



# Zoom API Data Model

*for use with Invantive SQL*

23.0



# Copyright

(C) Copyright 2004-2023 Invantive Software B.V., the Netherlands. All rights reserved.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the publisher.

Despite all the care taken in the compilation of this text, neither the author nor the publisher can accept liability for any damage, which might result from any error, which might appear in this publication.

This manual is a reference guide intended to clarify usage. If data in the sample images match data in your system, the similarity is coincidental.

## Important Safety and Usage Information

**Intended Use and Limitations:** This software, developed by Invantive, is designed to support a variety of business and information technology data processing functions, such as accounting, financial reporting and sales reporting. It is important to note that this software is not designed, tested, or approved for use in environments where malfunction or failure could lead to life-threatening situations or severe physical or environmental damage. This includes, but is not limited to:

- Nuclear facilities: The software should not be used for operations or functions related to the control, maintenance, or operation of nuclear facilities.
- Defense and Military Applications: This software is not suitable for use in defense-related applications, including but not limited to weapon control, military strategy planning, or any other aspects of national defense.
- Aviation: The software is not intended for use in the operation, navigation, or communication systems of any aircraft or air traffic control environments.
- Healthcare and Medicine Production: This software should not be utilized for medical device operation, patient data analysis for critical health decisions, pharmaceutical production, or medical research where its failure or malfunction could impact patient health.
- Chemical and Hazardous Material Handling: This software is not intended for the management, control, or operational aspects of chemical plants or hazardous material handling facilities. Any malfunction in software used in these settings could result in dangerous chemical spills, explosions, or environmental disasters.
- Transportation and Traffic Control Systems: The software should not be used for the control, operation, or management of transportation systems, including railway signal controls, subway systems, or traffic light management. Malfunctions in such critical systems could lead to severe accidents and endanger public safety.
- Energy Grid and Utility Control Systems: This software is not designed for the control or operation of energy grid systems, including electrical substations, renewable energy control systems, or water utility control systems. The failure of software in these areas could lead to significant power outages, water supply disruptions, or other public utility failures, potentially endangering communities and causing extensive damage.
- Other High-Risk Environments: Any other critical infrastructure and environments where a failure of the software could result in significant harm to individuals or the environment.

**User Responsibility:** Users must ensure that they understand the intended use of the software and refrain from deploying it in any setting that falls outside of its designed purpose. It is the responsibility of the user to assess the suitability of the software for their intended application, especially in any scenarios that might pose a risk to life, health, or the environment.

**Disclaimer of Liability:** Invantive disclaims any responsibility for damage, injury, or legal consequences resulting from the use or misuse of this software in prohibited or unintended applications.

# Contents

<b>1</b>	<b>SQL Driver for Zoom API</b>	<b>1</b>
<b>2</b>	<b>SQL Driver Attributes for Zoom API</b>	<b>2</b>
<b>3</b>	<b>Schema: Accounts</b>	<b>15</b>
<b>3.1</b>	<b>Tables</b> .....	<b>15</b>
3.1.1	Accounts .....	15
3.1.2	AccountsByAccountId .....	16
3.1.3	AccountsByAccountId_DomainsManagedDomains .....	17
3.1.4	AccountsByAccountIdManagedDomains .....	18
3.1.5	AccountsByAccountIdSettings .....	19
3.1.6	deleteAccountsByAccountId .....	24
3.1.7	patchAccountsByAccountIdOptions .....	25
3.1.8	patchAccountsByAccountIdSettings .....	26
3.1.9	postAccounts .....	27
<b>4</b>	<b>Schema: Billing</b>	<b>28</b>
<b>4.1</b>	<b>Tables</b> .....	<b>28</b>
4.1.1	AccountsByAccountId_Plan_large_meetingPlans .....	28
4.1.2	AccountsByAccountId_Plan_w_ebinaarPlans .....	30
4.1.3	AccountsByAccountIdBilling .....	32
4.1.4	AccountsByAccountIdPlans .....	33
4.1.5	patchAccountsByAccountIdBilling .....	34
4.1.6	postAccountsByAccountId_Plan_large_meetingPlans .....	35
4.1.7	postAccountsByAccountId_Plan_w_ebinaarPlans .....	37
4.1.8	postAccountsByAccountIdPlans .....	39
4.1.9	postAccountsByAccountIdPlansAddons .....	40
4.1.10	putAccountsByAccountIdPlansAddons .....	41
4.1.11	putAccountsByAccountIdPlansBase .....	42
<b>5</b>	<b>Schema: CloudRecording</b>	<b>44</b>
<b>5.1</b>	<b>Tables</b> .....	<b>44</b>
5.1.1	deleteMeetingsByMeetingIdRecordings .....	44
5.1.2	deleteMeetingsByMeetingIdRecordingsRecordingId .....	45
5.1.3	MeetingsByMeetingIdRecordings .....	46
5.1.4	MeetingsByMeetingIdRecordingsSettings .....	47
5.1.5	patchMeetingsByMeetingIdRecordingsSettings .....	48
5.1.6	putMeetingsByMeetingIdRecordingsRecordingIdStatus .....	49
5.1.7	putMeetingsByMeetingIdRecordingsStatus .....	50
5.1.8	UsersByUserId_MeetingsRecordings .....	51
5.1.9	UsersByUserIdRecordings .....	53
<b>6</b>	<b>Schema: Dashboards</b>	<b>55</b>
<b>6.1</b>	<b>Tables</b> .....	<b>55</b>
6.1.1	Metrics_Crc_ports_usageCrc_ports_hour_usageCrc .....	55
6.1.2	Metrics_MeetingsMeetings .....	56
6.1.3	Metrics_UsersIm .....	58
6.1.4	Metrics_WebinarsWebinars .....	60
6.1.5	MetricsCrc .....	62
6.1.6	MetricsIm .....	63

6.1.7	MetricsMeetings .....	64
6.1.8	MetricsMeetingsByMeetingId .....	65
6.1.9	MetricsMeetingsByMeetingIdParticipants .....	67
6.1.10	MetricsMeetingsByMeetingIdParticipants_ParticipantsDetailsSharing .....	69
6.1.11	MetricsMeetingsByMeetingIdParticipantsParticipantIdQos .....	70
6.1.12	MetricsMeetingsByMeetingIdParticipantsQos .....	72
6.1.13	MetricsMeetingsByMeetingIdParticipantsSharing .....	74
6.1.14	MetricsWebinars .....	75
6.1.15	MetricsWebinarsByWebinarId .....	77
6.1.16	MetricsWebinarsByWebinarIdParticipants .....	78
6.1.17	MetricsWebinarsByWebinarIdParticipants_ParticipantsDetailsSharing .....	80
6.1.18	MetricsWebinarsByWebinarIdParticipantsParticipantIdQos .....	81
6.1.19	MetricsWebinarsByWebinarIdParticipantsQos .....	83
6.1.20	MetricsWebinarsByWebinarIdParticipantsSharing .....	85
6.1.21	MetricsZoomrooms .....	86
6.1.22	MetricsZoomrooms_Past_meetingsMeetingsByZoomroomId .....	88
6.1.23	MetricsZoomroomsByZoomroomId .....	90
<b>7</b>	<b>Schema: Devices</b>	<b>92</b>
<b>7.1</b>	<b>Tables</b> .....	<b>92</b>
7.1.1	deleteH323DevicesByDeviceId .....	92
7.1.2	H323Devices .....	93
7.1.3	patchH323DevicesByDeviceId .....	93
7.1.4	postH323Devices .....	94
<b>8</b>	<b>Schema: Groups</b>	<b>95</b>
<b>8.1</b>	<b>Tables</b> .....	<b>95</b>
8.1.1	deleteGroupsByGroupId .....	95
8.1.2	deleteGroupsByGroupIdMembersMemberId .....	96
8.1.3	Groups .....	97
8.1.4	Groups_Groups .....	98
8.1.5	GroupsByGroupId .....	98
8.1.6	GroupsByGroupIdMembers .....	99
8.1.7	patchGroupsByGroupId .....	101
8.1.8	postGroups .....	102
8.1.9	postGroupsByGroupIdMembers .....	103
<b>9</b>	<b>Schema: IMChat</b>	<b>104</b>
<b>9.1</b>	<b>Tables</b> .....	<b>104</b>
9.1.1	ImChat_SessionsSessions .....	104
9.1.2	ImChatSessions .....	105
9.1.3	ImChatSessions_MessagesBySessionId .....	107
9.1.4	ImChatSessionsBySessionId .....	108
<b>10</b>	<b>Schema: IMGroups</b>	<b>110</b>
<b>10.1</b>	<b>Tables</b> .....	<b>110</b>
10.1.1	deleteImGroupsByGroupId .....	110
10.1.2	deleteImGroupsByGroupIdMembersMemberId .....	111
10.1.3	ImGroups .....	112
10.1.4	ImGroupsByGroupId .....	112
10.1.5	ImGroupsByGroupIdMembers .....	113
10.1.6	patchImGroupsByGroupId .....	115
10.1.7	postImGroups .....	116
10.1.8	postImGroupsByGroupIdMembers .....	117
<b>11</b>	<b>Schema: Meetings</b>	<b>118</b>
<b>11.1</b>	<b>Tables</b> .....	<b>118</b>

