run/{testCaseRunId}
```

Use the IDS-SESSION-ID header in the GET request.

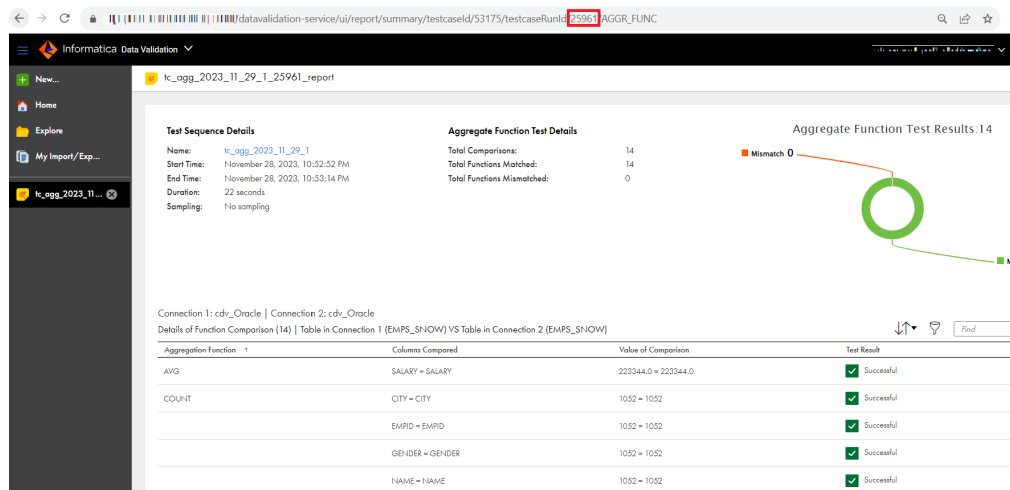
### GET request

Use the test case ID and the test case run ID to download a summary report for a test case run.

When you open a test case, the numeric value that you see in the URI is the test case ID. For example, in the following image, the test case ID is **41846**:



When you open the report of a test case, the numeric value that you see in the URI after the **testCaseRunId** parameter is the test case run ID. For example, in the following image, the test case run ID is **25961**:



## GET response

If the test case ID and test case run ID were found, the GET request returns the following response fields:

Field	Type	Description
testCaseRunId	String	Run ID of the test case.
testCaseId	String	ID of the test case.

Field	Type	Description
rightConnWhereClause	String	WHERE clause used in the second connection.
leftConnWhereClause	String	WHERE clause used in the first connection.
finishDateTime	String	Date and time when the report was generated.
duration	String	Time taken for the report generation in milliseconds.
summary_report	String	You see the following details: <ul style="list-style-type: none"> <li>- Name of the test case</li> <li>- Total number of records processed</li> <li>- Total number of records processed in the first connection</li> <li>- Total number of records processed in the second connection</li> <li>- Total number of missing records in the second table</li> <li>- ID of the test case</li> <li>- Total number of matched records</li> <li>- Matching result</li> <li>- Row count comparison result</li> <li>- Total number of unmatched records</li> <li>- Total number of extra records in the second table</li> <li>- Column matching status</li> <li>- Date when the test case was run</li> <li>- Date when the report was generated</li> </ul>
columns	Array	You see the column mapping along with the column name, type, scale, precision, and details of the aggregation functions used.
functions_compared	String	Aggregation functions used for comparison.
total_comparisons	String	Total number of comparisons made.
total_failed_matches	String	Total number of failed matches.
connection_info	String	Connection name and table name of the first connection and second connection.
sampling	String	Sampling type and value.

If the test case ID or test case run ID was incorrect, the GET request returns a 500 Internal Server response.

The GET request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

# Viewing a detailed report for a test case run

Use the following URI to download a detailed report for a test case run:

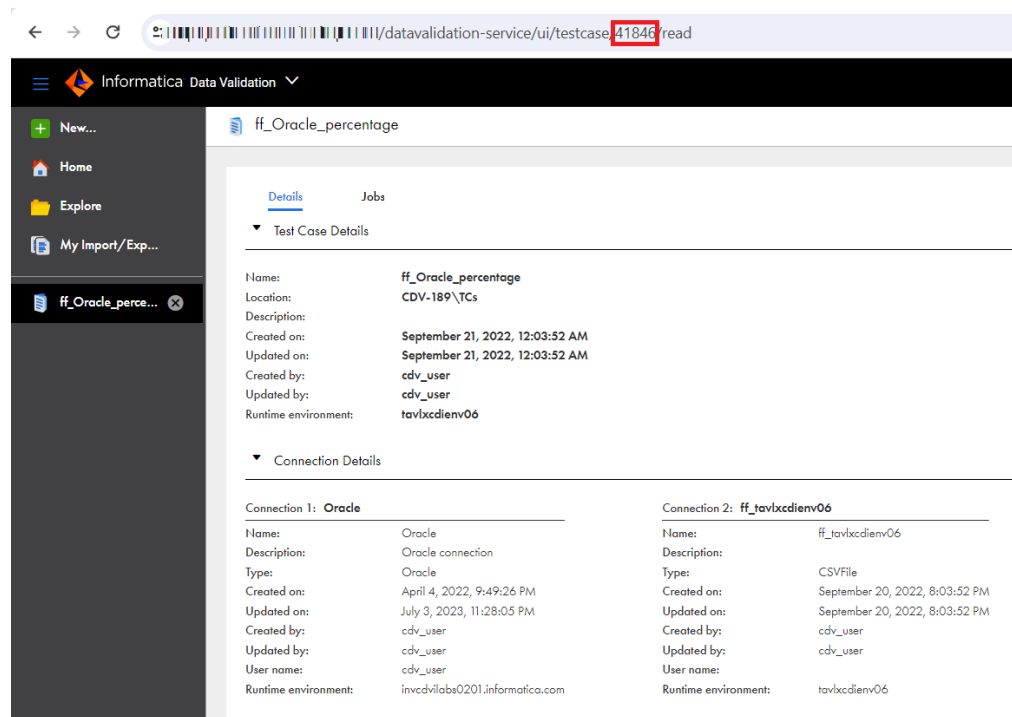
```
GET /datavalidation-service/api/v1/report/detailed/testcase/{testCaseId}/run/{testCaseRunId}
```

Use the IDS-SESSION-ID header in the GET request.

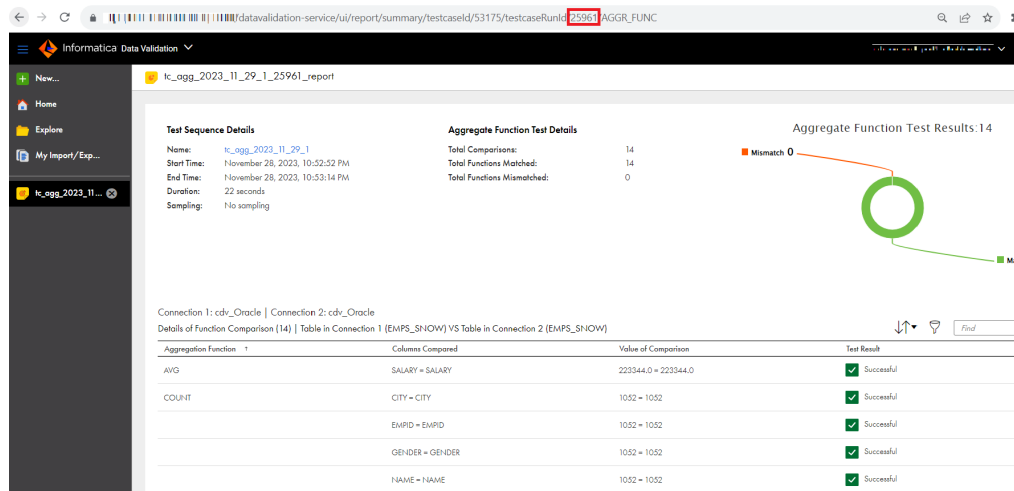
## GET request

Use the test case ID and the test case run ID to download a detailed report for a test case run.

When you open a test case, the numeric value that you see in the URI is the test case ID. For example, in the following image, the test case ID is **41846**:



When you open the report of a test case, the numeric value that you see in the URI after the **testCaseRunID** parameter is the test case run ID. For example, in the following image, the test case run ID is **25961**:



## GET response

If the test case ID and test case run ID were found, the GET request returns the following response fields:

Field	Type	Description
testCaseId	String	ID of the test case.
testCaseRunId	String	Run ID of the test case.
detailed_report	Array	For each column in the first connection and second connection, you see the extra rows in the second table, missing rows in the second table, and unmatched rows.

If the test case ID or test case run ID were incorrect, the GET request returns a 500 Internal Server response.