11.1.1	deleteMeetingsByMeetingId .....	118
11.1.2	deleteMeetingsByMeetingIdPollsPollId .....	119
11.1.3	MeetingsByMeetingId .....	120
11.1.4	MeetingsByMeetingIdInvitation .....	122
11.1.5	MeetingsByMeetingIdPolls .....	123
11.1.6	MeetingsByMeetingIdPollsPollId .....	124
11.1.7	MeetingsByMeetingIdRegistrants .....	125
11.1.8	PastMeetingsByMeetingIdInstances .....	126
11.1.9	PastMeetingsByMeetingUUID .....	127
11.1.10	PastMeetingsByMeetingUUIDParticipants .....	128
11.1.11	patchMeetingsByMeetingId .....	130
11.1.12	patchMeetingsByMeetingIdLivestream .....	131
11.1.13	patchMeetingsByMeetingIdLivestreamStatus .....	132
11.1.14	postMeetingsByMeetingIdPolls .....	133
11.1.15	postMeetingsByMeetingIdRegistrants .....	134
11.1.16	postUsersByUserIdMeetings .....	135
11.1.17	putMeetingsByMeetingIdPollsPollId .....	137
11.1.18	putMeetingsByMeetingIdRegistrantsStatus .....	138
11.1.19	putMeetingsByMeetingIdStatus .....	139
11.1.20	UsersByUserIdMeetings .....	140
<b>12</b>	<b>Schema: Native</b>	<b>141</b>
<b>12.1</b>	<b>Tables</b> .....	<b>141</b>
12.1.1	NATIVEPLATFORMSCALARREQUESTS: Zoom Native Platform Scalar Requests .....	141
<b>13</b>	<b>Schema: PAC</b>	<b>143</b>
<b>13.1</b>	<b>Tables</b> .....	<b>143</b>
13.1.1	UsersByUserId_Tsp_accountsDedicated_dial_in_numberPac .....	143
13.1.2	UsersByUserId_Tsp_accountsGlobal_dial_in_numbersPac .....	144
13.1.3	UsersByUserIdPac .....	145
<b>14</b>	<b>Schema: Reports</b>	<b>146</b>
<b>14.1</b>	<b>Tables</b> .....	<b>146</b>
14.1.1	Report_DatesDaily .....	146
14.1.2	Report_Telephony_usageTelephone .....	148
14.1.3	Report_UsersUsers .....	149
14.1.4	ReportCloudRecording .....	151
14.1.5	ReportDaily .....	152
14.1.6	ReportMeetings_Tracking_fieldsByMeetingId .....	153
14.1.7	ReportMeetingsByMeetingId .....	154
14.1.8	ReportMeetingsByMeetingId_QuestionsPolls .....	156
14.1.9	ReportMeetingsByMeetingId_QuestionsQuestion_detailsPolls .....	157
14.1.10	ReportMeetingsByMeetingIdParticipants .....	158
14.1.11	ReportMeetingsByMeetingIdPolls .....	159
14.1.12	ReportTelephone .....	160
14.1.13	ReportUsers .....	161
14.1.14	ReportUsersByUserId_MeetingsMeetings .....	163
14.1.15	ReportUsersByUserIdMeetings .....	165
14.1.16	ReportWebinars_Tracking_fieldsByWebinarId .....	166
14.1.17	ReportWebinarsByWebinarId .....	167
14.1.18	ReportWebinarsByWebinarId_QuestionsPolls .....	169
14.1.19	ReportWebinarsByWebinarId_QuestionsQa .....	170
14.1.20	ReportWebinarsByWebinarId_QuestionsQuestion_detailsPolls .....	171
14.1.21	ReportWebinarsByWebinarId_QuestionsQuestion_detailsQa .....	172
14.1.22	ReportWebinarsByWebinarIdParticipants .....	173
14.1.23	ReportWebinarsByWebinarIdPolls .....	175
14.1.24	ReportWebinarsByWebinarIdQa .....	176

<b>15</b>	<b>Schema: TrackingField</b>	<b>177</b>
<b>15.1</b>	<b>Tables</b>	<b>177</b>
15.1.1	deleteV2TrackingFieldsByFieldId	177
15.1.2	patchV2TrackingFieldsByFieldId	178
15.1.3	postV2TrackingFields	179
15.1.4	V2TrackingFields	180
15.1.5	V2TrackingFieldsByFieldId	180
<b>16</b>	<b>Schema: TSP</b>	<b>181</b>
<b>16.1</b>	<b>Tables</b>	<b>181</b>
16.1.1	deleteUsersByUserIdTspTspId	181
16.1.2	patchTsp	182
16.1.3	patchUsersByUserIdTspTspId	183
16.1.4	postUsersByUserId_Dial_in_numbersTsp	184
16.1.5	postUsersByUserIdTsp	186
16.1.6	Tsp	187
16.1.7	Tsp_Dial_in_numbers	187
16.1.8	UsersByUserId_Tsp_accountsDial_in_numbersTsp	188
16.1.9	UsersByUserIdTsp	189
16.1.10	UsersByUserIdTsp_Dial_in_numbersTspId	190
16.1.11	UsersByUserIdTspTspId	191
<b>17</b>	<b>Schema: Users</b>	<b>192</b>
<b>17.1</b>	<b>Tables</b>	<b>192</b>
17.1.1	deleteUsersByUserId	192
17.1.2	deleteUsersByUserIdAssistants	194
17.1.3	deleteUsersByUserIdAssistantsAssistantId	195
17.1.4	deleteUsersByUserIdSchedulers	196
17.1.5	deleteUsersByUserIdSchedulersSchedulerId	197
17.1.6	deleteUsersByUserIdToken	198
17.1.7	patchUsersByUserId	199
17.1.8	patchUsersByUserIdSettings	200
17.1.9	postUsers	201
17.1.10	postUsersByUserIdAssistants	201
17.1.11	postUsersByUserIdPicture	202
17.1.12	putUsersByUserIdEmail	203
17.1.13	putUsersByUserIdPassword	204
17.1.14	putUsersByUserIdStatus	205
17.1.15	Users	206
17.1.16	Users_Group_idsByUserId	207
17.1.17	Users_lm_group_idsByUserId	209
17.1.18	Users_UsersGroup_ids	210
17.1.19	Users_Userslm_group_ids	212
17.1.20	UsersByUserId	213
17.1.21	UsersByUserIdAssistants	215
17.1.22	UsersByUserIdPermissions	216
17.1.23	UsersByUserIdSchedulers	217
17.1.24	UsersByUserIdSettings	218
17.1.25	UsersByUserIdToken	220
17.1.26	UsersEmail	221
17.1.27	UsersVanityName	222
17.1.28	UsersZpk	223
<b>18</b>	<b>Schema: Webhooks</b>	<b>224</b>
<b>18.1</b>	<b>Tables</b>	<b>224</b>
18.1.1	deleteWebhooksByWebhookId	224

18.1.2	patchWebhooksByWebhookId .....	225
18.1.3	patchWebhooksOptions .....	226
18.1.4	postWebhooks .....	227
18.1.5	Webhooks .....	229
18.1.6	WebhooksByWebhookId .....	229

## **19 Schema: Webinars 230**

<b>19.1</b>	<b>Tables .....</b>	<b>230</b>
19.1.1	deleteWebinarsByWebinarId .....	230
19.1.2	deleteWebinarsByWebinarIdPanelists .....	231
19.1.3	deleteWebinarsByWebinarIdPanelistsPanelistId .....	232
19.1.4	deleteWebinarsByWebinarIdPollsPollId .....	233
19.1.5	PastWebinarsByWebinarIdInstances .....	234
19.1.6	patchWebinarsByWebinarId .....	235
19.1.7	postUsersByUserIdWebinars .....	236
19.1.8	postWebinarsByWebinarIdPanelists .....	238
19.1.9	postWebinarsByWebinarIdPolls .....	239
19.1.10	postWebinarsByWebinarIdRegistrants .....	240
19.1.11	putWebinarsByWebinarIdPollsPollId .....	241
19.1.12	putWebinarsByWebinarIdRegistrantsStatus .....	242
19.1.13	putWebinarsByWebinarIdStatus .....	243
19.1.14	UsersByUserIdWebinars .....	244
19.1.15	WebinarsByWebinarId .....	246
19.1.16	WebinarsByWebinarIdPanelists .....	247
19.1.17	WebinarsByWebinarIdPolls .....	248
19.1.18	WebinarsByWebinarIdPollsPollId .....	249
19.1.19	WebinarsByWebinarIdRegistrants .....	250

## **Index 253**

## 1 SQL Driver for Zoom API

Invantive SQL is the fastest, easiest and most reliable way to exchange data with the Zoom API.

Use the "Search" option in the left menu to search for a specific term such as the table or column description. When you already know the term, please use the "Index" option. When you can't find the information needed, please click on the Chat button at the bottom or place your question in the [user community](#). Other users or Invantive Support will try to help you to our best.

Zoom.

The Zoom driver covers 194 tables and 1433 columns.

### Zoom API Clients

Invantive SQL is available on many user interfaces ("clients" in traditional server-client paradigm). All Invantive SQL statements can be exchanged with a close to 100% compatibility across all clients and operating systems (Windows, MacOS, Linux, iOS, Android).

The clients include Microsoft Excel, Microsoft Power BI, Microsoft Power Query, Microsoft Word and Microsoft Outlook. Web-based clients include Invantive Cloud, Invantive Bridge Online as OData proxy, Invantive App Online for interactive apps, Online SQL Editor for query execution and Invantive Data Access Point as extended proxy.

The [Zoom Power BI connector](#) is based on the Invantive SQL driver for Zoom, completed by a high-performance OData connector which works straight on Power BI without any add-on. The OData protocol is always version 4, independent whether the backing platform uses OData, SOAP or another protocol.

For technical users there are command-line editions of Invantive Data Hub running on iOS, Android, Windows, MacOS and Linux. Invantive Data Hub is also often used for enterprise server applications such as ETL. High-volume replication of data taken from the Zoom API into traditional databases such as SQL Server (on-premise and Azure), MySQL, PostgreSQL and Oracle is possible using [Invantive Data Replicator](#). Invantive Data Replicator automatically creates and maintains Zoom datawarehouses, possibly in combination with data from over 70 other (cloud) platforms. Data Replicator supports data volumes up to over 1 TB and over 5.000 companies. The on-premise edition of Invantive Bridge offers an Zoom ADO.net provider.

Finally, online web apps can be build for Zoom using App Online of [Invantive Cloud](#).