The GET request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

# Downloading the report for a test case run

Use the following URI to download the report for a test case run:

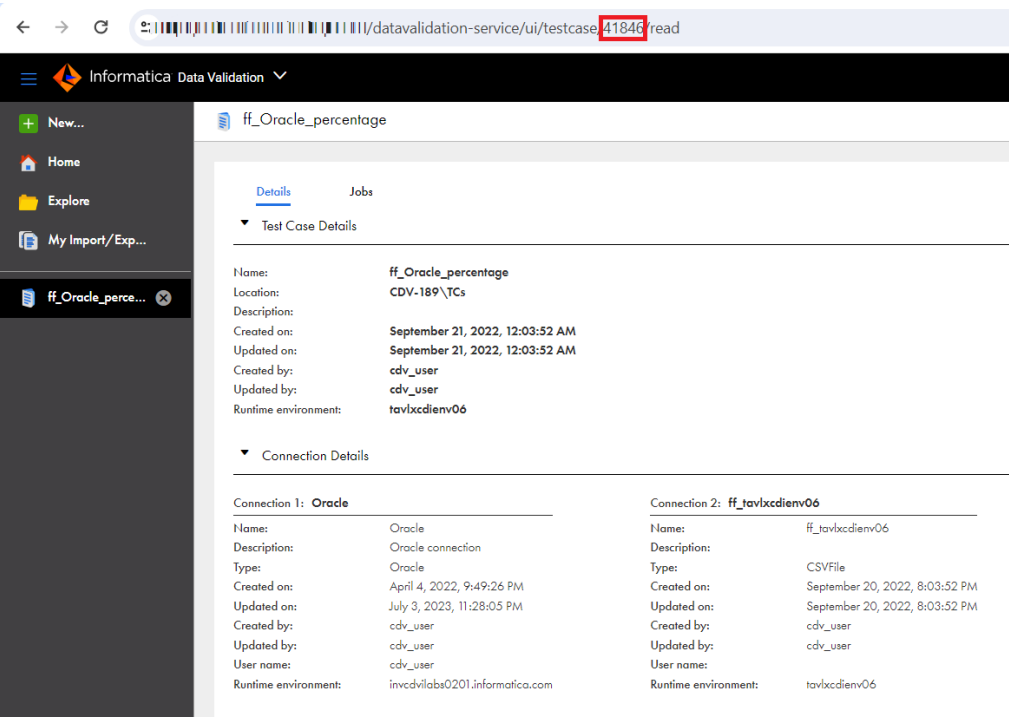
GET /datavalidation-service/api/v1/report/download/testcase/{testCaseId}/run/{testCaseRunId}

Use the IDS-SESSION-ID header in the GET request.

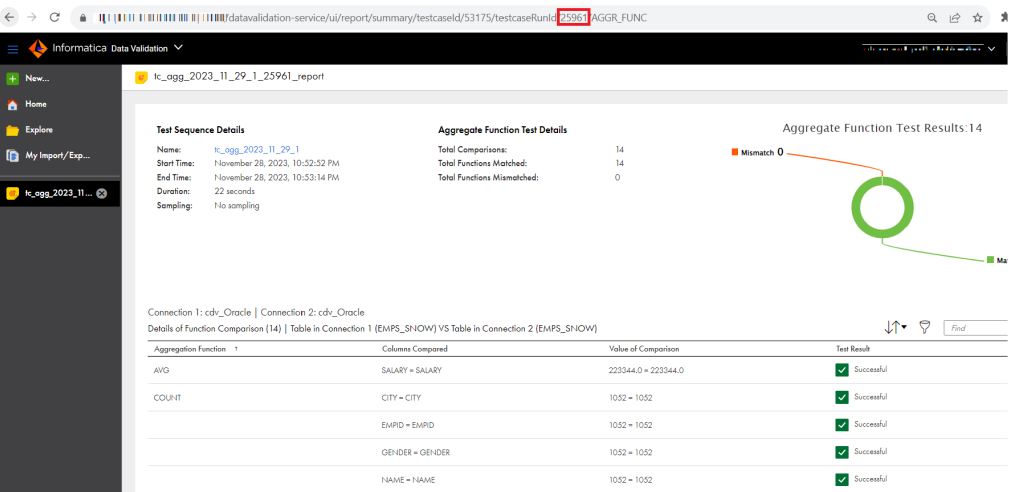
GET request

Use the test case ID and the test case run ID to download the report for a test case run.

When you open a test case, the numeric value that you see in the URI is the test case ID. For example, in the following image, the test case ID is **41846**:



When you open the report of a test case, the numeric value that you see in the URI after the **testCaseRunID** parameter is the test case run ID. For example, in the following image, the test case run ID is **25961**:



GET response

If the test case ID and test case run ID were found, a report is downloaded in the Microsoft Excel format.

The report shows the general details of the job, sampling data, total records, match results, row count, and column matching status. The row count and the column matching status show the data type, precision, and scale for each row.

The data type match result shows whether the data types of the columns compared in both connection tables match.

The report shows details about the missing, unmatched, and extra records in the job.

If the test case ID or test case run ID were incorrect, the GET request returns a 500 Internal Server response.

The GET request also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## CHAPTER 4

# Test suite REST APIs

You can use REST APIs to perform the following tasks:

- Create a test suite.
- Update a test suite.
- Run test cases in a test suite.
- View a test suite.

## Creating a test suite

Use the following URI to create a test suite:

`POST /datavalidation-service/api/v1/testsuite`

Use the IDS-SESSION-ID header in the POST request.

### POST request

Use the following fields in the POST request:

Field	Required?	Description
name	Yes	Name of the test suite.
description	No	Description of the test suite.