### Monitor API Calls

When a query or DML-statement has been executed on Invantive SQL a developer can evaluate the actual calls made to the Zoom API using a query on `sessionios@DataDictionary`. As an alternative, extensive request and response logging can be enabled by setting `log-native-calls-to-disk` to true. In the `%USERPROFILE%\Invantive\NativeLog` folder Invantive SQL will create log files per API request and response.

### Specifications

The SQL driver for Zoom does not support partitioning. Define one data container in a database for each company in Zoom to enable parallel access for data from multiple companies.

An introduction into the concepts of Invantive SQL such as databases, data containers and partitioning can be found in the [Invantive SQL grammar](#).



The configuration can be changed using various attributes during log on and use. A full list of configuration options is listed in the [driver attributes](#) <sup>2</sup>.

The catalog name is used to compose the full qualified name of an object like a table or view. The schema name is used to compose the full qualified name of an object like a table or view. On Zoom the comparison of two texts is case sensitive by default.

Changes and bug fixes on the Zoom SQL driver can be found in the [release notes](#). There is currently no specific section on the [Invantive forums](#) for Zoom. Please reach out to other users of Zoom by leaving a question or contact request.

Driver code for use in settings.xml: `ZOOM`

Alias: `zoom`

Recommended alias: `zom`

More technical documentation as provided by the supplier of the Zoom API on the native API-connection used can be found at <https://marketplace.zoom.us/docs/api-reference/zoom-api>.

General documentation on Zoom is available at <https://zoom.com/>

Updated: 16-06-2022 17:21 using Invantive SQL version 22.1.46-BETA+3385.

## 2 SQL Driver Attributes for Zoom API

The SQL driver for Zoom has many attributes that can be finetuned to improve handling in scenarios with unreliable network connections to the API server of Zoom or high-volumes of data. Also, many drivers have driver-specific attributes to finetune actual behaviour or handle data not matching specifications.

The Zoom driver attributes are assigned a default value which seldom requires change. However, changes can be applied when needed on four levels, which are reflected in the table below by separate checkmarks:

- Connection string: the connection string from the settings\*.xml file and applied during log on.
- Set SQL statement: a set SQL-statement to be executed once connection has been established.
- Drivers file: the providers.xml file (obsolete starting release 17.32).
- Log on: value to be specified interactively by user during log on in a user interface.

The connection string for Zoom can be found in the settings\*.xml file used for the database. Settings\*.xml files are typically located in the %USERPROFILE%\invantive folder in most deployment scenarios. The reference manuals contain instructions how to relocate the settings\*.xml files. Each data container of a database in the connection string can have a `connectionString` element specifying the name and values of attributes. Both name and value must be properly escaped according to XML-semantics. Actual application of the value is solely done during log on. A new connection must be established to change the value of a driver attribute using a connection string.

The set SQL statement can be executed after log on. The syntax is: `set NAME VALUE`, or for a distributed database: `set NAME@ALIAS VALUE`. In some scenarios you may need to enclose the driver attribute name in square brackets to escape it from parsing, for instance when a reserved SQL keyword is part of the name. The new value takes effect straight after execution of the set-statement. The set-statement can be executed as often as needed during a session.

Driver attributes that can be interactively set to a value are typically presented in the log on window. Depending on the platform and design decisions of the user interface designer, some or all of the available driver attributes can have been made available.

The Zoom driver can be configured using the following attributes:

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
add-odata-mandatory-filters	Whether to automatically add OData filters deemed necessary by the platform.	OData	False	✓	✓	✓	
analysis-enforce-row-uniqueness	Use for analysis only! Enforce rows to be unique.	Shared	False	✓	✓	✓	
api-access-token	Access Token is a security token for multiple OAuth2 Flows. With an Access Token you can access protected resources. An Access Token must be stored securely since once compromised allows access to your protected resources.	OData		✓		✓	✓
api-client-id	The client ID is a unique identifier of your application. It is generated by registering an application.	OData		✓		✓	✓
api-client-secret	The client secret is to be kept confidential. Such as a password for a logon code, the client secret is the confidential part of an app identified by a client ID. It is needed during the OAuth2 Code Grant Flow together with the refresh token to get access.	OData		✓		✓	✓
api-pre-expiry-refresh-sec	The number of seconds before the token expires to acquire a new token.	OData		✓	✓	✓	
api-redirect-url	The redirect URI is the website a browser session is redirected to after the OAuth2 authentication process has been completed.	OData		✓		✓	✓
api-refresh-token	Refresh Token is a security token for the OAuth2 Code Grant Flow. With a Refresh Token and client secret you can retrieve a renewed access token to access protected resources. A Refresh Token and client secret must be stored securely since once compromised allows access to your protected resources.	OData		✓		✓	✓
api-scope	The scope to request an OAuth token for.	OData		✓		✓	
api-token-url	The token URI is the OAuth2 endpoint to exchange tokens.	OData		✓		✓	
api-url	URL to access the API.	OData		✓		✓	
bulk-delete-page-size-rows	Number of rows to delete per batch when bulk deleting	Shared	10000	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
bulk-insert-page-size-bytes	Approximate maximum size in bytes of batch when bulk inserting	Shared	10000000	✓	✓	✓	
bulk-insert-page-size-rows	Number of rows to insert per batch when bulk inserting	Shared	250	✓	✓	✓	
download-error-400-bad-request-max-tries	Maximum number of tries when OData server reports bad format during retrieval of data.		3	✓	✓	✓	
download-error-400-bad-request-sleep-initial-ms	Initial sleep in milliseconds between retries when OData server reports that the API server is unavailable during retrieval of data.		500	✓	✓	✓	
download-error-400-bad-request-sleep-max-ms	Maximum sleep in milliseconds between retries when OData server reports that the API server is unavailable during retrieval of data.		5000	✓	✓	✓	
download-error-400-bad-request-sleep-multiplicator	Multiplication factor for sleep between retries OData server reports that the API server is unavailable during retrieval of data.		2	✓	✓	✓	
download-error-408-request-timeout-max-tries	Maximum number of tries when the website reports a HTTP status 408.		10	✓	✓	✓	
download-error-408-request-timeout-sleep-initial-ms	Initial sleep in milliseconds between retries when the website reports a HTTP status 408.		10000	✓	✓	✓	
download-error-408-request-timeout-sleep-max-ms	Maximum sleep in milliseconds between retries when the website reports a HTTP status 408.		300000	✓	✓	✓	
download-error-408-request-timeout-sleep-multiplicator	Multiplication factor for sleep between retries when the website reports a HTTP status 408.		2	✓	✓	✓	
download-error-422-bad-request-max-tries	Maximum number of tries when OData server reports unprocessable entity during retrieval of data.		30	✓	✓	✓	
download-error-422-bad-request-sleep-initial-ms	Initial sleep in milliseconds between retries when OData server reports unprocessable entity during retrieval of data.		10000	✓	✓	✓	
download-error-422-bad-request-sleep-max-ms	Maximum sleep in milliseconds between retries when OData server reports unprocessable entity during retrieval of data.		300000	✓	✓	✓	
download-error-422-bad-request-sleep-multiplicator	Multiplication factor for sleep between retries OData server reports unprocessable entity during retrieval of data.		2	✓	✓	✓	
download-error-429-too-many-requests-max-tries	Maximum number of tries when the website reports that too many requests have been made during a timeslot of one minute or one day.		10	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
dow nload-error-429-too-many-requests-sleep-initial-ms	Initial sleep in milliseconds between retries when the website reports that too many requests have been made during a timeslot of one minute or one day.		10000	✓	✓	✓	
dow nload-error-429-too-many-requests-sleep-max-ms	Maximum sleep in milliseconds between retries when the website reports that too many requests have been made during a timeslot of one minute or one day.		300000	✓	✓	✓	
dow nload-error-429-too-many-requests-sleep-multiplier	Multiplication factor for sleep between retries when the website reports that too many requests have been made during a timeslot of one minute or one day.		2	✓	✓	✓	
dow nload-error-502-server-unavailable-max-tries	Maximum number of tries when OData server reports a bad gateway during retrieval of data.		30	✓	✓	✓	
dow nload-error-502-server-unavailable-sleep-initial-ms	Initial sleep in milliseconds between retries when OData server reports a bad gateway during retrieval of data.		10000	✓	✓	✓	
dow nload-error-502-server-unavailable-sleep-max-ms	Maximum sleep in milliseconds between retries when OData server reports that a bad gateway during retrieval of data.		300000	✓	✓	✓	
dow nload-error-502-server-unavailable-sleep-multiplier	Multiplication factor for sleep between retries OData server reports a bad gateway during retrieval of data.		2	✓	✓	✓	
dow nload-error-503-server-unavailable-max-tries	Maximum number of tries when OData server reports that the API server is unavailable during retrieval of data.		30	✓	✓	✓	
dow nload-error-503-server-unavailable-sleep-initial-ms	Initial sleep in milliseconds between retries when OData server reports that the API server is unavailable during retrieval of data.		10000	✓	✓	✓	
dow nload-error-503-server-unavailable-sleep-max-ms	Maximum sleep in milliseconds between retries when OData server reports that the API server is unavailable during retrieval of data.		300000	✓	✓	✓	
dow nload-error-503-server-unavailable-sleep-multiplier	Multiplication factor for sleep between retries OData server reports that the API server is unavailable during retrieval of data.		2	✓	✓	✓	
dow nload-error-504-gateway-timeout-max-tries	Maximum number of tries when the website reports a gateway timeout.		10	✓	✓	✓	
dow nload-error-504-gateway-	Initial sleep in milliseconds between retries when the website reports a gateway timeout.		10000	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
timeout-sleep-initial-ms							
dow nload-error-504-gatew ay-timeout-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen the w ebsite reports a gatew ay timeout.		300000	✓	✓	✓	
dow nload-error-504-gatew ay-timeout-sleep-multiplier	Multiplication factor for sleep betw een retries w hen the w ebsite reports a gatew ay timeout.		2	✓	✓	✓	
dow nload-error-590-netw ork-connect-timeout-max-tries	Maximum number of tries w hen the w ebsite reports a HTTP status 590.		10	✓	✓	✓	
dow nload-error-590-netw ork-connect-timeout-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen the w ebsite reports a HTTP status 590.		10000	✓	✓	✓	
dow nload-error-590-netw ork-connect-timeout-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen the w ebsite reports a HTTP status 590.		300000	✓	✓	✓	
dow nload-error-590-netw ork-connect-timeout-sleep-multiplier	Multiplication factor for sleep betw een retries w hen the w ebsite reports a HTTP status 590.		2	✓	✓	✓	
dow nload-error-599-netw ork-connect-timeout-max-tries	Maximum number of tries w hen the w ebsite reports a HTTP status 599.		10	✓	✓	✓	
dow nload-error-599-netw ork-connect-timeout-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen the w ebsite reports a HTTP status 599.		10000	✓	✓	✓	
dow nload-error-599-netw ork-connect-timeout-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen the w ebsite reports a HTTP status 599.		300000	✓	✓	✓	
dow nload-error-599-netw ork-connect-timeout-sleep-multiplier	Multiplication factor for sleep betw een retries w hen the w ebsite reports a HTTP status 599.		2	✓	✓	✓	
dow nload-error-argument-exception-max-tries	Maximum number of tries w hen an argument exception is returned w hen dow nloading a blob.		10	✓	✓	✓	
dow nload-error-argument-exception-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen an argument exception is returned w hen dow nloading a blob.		10000	✓	✓	✓	
dow nload-error-argument-exception-	Maximum sleep in milliseconds betw een retries w hen an argument		300000	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
sleep-max-ms	exception is returned when downloading a blob.						
download-error-argument-exception-sleep-multiplicator	Multiplication factor for sleep between retries when an argument exception is returned when downloading a blob.		2	✓	✓	✓	
download-error-internet-download-max-tries	Maximum number of tries when the Internet connection seems down during retrieval of data.		10	✓	✓	✓	
download-error-internet-download-sleep-initial-ms	Initial sleep in milliseconds between retries when the Internet connection seems down during retrieval of data.		10000	✓	✓	✓	
download-error-internet-download-sleep-max-ms	Maximum sleep in milliseconds between retries when the Internet connection seems down during retrieval of data.		300000	✓	✓	✓	
download-error-internet-download-sleep-multiplicator	Multiplication factor for sleep between retries when the Internet connection seems down during retrieval of data.		2	✓	✓	✓	
download-error-io-exception-max-tries	Maximum number of tries when a network I/O connection failure occurs during retrieval of data.		10	✓	✓	✓	
download-error-io-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when a network I/O connection failure occurs during retrieval of data.		10000	✓	✓	✓	
download-error-io-exception-sleep-max-ms	Maximum sleep in milliseconds between retries when a network I/O connection failure occurs during retrieval of data.		300000	✓	✓	✓	
download-error-io-exception-sleep-multiplicator	Multiplication factor for sleep between retries when a network I/O connection failure occurs during retrieval of data.		2	✓	✓	✓	
download-error-json-exception-max-tries	Maximum number of tries when an invalid JSON body is returned.		3	✓	✓	✓	
download-error-json-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when an invalid JSON body is returned.		1000	✓	✓	✓	
download-error-json-exception-sleep-max-ms	Maximum sleep in milliseconds between retries when an invalid JSON body is returned.		10000	✓	✓	✓	
download-error-json-exception-sleep-multiplicator	Multiplication factor for sleep between retries when an invalid JSON body is returned.		2	✓	✓	✓	
download-error-other-exception-max-tries	Maximum number of tries when an unqualified error occurs during retrieval of data.		3	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
dow nload-error-other-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when an unqualified error occurs during retrieval of data.		10000	✓	✓	✓	
dow nload-error-other-exception-sleep-max-ms	Maximum sleep in milliseconds between retries when an unqualified error occurs during retrieval of data.		300000	✓	✓	✓	
dow nload-error-other-exception-sleep-multiplicator	Multiplication factor for sleep between retries when an unqualified error occurs during retrieval of data.		2	✓	✓	✓	
dow nload-error-socket-exception-max-tries	Maximum number of tries when the network connection is forcibly dropped during retrieval of data.		10	✓	✓	✓	
dow nload-error-socket-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when the network connection is forcibly dropped during retrieval of data.		10000	✓	✓	✓	
dow nload-error-socket-exception-sleep-max-ms	Maximum sleep in milliseconds between retries when the network connection is forcibly dropped during retrieval of data.		300000	✓	✓	✓	
dow nload-error-socket-exception-sleep-multiplicator	Multiplication factor for sleep between retries when the network connection is forcibly dropped during retrieval of data.		2	✓	✓	✓	
dow nload-error-web-exception-max-tries	Maximum number of tries when a web connection failure occurs during retrieval of data.		10	✓	✓	✓	
dow nload-error-web-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when a web connection failure occurs during retrieval of data.		10000	✓	✓	✓	
dow nload-error-web-exception-sleep-max-ms	Maximum sleep in milliseconds between retries when a web connection failure occurs during retrieval of data.		300000	✓	✓	✓	
dow nload-error-web-exception-sleep-multiplicator	Multiplication factor for sleep between retries when a web connection failure occurs during retrieval of data.		2	✓	✓	✓	
dow nload-error-web-not-implemented-max-tries	Maximum number of tries when the connection reports not implemented.		1	✓	✓	✓	
dow nload-error-web-not-implemented-sleep-initial-ms	Initial sleep in milliseconds between retries when the connection reports not implemented.		10000	✓	✓	✓	
dow nload-error-web-not-implemented-sleep-max-ms	Maximum sleep in milliseconds between retries when the connection reports not implemented.		300000	✓	✓	✓	
dow nload-error-web-not-implemented-sleep-multiplicator	Multiplication factor for sleep between retries when the connection reports not implemented.		2	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Drivers File	Set from Log On
dow nload-error-w eb-timeout-max-tries	Maximum number of tries w hen the connection reports a timeout.		10	✓	✓	✓	
dow nload-error-w eb-timeout-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen the connection reports a timeout.		1000	✓	✓	✓	
dow nload-error-w eb-timeout-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen the connection reports a timeout.		30000	✓	✓	✓	
dow nload-error-w eb-timeout-sleep-multiplicator	Multiplication factor for sleep betw een retries w hen the connection reports a timeout.		2	✓	✓	✓	
dow nload-error-w eb-unauthorized-max-tries	Maximum number of tries w hen the connection reports an unauthorized error.		1	✓	✓	✓	
dow nload-error-w eb-unauthorized-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen the connection reports an unauthorized error.		10000	✓	✓	✓	
dow nload-error-w eb-unauthorized-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen the connection reports an unauthorized error.		300000	✓	✓	✓	
dow nload-error-w eb-unauthorized-sleep-multiplicator	Multiplication factor for sleep betw een retries w hen the connection reports an unauthorized error.		2	✓	✓	✓	
force-case-sensit-ive-identifiers	Consider identifiers as case-sensit-ive independent of the platform capabilities.	Shared	False	✓	✓	✓	
forced-casing-identi-fiers	Forced casing of identifiers. Choose from Unset, Low er, Upper and Mixed.	Shared		✓	✓	✓	
http-disk-cache-compression-level	Compression level for the HTTP disk cache, ranging from 1 (little) to 9 (intense). Default is 5.	Shared	5	✓	✓	✓	
http-disk-cache-dir-ectory	Directory w here HTTP cache is stored.	Shared	C:\Users\gle3.WS212\In-vantive\Cach e\http\gle3\sh ared	✓	✓	✓	
http-disk-cache-ig-nore-w rite-errors	Whether to ignore w rite errors to disk cache.	Shared	False	✓	✓	✓	
http-disk-cache-max-age-sec	Maximum acceptable age in seconds for use of data in the HTTP disk cache.	Shared	2592000	✓	✓	✓	
http-get-timeout-max-ms	HTTP GET maximum timeout on retry (ms).		300000	✓	✓	✓	
http-get-timeout-ms	HTTP GET timeout (ms).		60000	✓	✓	✓	
http-memory-cache-compression-level	Compression level for the HTTP memory cache, ranging from 1 (little) to 9 (intense). Default is 5.	OData	5	✓	✓	✓	



Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
http-memory-cache-max-age-sec	Maximum acceptable age in seconds for use of data in the HTTP memory cache.	OData	14400	✓	✓	✓	
http-post-timeout-max-ms	HTTP POST maximum timeout on retry (ms).		300000	✓	✓	✓	
http-post-timeout-ms	HTTP POST timeout (ms).		300000	✓	✓	✓	
ignore-http-400-errors	Ignore HTTP 400 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-401-errors	Ignore HTTP 401 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-402-errors	Ignore HTTP 402 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-403-errors	Ignore HTTP 403 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-404-errors	Ignore HTTP 404 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-422-errors	Ignore HTTP 422 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-429-errors	Ignore HTTP 429 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-500-errors	Ignore HTTP 500 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-502-errors	Ignore HTTP 502 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-http-503-errors	Ignore HTTP 503 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
ignore-unknown-path-type	Whether to ignore path types not yet supported. An error will be generated when an unsupported type occurs.		True	✓	✓	✓	✓
ignore-values-unknown-path	Whether to ignore values outside of processed paths. An error will be generated when a value occurs outside a path otherwise.		True	✓	✓	✓	✓
invalid-json-on-get-max-tries	Maximum number of tries when the JSON received on GET is invalid.		10	✓	✓	✓	
invalid-json-on-get-sleep-initial-ms	Initial sleep in milliseconds between retries when the JSON received on GET is invalid.		10000	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
invalid-json-on-get-sleep-max-ms	Maximum sleep in milliseconds between retries when the JSON received on GET is invalid.		300000	✓	✓	✓	
invalid-json-on-get-sleep-multiplicator	Multiplication factor for sleep between retries when the JSON received on GET is invalid.		2	✓	✓	✓	
invalid-json-on-post-max-tries	Maximum number of tries when the JSON received on POST is invalid.		1	✓	✓	✓	
invalid-json-on-post-sleep-initial-ms	Initial sleep in milliseconds between retries when the JSON received on POST is invalid.		10000	✓	✓	✓	
invalid-json-on-post-sleep-max-ms	Maximum sleep in milliseconds between retries when the JSON received on POST is invalid.		300000	✓	✓	✓	
invalid-json-on-post-sleep-multiplicator	Multiplication factor for sleep between retries when the JSON received on POST is invalid.		2	✓	✓	✓	
invantive-sql-compress-sparse-arrays	Whether to compress sparse arrays in result sets during compression.	SQL Engine V1	True	✓	✓	✓	
invantive-sql-correct-invalid-date	Whether to correct dates considered invalid since they are before 01-01-1753. When nullable, they are removed. Otherwise they are replaced by 01-01-1753.	SQL Engine V1	False	✓	✓	✓	
invantive-sql-forward-filters-to-data-containers	Whether to forward filters to data containers.	SQL Engine V1	True	✓	✓	✓	
invantive-sql-share-byte-arrays	Whether to share the memory used by identical byte arrays in result sets during compression.	SQL Engine V1	True	✓	✓	✓	
invantive-sql-share-strings	Whether to share the memory used by identical strings in result sets during compression.	SQL Engine V1	True	✓	✓	✓	
invantive-sql-shuffle-fetch-results-data-containers	Whether to shuffle results fetched from data containers.	SQL Engine V1	False	✓	✓	✓	
invantive-use-cache	Whether to cache the results of a query.	SQL Engine V1	True	✓	✓	✓	
join-set-points-per-request	Maximum number of values in a request when executing a join set.	OData	60	✓	✓	✓	
limit-partition-calls-left	Minimum number of remaining API calls on a partition towards a hard limit. When below, an error is raised.	OData	500	✓	✓	✓	
log-native-calls-to-disk-max-events	Maximum number of events to register from last activation.	Shared		✓	✓	✓	
log-native-calls-to-disk-max-seconds	Maximum number of seconds to register from last activation.	Shared		✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
log-native-calls-to-disk-on-error	Registers native calls to data container backend as disk files when an error occurred.	Shared	False	✓	✓	✓	
log-native-calls-to-disk-on-success	Registers native calls to data container backend as disk files when successful.	Shared	False	✓	✓	✓	
log-native-calls-to-trace	Log native calls to data container backend on the trace.	Shared	False	✓	✓	✓	
maximum-discovered-column-count	Maximum number of discovered columns. An error will be generated when the column exceeds this value.		250	✓	✓	✓	✓
maximum-length-identifiers	Non-default maximum length in characters of identifier names.	Shared		✓	✓	✓	
max-odata-filters	The maximum number of OData filter elements.	OData	100	✓	✓	✓	
max-url-length-accepted	The maximum accepted URL length before raising an error.	Shared	8000	✓	✓	✓	
max-url-length-desired	The maximum desired URL length.	Shared	8000	✓	✓	✓	
metadata-cache-max-age-sec	Maximum acceptable age in seconds for re-use of metadata.	OData		✓	✓	✓	
oauth-unauthorized-max-tries	Maximum number of tries when an OAuth exception occurs.	OData	2	✓	✓	✓	
oauth-unauthorized-sleep-initial-ms	Initial sleep in milliseconds between OAuth reauthentication tries when the OAuth authentication fails.	OData	10000	✓	✓	✓	
oauth-unauthorized-sleep-max-ms	Maximum sleep in milliseconds between OAuth reauthentication tries when the OAuth authentication fails.	OData	1000	✓	✓	✓	
oauth-unauthorized-sleep-multiplicator	Multiplication factor for sleep between OAuth reauthentication tries when the OAuth authentication fails.	OData	2	✓	✓	✓	
page-size-rows	Number of rows to retrieve per page.	Zoom	100	✓	✓	✓	
partition-slot-based-rate-limit-length-ms	Total length in ms across all slots of a partition-based rate limit.	Shared	60000	✓		✓	
partition-slot-based-rate-limit-slots	Number of slots per partition-based rate limit. Null means no slot-based rate limit	Shared		✓		✓	
pre-request-delay-ms	Pre-request delay in milliseconds per request.	Shared	0	✓	✓	✓	
requested-page-size	Preferred number of rows to exchange per round trip; only effective on limited platforms such as AFAS Online	Shared		✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
requests-parallel-max	Maximum number of parallel data requests from individual partitions on the data container.	Shared	32	✓	✓	✓	
simulate-http-400-errors	Simulate HTTP 400 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-400-errors-percentage	Percentage of simulated HTTP 400 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-401-errors	Simulate HTTP 401 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-401-errors-percentage	Percentage of simulated HTTP 401 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-403-errors	Simulate HTTP 403 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-403-errors-percentage	Percentage of simulated HTTP 403 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-408-errors	Simulate HTTP 408 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-408-errors-percentage	Percentage of simulated HTTP 408 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-429-errors	Simulate HTTP 429 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-429-errors-percentage	Percentage of simulated HTTP 429 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-500-errors	Simulate HTTP 500 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-500-errors-percentage	Percentage of simulated HTTP 500 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-502-errors	Simulate HTTP 502 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-502-errors-percentage	Percentage of simulated HTTP 502 errors when exchanging results with the OData endpoint.		0	✓	✓	✓	
simulate-http-503-errors	Simulate HTTP 503 errors when exchanging results with the OData endpoint.		False	✓	✓	✓	
simulate-http-503-errors-percentage	Percentage of simulated HTTP 503 errors when exchanging results		0	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
	w ith the OData endpoint.						
simulate-http-protocol-errors	Simulate HTTP protocol errors w hen exchanging results w ith the OData endpoint.		False	✓	✓	✓	
simulate-http-protocol-errors-percentage	Percentage of simulated HTTP protocol errors w hen exchanging results w ith the OData endpoint.		0	✓	✓	✓	
simulate-http-timeout-errors	Simulate HTTP timeout errors w hen exchanging results w ith the OData endpoint.		False	✓	✓	✓	
simulate-http-timeout-errors-percentage	Percentage of simulated HTTP timeout errors w hen exchanging results w ith the OData endpoint.		0	✓	✓	✓	
slot-based-rate-limit-length-ms	Total length in ms across all slots of a slot-based rate limit.	Shared	60000	✓		✓	
slot-based-rate-limit-slots	Number of slots of a slot-based rate limit. Null means no slot-based rate limit	Shared		✓		✓	
standardize-identifiers	Rew rite all identifiers to the preferred standards as configured by standardize-identifiers-casing and maximum-length-identifiers.	Shared	True	✓	✓	✓	
standardize-identifiers-casing	Rew rite all identifiers to the recommended standard platform-specific casing w hen changing a data model on a case-dependent platform.	Shared	True	✓	✓	✓	
swagger-specification-file	The Swagger file path, such as C:\temp\swagger.json.			✓	✓	✓	✓
swagger-specification-http-disk-cache-max-age-sec	Maximum acceptable age in seconds for use of Swagger specification data in the HTTP disk cache.		86400	✓	✓	✓	
swagger-specification-url	The Swagger URL such as https://example.org/rest/swagger.json.			✓	✓	✓	✓
use-batch-insert	Whether to use batch insert.	OData	True	✓	✓	✓	
use-http-disk-cache-read	Whether to use HTTP responses from previous queries stored on disk to answer the current query.	Shared	True	✓	✓	✓	
use-http-disk-cache-write	Whether to memorize HTTP responses on disk.	Shared	True	✓	✓	✓	
use-http-memory-cache-read	Whether to use HTTP responses from previous queries stored in memory that can answer the current query.	OData	True	✓	✓	✓	
use-http-memory-cache-write	Whether to memorize HTTP responses from previous queries for use by future queries.	OData	True	✓	✓	✓	

## 3 Schema: Accounts

### 3.1 Tables

#### 3.1.1 Accounts

List sub accounts List all the sub accounts under the master account

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts`

Insert Zoom API URL: `/accounts`

Update Zoom API URL: `/accounts`

Delete Zoom API URL: `/accounts`

Field Selection Method: `NotRequired`

Base Path: `accounts[*]`

Select Zoom API Operation: `get /accounts`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Accounts`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>page_number</code>	<code>int64</code>	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	<code>int64</code>	<input type="checkbox"/>	30	The number of records returned within a single API call

### Table Function Columns

The columns of the table function `Accounts` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
account_name	string		<input type="checkbox"/>	Account name
account_type	string		<input type="checkbox"/>	Account type
created_at	datetime		<input type="checkbox"/>	Account creation date/time
id	guid		<input type="checkbox"/>	Account ID
owner_email	string		<input type="checkbox"/>	Account owner email
seats	int64		<input type="checkbox"/>	Account seats
subscription_end_time	datetime		<input type="checkbox"/>	Account subscription end date/time
subscription_start_time	datetime		<input type="checkbox"/>	Account subscription start date/time

### 3.1.2 AccountsByAccountId

Retrieve a sub account under the master account. *<aside>Your account must be a master account and have this privilege to read sub accounts. Zoom only assigns this privilege to trusted partners</aside>*.