Field	Required?	Description
frsDocLocation	Yes	<p>The location to save the test suite.</p> <p>To save the test suite in a project, use the following syntax:</p> <pre>"{"type\":"Project\","id\":"&lt;project_ID&gt;","path\":"&lt;project_name&gt;\"}"</pre> <p>To save the test suite in a folder, use the following syntax:</p> <pre>"{"type\":"Folder\","id\":"&lt;folder_ID&gt;","path\":"&lt;folder_name&gt;\"}"</pre> <p>When you open a project, the numeric value that you see in the URL is the project ID. Similarly, when you open a folder, the numeric value that you see in the URL is the folder ID.</p> <p>By default, the test suite is saved under the <b>Default</b> project.</p>
testCases	Yes	<p>For each test case that you want to include in the test suite, enter the following details:</p> <ul style="list-style-type: none"> <li>- <b>testCaseId</b>. ID of the test case.</li> <li>- <b>name</b>. Name of the test case.</li> <li>- <b>folder</b>. Folder where the test case is saved.</li> <li>- <b>project</b>. Project where the test case is saved.</li> </ul>

#### POST request sample

The following snippet shows a POST request sample to create a test suite:

```
{
  "name": "CDVTestSuite",
  "frsDocLocation": "{"type\":"Project\","id\":"ifwKGemVFM0cWFKC9WvqU8","path\":"CDV\"}",
  "testCases": [
    {
      "name": "TestReqFieldsWithoutVerboseMode",
      "folder": "",
      "project": "CDV",
      "testCaseId": "53201"
    },
    {
      "name": "TestReqFieldsWithoutKeepDIAssetsTrimStringIgnoreCase",
      "folder": "",
      "project": "CDV",
      "testCaseId": "53203"
    }
  ]
}
```

#### POST response

If the test suite was created successfully, the POST request returns a 200 `Successful operation` response and returns the following response fields:

Field	Type	Description
id	String	ID of the test suite.
frsDocId	String	ID of the test suite in the Informatica repository.
testSuiteName	String	Name of the test suite.

If the test suite creation failed, the POST request returns a `500 Internal Server Error` response. It also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Updating a test suite

Use the following URI to update a test suite:

```
PUT /datavalidation-service/api/v1/testsuite/{testSuiteId}
```

Use the `IDS-SESSION-ID` header in the PUT request.

### PUT request

Use the following fields in the PUT request:

Field	Required?	Description
name	Yes	Name of the test suite.
description	No	Description of the test suite.
frsDocLocation	Yes	The location to save the test suite.
testCases	Yes	For each test case that you want to update in the test suite, enter the following details: <ul style="list-style-type: none"><li>- <b>testCaseId</b>. ID of the test case.</li><li>- <b>name</b>. Name of the test case.</li><li>- <b>folder</b>. Folder where the test case is saved.</li><li>- <b>project</b>. Project where the test case is saved.</li></ul>

### PUT response

If the test suite was updated successfully, the PUT request returns a `200 Successful operation` response and returns the following response fields:

Field	Type	Description
name	String	Name of the test suite.
description	String	Description of the test suite.

Field	Type	Description
frsDocLocation	String	The location where the test suite is saved.
testCases	Array	For each test case that was updated in the test suite, you see the following details: <ul style="list-style-type: none"> <li>- <b>testCaseId</b>. ID of the test case.</li> <li>- <b>name</b>. Name of the test case.</li> <li>- <b>folder</b>. Folder where the test case is saved.</li> <li>- <b>project</b>. Project where the test case is saved.</li> </ul>

If the test suite update failed, the PUT request returns a `500 Internal Server Error` response. It also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Running test cases in a test suite

Use the following URI to run specific test cases in a test suite:

`GET /datavalidation-service/api/v1/testsuite/run/{testSuiteId}`

Use the IDS-SESSION-ID header in the POST request.

### POST request

Use the following fields in the POST request:

Field	Type	Description
testCaseIdsToRun	Array	Enter the IDs of the test cases that you want to run in the test suite.

### POST response

If the test cases in the test suite were run successfully, the POST request returns a `200 Successful operation` response and returns the following response fields:

Field	Type	Description
testCaseIdsToRun	Array	You see the IDs of the test cases that were run.

If the test cases in the test suite did not run successfully, the POST request returns a 500 `Internal Server Error` response. It also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## Viewing a test suite

Use the following URI to view a test suite:

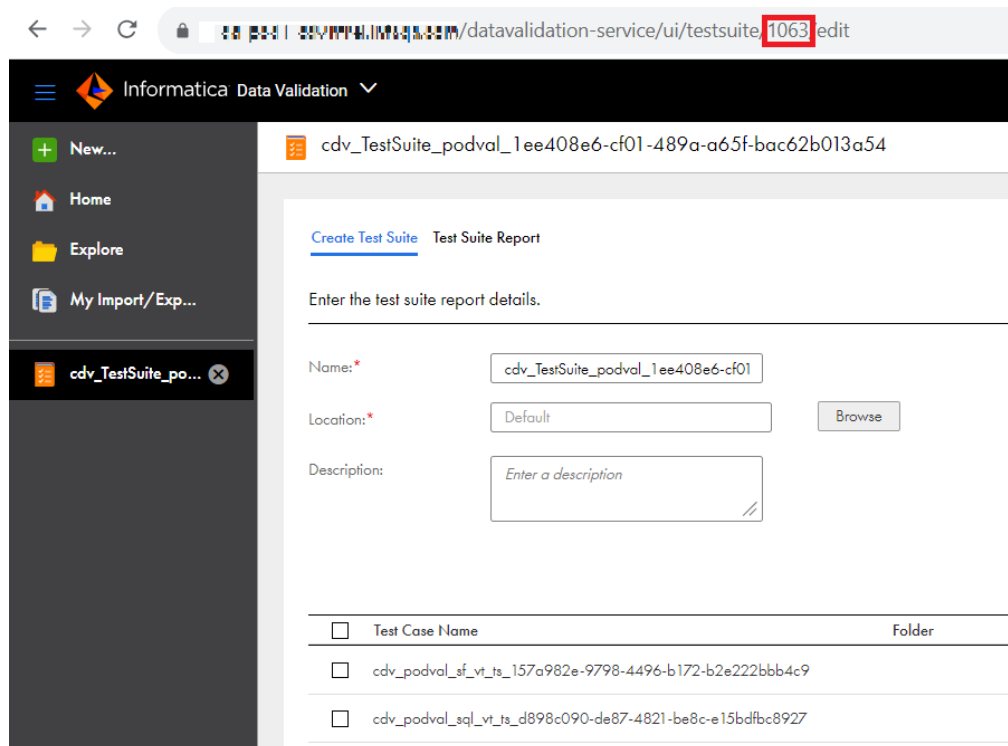
```
GET /datavalidation-service/api/v1/testsuite/{testSuiteId}
```

Use the `IDS-SESSION-ID` header in the GET request.

### GET request

Use the test suite ID to view a test suite.

When you open a test suite, the numeric value that you see in the URI is the test suite ID. For example, in the following image, the test suite ID is **1063**:



### GET response

If the test suite ID was found, the GET request returns a 200 `Successful operation` response and returns the following response fields:

Field	Type	Description
name	String	Name of the test suite.
description	String	Description of the test suite.
frsDocLocation	String	Location where the test suite is saved.
testCases	Array	For each test case within the test suite, you see the following details: <ul style="list-style-type: none"><li>- <b>testCaseId</b>. ID of the test case.</li><li>- <b>name</b>. Name of the test case.</li><li>- <b>folder</b>. Folder where the test case is saved.</li><li>- <b>project</b>. Project where the test case is saved.</li></ul>
testSuiteId	String	ID of the test suite.
orgId	String	ID of the organization that contains the test suite.
createDateTime	String	Date and time when the test suite was created.
updateDateTime	String	Date and time when the test suite was last updated.

If the test suite creation failed, the GET request returns a 500 `Internal Server Error` response. It also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
details	String	Detailed information about the error message.

## CHAPTER 5

# Saved SQL query REST APIs

You can use REST APIs to perform the following tasks:

- Create a saved SQL query.
- Update a saved SQL query.
- View a saved SQL query.
- View the supported database types for a saved SQL query.

## Creating a saved SQL query

Use the following URI to create a saved SQL query:

`POST /datavalidation-service/api/v1/savedSqlQuery`

Use the IDS-SESSION-ID header in the POST request.

### POST request

Use the following fields in the POST request:

Field	Required?	Description
name	Yes	Name of the saved SQL query. The name can contain ASCII, Chinese, Hebrew, and Japanese characters, digits, spaces, and the following characters: , _ - The name can't contain the following characters: ` " ' ! " # \$ % & ( ) * + ' . / : ; < > = ? @ [ ] \ ^ ~ { }
description	No	Description of the saved SQL query.
query	Yes	Saved SQL query to be used. The SQL query can have a maximum of 30,000 characters.
databaseType	Yes	Connection ID and database type for the saved SQL query.

Field	Required?	Description
databaseTypeName	No	Type of the database that the saved SQL query fetches data from. Specify one of the following values: <ul style="list-style-type: none"> <li>- Amazon Redshift v2</li> <li>- Databricks</li> <li>- DB2</li> <li>- Google BigQuery V2</li> <li>- Microsoft Azure Synapse SQL</li> <li>- MySQL</li> <li>- Oracle</li> <li>- PostgreSQL</li> <li>- Snowflake Data Cloud</li> <li>- SqlServer</li> </ul>
frsdocLocation	No	The location to save the query. To save the query in a project, use the following syntax: <pre>"{"type":"Project","id":"&lt;project_ID&gt;","path":"&lt;project_name&gt;"}</pre> To save the query in a folder, use the following syntax: <pre>"{"type":"Folder","id":"&lt;folder_ID&gt;","path":"&lt;folder_name&gt;"}</pre> When you open a project, the numeric value that you see in the URL is the project ID. Similarly, when you open a folder, the numeric value that you see in the URL is the folder ID. By default, the query is saved under the <b>Default</b> project.
validationInfo	No	Connection details to validate the saved SQL query.
connectionId	Yes	ID of the connection that Data Validation uses to validate the saved SQL query. When you open the connection in Administrator, the numeric value that you see in the URL is the connection ID. For example, if the URL is <Informatica Intelligent Cloud Services URL>/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000ATW/read, the connection ID is 014N8B0B000000000ATW.