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}`

Insert Zoom API URL: `/accounts/{accountId}`

Update Zoom API URL: `/accounts/{accountId}`

Delete Zoom API URL: `/accounts/{accountId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /accounts/{accountId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
created_at	datetime		<input type="checkbox"/>	Account creation date/time
id	string		<input type="checkbox"/>	Account ID
options_meeting_connectors	string		<input type="checkbox"/>	Meeting Connector, multiple values separated by comma
options_pay_mode	string		<input type="checkbox"/>	Payee
options_room_connectors	string		<input type="checkbox"/>	Virtual Room Connector, multiple value separated by comma
options_share_mc	boolean		<input type="checkbox"/>	Enable Share Meeting Connector
options_share_rc	boolean		<input type="checkbox"/>	Enable Share Virtual Room Connector
owner_email	string		<input type="checkbox"/>	Account Owner email
owner_id	string		<input type="checkbox"/>	Account Owner ID
vanity_url	string		<input type="checkbox"/>	Account Vanity URL

### 3.1.3 AccountsByAccountId\_DomainsManagedDomains

Retrieve a sub account's managed domains  
Retrieve a sub account's managed domains under the master account

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/managed_domains`

Insert Zoom API URL: `/accounts/{accountId}/managed_domains`

Update Zoom API URL: `/accounts/{accountId}/managed_domains`

Delete Zoom API URL: `/accounts/{accountId}/managed_domains`

Field Selection Method: NotRequired

Base Path: `domains[*]`

Select Zoom API Operation: `get /accounts/{accountId}/managed_domains`

## Parameters of Table Function



The following parameters can be used to control the behaviour of the table function `AccountsByAccountId_DomainsManagedDomains`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountId_DomainsManagedDomains` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
domain	string		<input type="checkbox"/>	Domain Name
status	string		<input type="checkbox"/>	Domain Status
total_records	int64		<input type="checkbox"/>	Total records

### 3.1.4 AccountsByAccountIdManagedDomains

Retrieve a sub account's managed domains  
Retrieve a sub account's managed domains under the master account

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/managed_domains`

Insert Zoom API URL: `/accounts/{accountId}/managed_domains`

Update Zoom API URL: `/accounts/{accountId}/managed_domains`

Delete Zoom API URL: `/accounts/{accountId}/managed_domains`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /accounts/{accountId}/managed_domains`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountIdManagedDomains`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountIdManagedDomains` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
total_records	int64		<input type="checkbox"/>	Total records

### 3.1.5 AccountsByAccountIdSettings

Retrieve a sub account's settings  
Retrieve a sub account's settings under the master account

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/settings`

Insert Zoom API URL: `/accounts/{accountId}/settings`

Update Zoom API URL: `/accounts/{accountId}/settings`

Delete Zoom API URL: `/accounts/{accountId}/settings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /accounts/{accountId}/settings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountIdSettings`. A value must be provided at all times for required parameters, but op-

tional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountIdSettings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email_notification_alternative_host_reminder	boolean		<input type="checkbox"/>	Notify w hen an alternative host is set or removed from a meeting
email_notification_cancel_meeting_reminder	boolean		<input type="checkbox"/>	Notify host and participants w hen the meeting is cancelled
email_notification_cloud_recording_available_reminder	boolean		<input type="checkbox"/>	Notify host w hen cloud recording is available
email_notification_jbh_reminder	boolean		<input type="checkbox"/>	Notify host w hen participants join the meeting before them
email_notification_low_host_count_reminder	boolean		<input type="checkbox"/>	Notify w hen host licenses are running low
feature_meeting_capacity	int64		<input type="checkbox"/>	Set the maximum number of participants this user can have in a single meeting
in_meeting_alert_guest_join	boolean		<input type="checkbox"/>	Identify guest participants in the meeting/w ebinar
in_meeting_allow_live_streaming	boolean		<input type="checkbox"/>	Allow live streaming
in_meeting_allow_show_zoom_windows	boolean		<input type="checkbox"/>	Show Zoom Desktop application w hen sharing screen
in_meeting_annotation	boolean		<input type="checkbox"/>	Allow participants to use annotation tools to add information to shared screens
in_meeting_anonymous_question_answer	boolean		<input type="checkbox"/>	Allow Anonymous Q&A in Webinar
in_meeting_attendee_on_hold	boolean		<input type="checkbox"/>	Allow hosts to temporarily remove an attendee from the meeting
in_meeting_attention_tracking	boolean		<input type="checkbox"/>	Lets the host see an indicator in the participant panel if a meeting/w ebinar attendee does not

Name	Data Type	Label	Required	Documentation
				have Zoom in focus during screen sharing
in_meeting_auto_answer	boolean		<input type="checkbox"/>	Enable users to see and add contacts to 'auto-answer group' in the contact list on chat. Any call from members of this group will be automatically answered.
in_meeting_auto_saving_chat	boolean		<input type="checkbox"/>	Automatically save all in-meeting chats so that hosts do not need to manually save the text of the chat after the meeting starts
in_meeting_breakout_room	boolean		<input type="checkbox"/>	Allow host to split meeting participants into separate, smaller rooms
in_meeting_chat	boolean		<input type="checkbox"/>	Allow meeting participants to send a message visible to all participants
in_meeting_closed_caption	boolean		<input type="checkbox"/>	Allow host to type closed captions or assign a participant/third party device to add closed captions
in_meeting_co_host	boolean		<input type="checkbox"/>	Allow the host to add co-hosts
in_meeting_custom_live_streaming	boolean		<input type="checkbox"/>	Custom live streaming
in_meeting_custom_service_instructions	string		<input type="checkbox"/>	Custom service instructions
in_meeting_dscp_audio	int64		<input type="checkbox"/>	DSCP Audio
in_meeting_dscp_marking	boolean		<input type="checkbox"/>	DSCP marking
in_meeting_dscp_video	int64		<input type="checkbox"/>	DSCP Video
in_meeting_e2e_encryption	boolean		<input type="checkbox"/>	Require that all meetings are encrypted using AES
in_meeting_far_end_camera_control	boolean		<input type="checkbox"/>	Allow another user to take control of your camera during a meeting
in_meeting_feedback	boolean		<input type="checkbox"/>	Add a Feedback tab to the Windows Settings or Mac Preferences dialog, and also enable users to provide feedback to Zoom at the end of the meeting
in_meeting_file_transfer	boolean		<input type="checkbox"/>	Hosts and participants can send files through the in-meeting chat
in_meeting_group_hd	boolean		<input type="checkbox"/>	Activate higher quality video for host and participants. (This will use more bandwidth.)
in_meeting_original_audio	boolean		<input type="checkbox"/>	Allow users to select original sound in their client settings
in_meeting_p2p_connection	boolean		<input type="checkbox"/>	Peer to Peer connection while only 2 people are in a meeting
in_meeting_p2p_ports	boolean		<input type="checkbox"/>	P2P listening ports range
in_meeting_polling	boolean		<input type="checkbox"/>	Add 'Polls' to the meeting controls.
in_meeting_ports_range	string		<input type="checkbox"/>	Listening ports range, separated by comma (ex 55,56). The ports

Name	Data Type	Label	Required	Documentation
				range must be between 1 to 65535.
in_meeting_post_meeting_feedback	boolean		<input type="checkbox"/>	Display a thumbs up/down survey at the end of each meeting
in_meeting_private_chat	boolean		<input type="checkbox"/>	Allow meeting participants to send a private 1:1 message to another participants
in_meeting_remote_control	boolean		<input type="checkbox"/>	Allow users to request remote control
in_meeting_screen_sharing	boolean		<input type="checkbox"/>	Allow screen sharing
in_meeting_sending_default_email_invites	boolean		<input type="checkbox"/>	Only show default email when sending email invites
in_meeting_show_meeting_control_toolbar	boolean		<input type="checkbox"/>	Always show meeting control toolbar
in_meeting_stereo_audio	boolean		<input type="checkbox"/>	Allow users to select stereo audio in their client settings
in_meeting_use_html_format_email	boolean		<input type="checkbox"/>	Use HTML format email for Outlook plugin
in_meeting_virtual_background	boolean		<input type="checkbox"/>	Allow users to replace their background with any selected image. Choose or upload an image in the Zoom Desktop application settings.
in_meeting_watermark	boolean		<input type="checkbox"/>	Add watermark when viewing shared screen
in_meeting_webinar_question_answer	boolean		<input type="checkbox"/>	Q&A in webinar
in_meeting_whiteboard	boolean		<input type="checkbox"/>	Allow participants to share a whiteboard that includes annotation tools
in_meeting_workplace_by_facebook	boolean		<input type="checkbox"/>	Workplace by facebook
integration_box	boolean		<input type="checkbox"/>	Enables users who join a meeting from their mobile device to share content from their Box account
integration_dropbox	boolean		<input type="checkbox"/>	Enables users who join a meeting from their mobile device to share content from their Dropbox account
integration_google_calendar	boolean		<input type="checkbox"/>	Enables meetings to be scheduled using Google Calendars
integration_google_drive	boolean		<input type="checkbox"/>	Enables users who join a meeting from their mobile device to share content from their Google Drive
integration_kubi	boolean		<input type="checkbox"/>	Enables users to control a connected Kubi device from within a Zoom meeting
integration_microsoft_one_drive	boolean		<input type="checkbox"/>	Enables users who join a meeting from their mobile device to share content from their Microsoft OneDrive account

Name	Data Type	Label	Required	Documentation
recording_account_user_access_recording	boolean		<input type="checkbox"/>	Cloud recordings are only accessible to account members. People outside of your organization cannot open links that provide access to cloud recordings.
recording_auto_delete_cmr_days	int64		<input type="checkbox"/>	When `auto_delete_cmr` is `true` this value will set the number of days before auto deletion of cloud recordings
recording_auto_delete_cmr	boolean		<input type="checkbox"/>	Allow Zoom to automatically delete recordings permanently after a specified number of days
recording_auto_recording	string		<input type="checkbox"/>	Record meetings automatically as they start
recording_cloud_recording_download_host	boolean		<input type="checkbox"/>	Only the host can download cloud recordings
recording_cloud_recording_download	boolean		<input type="checkbox"/>	Cloud Recording Downloads
recording_cloud_recording	boolean		<input type="checkbox"/>	Allow hosts to record and save the meeting in the cloud
recording_local_recording	boolean		<input type="checkbox"/>	Allow hosts and participants to record the meeting to a local file
recording_record_audio_file	boolean		<input type="checkbox"/>	Record an audio only file
recording_record_gallery_view	boolean		<input type="checkbox"/>	Record gallery view with shared screen
recording_record_speaker_view	boolean		<input type="checkbox"/>	Record active speaker with shared screen
recording_recording_audio_transcript	boolean		<input type="checkbox"/>	Automatically transcribe the audio of the meeting or webinar to the cloud
recording_save_chat_text	boolean		<input type="checkbox"/>	Save chat text from the meeting
recording_show_timestamp	boolean		<input type="checkbox"/>	Add a timestamp to the recording
schedule_meting_audio_type	string		<input type="checkbox"/>	Determine how participants can join the audio portion of the meeting
schedule_meting_enforce_login_domains	string		<input type="checkbox"/>	Only signed-in users with a specified domains
schedule_meting_enforce_login_with_domains	boolean		<input type="checkbox"/>	Only signed-in users with a specific domain can join meetings
schedule_meting_enforce_login	boolean		<input type="checkbox"/>	Only signed-in (Zoom users) users can join meetings
schedule_meting_force_pmi_jbh_password	boolean		<input type="checkbox"/>	Require a password for Personal Meetings if attendees can join before host
schedule_meting_host_video	boolean		<input type="checkbox"/>	Start meetings with host video on
schedule_meting_join_before_host	boolean		<input type="checkbox"/>	Allow participants to join the meeting before the host arrives
schedule_meting_not_store_meeting_topic	boolean		<input type="checkbox"/>	Always display "Zoom Meeting" as the meeting topic