### POST request sample

The following snippet shows a POST request sample to create a saved SQL query:

```
{
  "savedSqlQuery": {
    "name": "Saved_SQL_Query_EMP_DEPT",
    "description": "Fetches employee details by joining Employee and Department tables",
    "query": "select * from employees e inner join dept d on e.did = d.id",
    "databaseType": {
      "id": "1",
      "databaseTypeName": "Oracle"
    },
    "frsdocLocation": "SSQ"
  },
  "validationInfo": {
    "connectionId": "0140EK0B0000000002W3"
  }
}
```

### POST response

If the saved SQL query was created successfully, the POST request returns a `200 Successful operation` response and the following response fields:

Field	Type	Description
id	String	ID of the saved SQL query.
name	String	Name of the saved SQL query.
frsDocId	String	Location where the SQL query is saved.
validationState	String	Validation status of the SQL query. Displays one of the following values: <ul style="list-style-type: none"><li>- <b>INVALID</b>. The query is not valid. Fix the query and validate it again.</li><li>- <b>VALID</b>. The query is valid and can be used in a test case.</li><li>- <b>NOT VALIDATED</b>. The query has not been validated. You must validate the query to use it in a test case.</li></ul>

If the saved SQL query creation failed, the POST request returns a `400 Bad request` response or a `500 Internal Server Error` response. It also returns the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
requestId	String	String that uniquely identifies the request. Used for debugging purposes.
details	String	Detailed information about the error message.

## Updating a saved SQL query

Use the following URI to update a saved SQL query:

```
PUT /datavalidation-service/api/v1/savedSqlQuery/{savedSqlQueryId}/
```

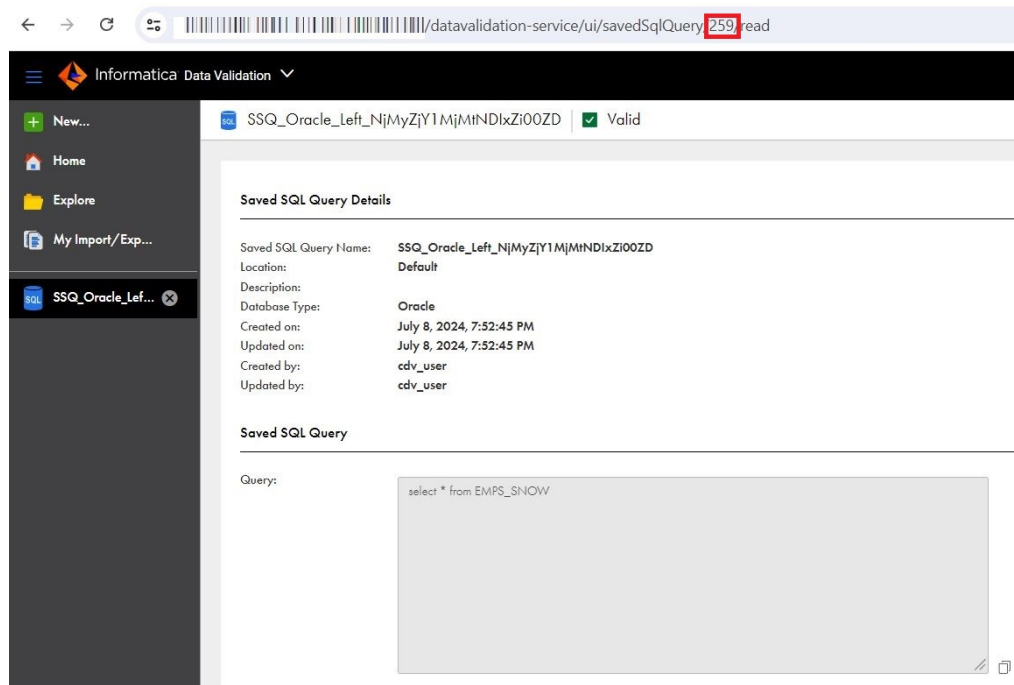
Use the `IDS-SESSION-ID` header in the PUT request:

### PUT request

Use the saved SQL query ID to update the query.

When you open a saved SQL query, the numeric value that you see in the URI is the saved SQL query ID. For example, in the following image, the saved SQL query ID is **259**:





Use the following fields in the PUT request:

Field	Required?	Description
name	Yes	<p>Name of the saved SQL query.</p> <p>The name can contain ASCII, Chinese, Hebrew, and Japanese characters, digits, spaces, and the following characters:</p> <p>‘ _ -</p> <p>The name can't contain the following characters:</p> <p>` " ' ! " # \$ % &amp; ( ) * + ' . / : ; &lt; &gt; = ? @ [ ] \ ^ ~ { }  </p>
description	No	Description of the saved SQL query.
query	Yes	<p>Saved SQL query to be used.</p> <p>The SQL query can have a maximum of 30,000 characters.</p>
databaseType	Yes	Connection ID and database type for the saved SQL query.
databaseTypeName	No	<p>Type of the database that the saved SQL query fetches data from.</p> <p>Specify one of the following values:</p> <ul style="list-style-type: none"> <li>- Amazon Redshift v2</li> <li>- Databricks</li> <li>- DB2</li> <li>- Google BigQuery V2</li> <li>- Microsoft Azure Synapse SQL</li> <li>- MySQL</li> <li>- Oracle</li> <li>- PostgreSQL</li> <li>- Snowflake Data Cloud</li> <li>- SqlServer</li> </ul>

Field	Required?	Description
frsdocLocation	No	<p>The location to save the query.</p> <p>To save the query in a project, use the following syntax:</p> <pre>{\"type\": \"Project\", \"id\": \"&lt;project_ID&gt;\", \"path\": \"&lt;project_name&gt;\"}</pre> <p>To save the query in a folder, use the following syntax:</p> <pre>{\"type\": \"Folder\", \"id\": \"&lt;folder_ID&gt;\", \"path\": \"&lt;folder_name&gt;\"}</pre> <p>When you open a project, the numeric value that you see in the URL is the project ID. Similarly, when you open a folder, the numeric value that you see in the URL is the folder ID.</p> <p>By default, the query is saved under the <b>Default</b> project.</p>
validationInfo	No	Connection details to validate the saved SQL query.
connectionId	Yes	<p>ID of the connection that Data Validation uses to validate the saved SQL query.</p> <p>When you open the connection in Administrator, the numeric value that you see in the URL is the connection ID.</p> <p>For example, if the URL is &lt;Informatica Intelligent Cloud Services URL&gt;/cloudUI/products/administer/main/ConnectionDetailsWS/014N8B0B000000000ATW/read, the connection ID is 014N8B0B000000000ATW.</p>

#### PUT request sample

The following snippet shows a PUT request sample to update a saved SQL query:

```
{
  "savedSqlQuery": {
    "name": "Saved_SQL_Query_EMP_DEPT",
    "description": "Fetches employee details by joining Employee and Department tables",
    "query": "select * from employees e inner join dept d on e.did = d.id",
    "databaseType": {
      "id": "1",
      "databaseTypeName": "Oracle"
    },
    "frsdocLocation": "SSQ"
  },
  "validationInfo": {
    "connectionId": "0140EK0B0000000002W3"
  }
}
```

#### PUT response

If the saved SQL query was updated successfully, the PUT request returns a 200 Successful operation response and the following response fields:

Field	Type	Description
id	String	ID of the saved SQL query.
name	String	Name of the saved SQL query.

Field	Type	Description
frsDocId	String	Location where the SQL query is saved.
validationState	String	Validation status of the SQL query. Displays one of the following values: <ul style="list-style-type: none"> <li>- <b>INVALID</b>. The query is not valid. Fix the query and validate it again.</li> <li>- <b>VALID</b>. The query is valid and can be used in a test case.</li> <li>- <b>NOT VALIDATED</b>. The query has not been validated. You must validate the query to use it in a test case.</li> </ul>

If the saved SQL query update failed, the POST request returns a `500 Internal Server Error` response and the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
requestId	String	String that uniquely identifies the request. Used for debugging purposes.
details	String	Detailed information about the error message.

## Viewing a saved SQL query

Use the following URI to view a saved SQL query:

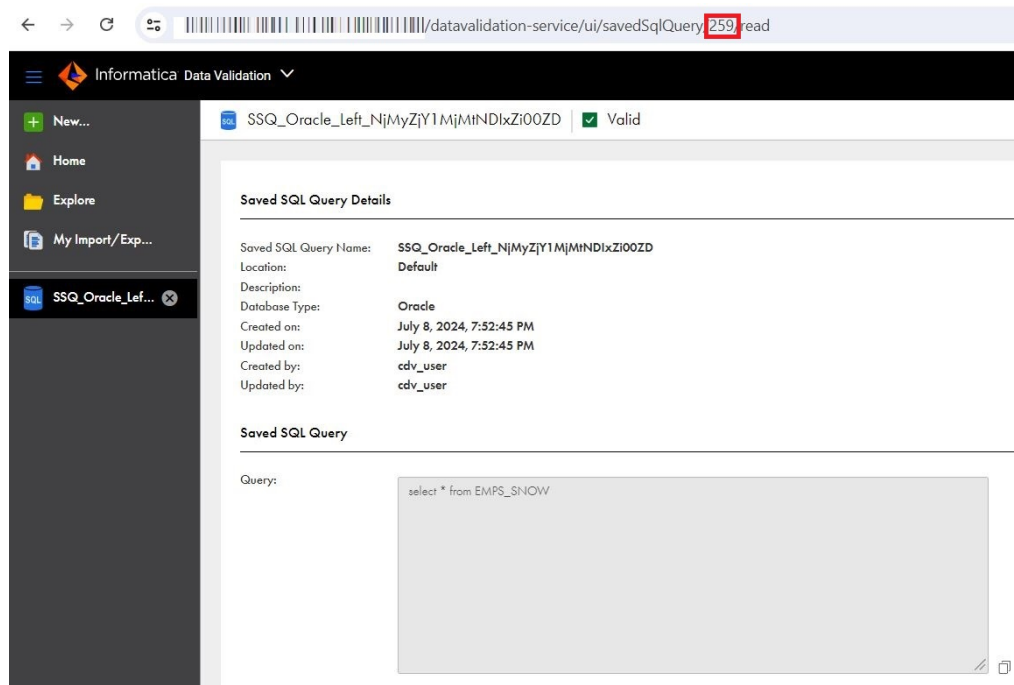
```
GET /datavalidation-service/api/v1/savedSqlQuery/{savedSqlQueryId}/
```

Use the `IDS-SESSION-ID` header in the GET request:

### GET request

Use the saved SQL query ID to view the query.