Name	Data Type	Label	Required	Documentation
schedule_meting_participant_video	boolean		<input type="checkbox"/>	Start meetings with participant video on. Participants can change this during the meeting.
security_admin_change_name_pic	boolean		<input type="checkbox"/>	Only account administrators can change user's username and picture
security_hide_billing_info	boolean		<input type="checkbox"/>	Hide billing information
security_import_photos_from_devices	boolean		<input type="checkbox"/>	Allow importing of photos from photo library on the user's device
telephony_audio_conference_info	string		<input type="checkbox"/>	3rd party audio conference info
telephony_third_party_audio	boolean		<input type="checkbox"/>	Users can join the meeting using the existing 3rd party audio configuration
zoom_rooms_auto_start_stop_scheduled_meetings	boolean		<input type="checkbox"/>	Automatic start/stop for scheduled meetings
zoom_rooms_cmr_for_instant_meeting	boolean		<input type="checkbox"/>	Cloud recording for instant meetings
zoom_rooms_force_private_meeting	boolean		<input type="checkbox"/>	Transform all meetings to private
zoom_rooms_hide_host_information	boolean		<input type="checkbox"/>	Hide host and meeting ID from private meetings
zoom_rooms_list_meetings_with_calendar	boolean		<input type="checkbox"/>	Display meeting list with calendar integration
zoom_rooms_start_airplay_manually	boolean		<input type="checkbox"/>	Start AirPlay service manually
zoom_rooms_ultrasonic	boolean		<input type="checkbox"/>	Automatic direct sharing using ultrasonic proximity signal
zoom_rooms_upcoming_meeting_alert	boolean		<input type="checkbox"/>	Upcoming meeting alert
zoom_rooms_weekly_system_restart	boolean		<input type="checkbox"/>	Weekly system restart
zoom_rooms_zr_post_meeting_feedback	boolean		<input type="checkbox"/>	Zoom Room post meeting feedback

### 3.1.6 deleteAccountsByAccountId

Disassociate an accountDisassociate a sub account from the master account. This will leave the account intact but the sub account will not longer be associated with the master account.

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}`

Insert Zoom API URL: `/accounts/{accountId}`

Update Zoom API URL: `/accounts/{accountId}`

Delete Zoom API URL: `/accounts/{accountId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `delete /accounts/{accountId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteAccountsByAccountId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `deleteAccountsByAccountId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 3.1.7 patchAccountsByAccountIdOptions

Update a sub account's options Update a sub account's options under the master account

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/options`

Insert Zoom API URL: `/accounts/{accountId}/options`



Update Zoom API URL: `/accounts/{accountId}/options`

Delete Zoom API URL: `/accounts/{accountId}/options`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /accounts/{accountId}/options`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchAccountsByAccountIdOptions`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `patchAccountsByAccountIdOptions` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 3.1.8 patchAccountsByAccountIdSettings

Update a sub account's settings Update a sub account's settings under the master account

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/settings`

Insert Zoom API URL: `/accounts/{accountId}/settings`

Update Zoom API URL: `/accounts/{accountId}/settings`

Delete Zoom API URL: `/accounts/{accountId}/settings`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `patch /accounts/{accountId}/settings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchAccountsByAccountIdSettings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `patchAccountsByAccountIdSettings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 3.1.9 postAccounts

Create a sub accountCreate a sub account under the master account. <aside>Your account must be a master account and have this privilege to create sub account. Zoom only assigns this privilege to trusted partners. The created user will not receive a confirmation email.</aside>.

Catalog: Zoom

Schema: Accounts

This is a read-only table function. The Zoom API may not support changing the data or the In-variantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts`

Insert Zoom API URL: `/accounts`

Update Zoom API URL: `/accounts`

Delete Zoom API URL: `/accounts`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `post /accounts`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postAccounts`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Account

## Table Function Columns

The columns of the table function `postAccounts` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
created_at	string		<input type="checkbox"/>	Account created date time
id	string		<input type="checkbox"/>	Account ID
owner_email	string		<input type="checkbox"/>	Account owner email
owner_id	string		<input type="checkbox"/>	Account Owner ID
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

## 4 Schema: Billing

### 4.1 Tables

#### 4.1.1 AccountsByAccountId\_Plan\_large\_meetingPlans

Retrieve plan information for a sub account  
 Retrieve plan information for a sub account under the master account Only for the sub account which is paid by master account

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans`

Insert Zoom API URL: `/accounts/{accountId}/plans`

Update Zoom API URL: `/accounts/{accountId}/plans`

Delete Zoom API URL: `/accounts/{accountId}/plans`

Field Selection Method: `NotRequired`

Base Path: `plan_large_meeting[*]`

Select Zoom API Operation: `get /accounts/{accountId}/plans`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountId_Plan_large_meetingPlans`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>accountId</code>	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountId_Plan_large_meetingPlans` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>hosts</code>	int64		<input type="checkbox"/>	Account plan number of hosts
<code>plan_audio_callout_countries</code>	string		<input type="checkbox"/>	Call-out countries, multiple value separated by comma
<code>plan_audio_ddi_numbers</code>	int64		<input type="checkbox"/>	Dedicated Dial-In Numbers
<code>plan_audio_premium_countries</code>	string		<input type="checkbox"/>	Premium countries, multiple value separated by comma
<code>plan_audio_tollfree_countries</code>	string		<input type="checkbox"/>	Toll-free countries, multiple value separated by comma
<code>plan_audio_type</code>	string		<input type="checkbox"/>	Additional Audio Conferencing <a href="#plans">plan type</a>

Name	Data Type	Label	Required	Documentation
plan_base_hosts	int64		<input checked="" type="checkbox"/>	Account base plan number of hosts. For a Pro Plan, please select a value between 1 and 9. For a Business Plan, please select a value between 10 and 49. For a Education Plan, please select a value between 20 and 149. For a Free Trial Plan, please select a value between 1 and 9999.
plan_base_type	string		<input checked="" type="checkbox"/>	Account base <a href="#plans">plan type</a>
plan_recording	string		<input type="checkbox"/>	Additional Cloud Recording Plan
plan_room_connector_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_room_connector_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
plan_zoom_rooms_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_zoom_rooms_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>

#### 4.1.2 AccountsByAccountId\_Plan\_webinarPlans

Retrieve plan information for a sub account  
 Retrieve plan information for a sub account under the master account Only for the sub account which is paid by master account

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans`

Insert Zoom API URL: `/accounts/{accountId}/plans`

Update Zoom API URL: `/accounts/{accountId}/plans`

Delete Zoom API URL: `/accounts/{accountId}/plans`

Field Selection Method: NotRequired

Base Path: `plan_webinar[*]`

Select Zoom API Operation: `get /accounts/{accountId}/plans`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountId_Plan_webinarPlans`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will

default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountId_Plan_webinarPlans` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_audio_callout_countries	string		<input type="checkbox"/>	Call-out countries, multiple value separated by comma
plan_audio_ddi_numbers	int64		<input type="checkbox"/>	Dedicated Dial-In Numbers
plan_audio_premium_countries	string		<input type="checkbox"/>	Premium countries, multiple value separated by comma
plan_audio_tollfree_countries	string		<input type="checkbox"/>	Toll-free countries, multiple value separated by comma
plan_audio_type	string		<input type="checkbox"/>	Additional Audio Conferencing <a href="#plans">plan type</a>
plan_base_hosts	int64		<input checked="" type="checkbox"/>	Account base plan number of hosts. For a Pro Plan, please select a value between 1 and 9. For a Business Plan, please select a value between 10 and 49. For an Education Plan, please select a value between 20 and 149. For a Free Trial Plan, please select a value between 1 and 9999.
plan_base_type	string		<input checked="" type="checkbox"/>	Account base <a href="#plans">plan type</a>
plan_recording	string		<input type="checkbox"/>	Additional Cloud Recording Plan
plan_room_connector_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_room_connector_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
plan_zoom_rooms_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_zoom_rooms_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>

Name	Data Type	Label	Required	Documentation
type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>

### 4.1.3 AccountsByAccountIdBilling

Retrieve billing information for a sub account under the master account

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/billing`

Insert Zoom API URL: `/accounts/{accountId}/billing`

Update Zoom API URL: `/accounts/{accountId}/billing`

Delete Zoom API URL: `/accounts/{accountId}/billing`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /accounts/{accountId}/billing`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountIdBilling`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountIdBilling` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
address	string		<input checked="" type="checkbox"/>	Billing Contact's address
apt	string		<input type="checkbox"/>	Billing Contact's apartment/suite
city	string		<input checked="" type="checkbox"/>	Billing Contact's city
country	string		<input checked="" type="checkbox"/>	Billing Contact's country
email	string		<input checked="" type="checkbox"/>	Billing Contact's email address
first_name	string		<input checked="" type="checkbox"/>	Billing Contact's first name
last_name	string		<input checked="" type="checkbox"/>	Billing Contact's last name
phone_number	string		<input checked="" type="checkbox"/>	Billing Contact's phone number
state	string		<input checked="" type="checkbox"/>	Billing Contact's state
zip	string		<input checked="" type="checkbox"/>	Billing Contact's zip/postal code