When you open a saved SQL query, the numeric value that you see in the URI is the saved SQL query ID. For example, in the following image, the saved SQL query ID is **259**:



## GET response

If the saved SQL query was found, the GET request returns a 200 Successful operation response and the following response fields:

Field	Description
name	Name of the saved SQL query.
description	Description of the saved SQL query.
query	Saved SQL query that is used.
databaseType	Connection ID and database type for the saved SQL query.
id	<>
databaseTypeName	Type of the database that the saved SQL query fetches data from. Displays one of the following values: <ul style="list-style-type: none"> <li>- Amazon Redshift v2</li> <li>- Databricks</li> <li>- DB2</li> <li>- Google BigQuery V2</li> <li>- Microsoft Azure Synapse SQL</li> <li>- MySQL</li> <li>- Oracle</li> <li>- PostgreSQL</li> <li>- Snowflake Data Cloud</li> <li>- SqlServer</li> </ul>
frsdocLocation	The location of the saved SQL query.

Field	Description
validationState	Validation status of the SQL query. Displays one of the following values: <ul style="list-style-type: none"> <li>- <b>INVALID</b>. The query is not valid. Fix the query and validate it again.</li> <li>- <b>VALID</b>. The query is valid and can be used in a test case.</li> <li>- <b>NOT VALIDATED</b>. The query has not been validated. You must validate the query to use it in a test case.</li> </ul>
orgId	ID of the organization that contains the saved SQL query.
createDateTime	Date and time when the saved SQL query was created.
updateDateTime	Date and time when the saved SQL query was last updated.

If the saved SQL query was not found, the GET request returns a 500 `Internal Server Error` response and the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
requestId	String	String that uniquely identifies the request. Used for debugging purposes.
details	String	Detailed information about the error message.

## Viewing the supported database types for a saved SQL query

Use the following URI to view the supported database types for a saved SQL query:

`GET /datavalidation-service/api/v1/databaseType`

Use the `IDS-SESSION-ID` header in the GET request.

### GET request

Optionally, you can provide a specific connection ID as a parameter to get the database type of that connection.

If you don't provide a connection ID, the API returns a list of all the supported database types for a saved SQL query.

### GET response

If you had provided a connection ID, the GET request returns a `200 Successful operation` response and the following response fields:

Field	Type	Description
id	String	<>
databaseTypeName	String	<p>If you provided a connection ID, displays the type of the database associated with the specified connection ID.</p> <p>If you had not provided a connection ID, displays the following list of all the supported database types for a saved SQL query:</p> <ul style="list-style-type: none"><li>- Amazon Redshift v2</li><li>- Databricks</li><li>- DB2</li><li>- Google BigQuery V2</li><li>- Microsoft Azure Synapse SQL</li><li>- MySQL</li><li>- Oracle</li><li>- PostgreSQL</li><li>- Snowflake Data Cloud</li><li>- SqlServer</li></ul>

If there was a server issue, the GET request returns a `500 Internal Server` response and the following response fields:

Field	Type	Description
code	String	Code of the error message.
message	String	Error message.
debugMessage	String	Message that can be used for debugging the issue.
requestId	String	String that uniquely identifies the request. Used for debugging purposes.
details	String	Detailed information about the error message.

# INDEX

## C

Cloud Application Integration community  
URL [5](#)  
Cloud Developer community  
URL [5](#)

## D

Data Integration community  
URL [5](#)

## I

Informatica Global Customer Support  
contact information [6](#)  
Informatica Intelligent Cloud Services  
web site [5](#)

## M

maintenance outages [6](#)

## R

REST API  
time zone codes [8](#)

## S

status  
Informatica Intelligent Cloud Services [6](#)  
system status [6](#)

## T

time zone codes  
REST API [8](#)  
trust site  
description [6](#)

## U

upgrade notifications [6](#)

## W

web site [5](#)