#### 4.1.4 AccountsByAccountIdPlans

Retrieve plan information for a sub account  
Retrieve plan information for a sub account under the master account  
<aside>Only for the sub account which is paid by master account</aside>

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the In-  
vative SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans`

Insert Zoom API URL: `/accounts/{accountId}/plans`

Update Zoom API URL: `/accounts/{accountId}/plans`

Delete Zoom API URL: `/accounts/{accountId}/plans`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /accounts/{accountId}/plans`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `AccountsByAccountIdPlans`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the



default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID

## Table Function Columns

The columns of the table function `AccountsByAccountIdPlans` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
plan_audio_callout_countries	string		<input type="checkbox"/>	Call-out countries, multiple value separated by comma
plan_audio_ddi_numbers	int64		<input type="checkbox"/>	Dedicated Dial-In Numbers
plan_audio_premium_countries	string		<input type="checkbox"/>	Premium countries, multiple value separated by comma
plan_audio_tollfree_countries	string		<input type="checkbox"/>	Toll-free countries, multiple value separated by comma
plan_audio_type	string		<input type="checkbox"/>	Additional Audio Conferencing <a href="#plans">plan type</a>
plan_base_hosts	int64		<input checked="" type="checkbox"/>	Account base plan number of hosts. For a Pro Plan, please select a value between 1 and 9. For a Business Plan, please select a value between 10 and 49. For an Education Plan, please select a value between 20 and 149. For a Free Trial Plan, please select a value between 1 and 9999.
plan_base_type	string		<input checked="" type="checkbox"/>	Account base <a href="#plans">plan type</a>
plan_recording	string		<input type="checkbox"/>	Additional Cloud Recording Plan
plan_room_connector_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_room_connector_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
plan_zoom_rooms_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_zoom_rooms_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>

### 4.1.5 patchAccountsByAccountIdBilling

Update billing information for a sub account  
Update billing information for a sub account under the master account <aside>Only for the sub account which is a paid account and paid by master account</aside>

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/billing`

Insert Zoom API URL: `/accounts/{accountId}/billing`

Update Zoom API URL: `/accounts/{accountId}/billing`

Delete Zoom API URL: `/accounts/{accountId}/billing`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `patch /accounts/{accountId}/billing`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchAccountsByAccountIdBilling`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `patchAccountsByAccountIdBilling` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 4.1.6 `postAccountsByAccountId_Plan_large_meetingPlans`

Subscribe plans for a sub account  
 <aside>Can only subscribe plans for the sub account which is a free account and paid by master account</aside>

Catalog: Zoom

## Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans`

Insert Zoom API URL: `/accounts/{accountId}/plans`

Update Zoom API URL: `/accounts/{accountId}/plans`

Delete Zoom API URL: `/accounts/{accountId}/plans`

Field Selection Method: `NotRequired`

Base Path: `plan_large_meeting[*]`

Select Zoom API Operation: `post /accounts/{accountId}/plans`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postAccountsById_Plan_large_meetingPlans`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>accountId</code>	string	<input checked="" type="checkbox"/>		The account ID
<code>body</code>	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `postAccountsById_Plan_large_meetingPlans` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>hosts</code>	int64		<input type="checkbox"/>	Account plan number of hosts
<code>plan_audio_callout_countries</code>	string		<input type="checkbox"/>	Call-out countries, multiple value separated by comma
<code>plan_audio_ddi_numbers</code>	int64		<input type="checkbox"/>	Dedicated Dial-In Numbers
<code>plan_audio_premium_countries</code>	string		<input type="checkbox"/>	Premium countries, multiple value separated by comma

Name	Data Type	Label	Required	Documentation
plan_audio_tollfree_countries	string		<input type="checkbox"/>	Toll-free countries, multiple value separated by comma
plan_audio_type	string		<input type="checkbox"/>	Additional Audio Conferencing <a href="#plans">plan type</a>
plan_base_hosts	int64		<input checked="" type="checkbox"/>	Account base plan number of hosts. For a Pro Plan, please select a value between 1 and 9. For a Business Plan, please select a value between 10 and 49. For a Education Plan, please select a value between 20 and 149. For a Free Trial Plan, please select a value between 1 and 9999.
plan_base_type	string		<input checked="" type="checkbox"/>	Account base <a href="#plans">plan type</a>
plan_recording	string		<input type="checkbox"/>	Additional Cloud Recording Plan
plan_room_connector_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_room_connector_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
plan_zoom_rooms_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_zoom_rooms_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>

#### 4.1.7 postAccountsByAccountId\_Plan\_webinarPlans

Subscribe plans for a sub account  
 <aside>Can only subscribe plans for the sub account which is a free account and paid by master account</aside>

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table NativePlatformScalarRequests to upload data to the Zoom API.

Select Zoom API URL: /accounts/{accountId}/plans

Insert Zoom API URL: /accounts/{accountId}/plans

Update Zoom API URL: /accounts/{accountId}/plans

Delete Zoom API URL: /accounts/{accountId}/plans

Field Selection Method: NotRequired

Base Path: plan\_webinar[\*]

Select Zoom API Operation: post /accounts/{accountId}/plans

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postAccountsByAccountId_Plan_webinarPlans`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `postAccountsByAccountId_Plan_webinarPlans` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_audio_callout_countries	string		<input type="checkbox"/>	Call-out countries, multiple value separated by comma
plan_audio_ddi_numbers	int64		<input type="checkbox"/>	Dedicated Dial-In Numbers
plan_audio_premium_countries	string		<input type="checkbox"/>	Premium countries, multiple value separated by comma
plan_audio_tollfree_countries	string		<input type="checkbox"/>	Toll-free countries, multiple value separated by comma
plan_audio_type	string		<input type="checkbox"/>	Additional Audio Conferencing <a href="#plans">plan type</a>
plan_base_hosts	int64		<input checked="" type="checkbox"/>	Account base plan number of hosts. For a Pro Plan, please select a value between 1 and 9. For a Business Plan, please select a value between 10 and 49. For an Education Plan, please select a value between 20 and 149. For a Free Trial Plan, please select a value between 1 and 9999.
plan_base_type	string		<input checked="" type="checkbox"/>	Account base <a href="#plans">plan type</a>
plan_recording	string		<input type="checkbox"/>	Additional Cloud Recording Plan
plan_room_connector_hosts	int64		<input type="checkbox"/>	Account plan number of hosts

Name	Data Type	Label	Required	Documentation
plan_room_connector_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
plan_zoom_rooms_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_zoom_rooms_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>

#### 4.1.8 postAccountsByAccountIdPlans

Subscribe plans for a sub account  
 <aside>Can only subscribe plans for the sub account which is a free account and paid by master account</aside>

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the In-variantive SQL driver for Zoom does not cover it. In the latter case, please use the table NativePlatformScalarRequests to upload data to the Zoom API.

Select Zoom API URL: /accounts/{accountId}/plans

Insert Zoom API URL: /accounts/{accountId}/plans

Update Zoom API URL: /accounts/{accountId}/plans

Delete Zoom API URL: /accounts/{accountId}/plans

Field Selection Method: NotRequired

Select Zoom API Operation: post /accounts/{accountId}/plans

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function postAccountsByAccountIdPlans. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `postAccountsByAccountIdPlans` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
plan_audio_callout_countries	string		<input type="checkbox"/>	Call-out countries, multiple value separated by comma
plan_audio_ddi_numbers	int64		<input type="checkbox"/>	Dedicated Dial-In Numbers
plan_audio_premium_countries	string		<input type="checkbox"/>	Premium countries, multiple value separated by comma
plan_audio_tollfree_countries	string		<input type="checkbox"/>	Toll-free countries, multiple value separated by comma
plan_audio_type	string		<input type="checkbox"/>	Additional Audio Conferencing <a href="#plans">plan type</a>
plan_base_hosts	int64		<input checked="" type="checkbox"/>	Account base plan number of hosts. For a Pro Plan, please select a value between 1 and 9. For a Business Plan, please select a value between 10 and 49. For a Education Plan, please select a value between 20 and 149. For a Free Trial Plan, please select a value between 1 and 9999.
plan_base_type	string		<input checked="" type="checkbox"/>	Account base <a href="#plans">plan type</a>
plan_recording	string		<input type="checkbox"/>	Additional Cloud Recording Plan
plan_room_connector_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_room_connector_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
plan_zoom_rooms_hosts	int64		<input type="checkbox"/>	Account plan number of hosts
plan_zoom_rooms_type	string		<input type="checkbox"/>	Account <a href="#plans">plan type</a>
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 4.1.9 postAccountsByAccountIdPlansAddons

Add an additional plan for sub account  
 Add an additional plan for sub account <aside>Can only add an Additional plan for the sub account which is a paid account and paid by master account</aside>

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans/addons`

Insert Zoom API URL: `/accounts/{accountId}/plans/addons`

Update Zoom API URL: `/accounts/{accountId}/plans/addons`

Delete Zoom API URL: `/accounts/{accountId}/plans/addons`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `post /accounts/{accountId}/plans/addons`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postAccountsByAccountIdPlansAddons`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `postAccountsByAccountIdPlansAddons` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 4.1.10 putAccountsByAccountIdPlansAddons

Update an additional plan for sub account  
 Update an additional plan for sub account  
 Can only update an Additional plan for the sub account which is a paid account and paid by master account

Catalog: Zoom



## Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans/addons`

Insert Zoom API URL: `/accounts/{accountId}/plans/addons`

Update Zoom API URL: `/accounts/{accountId}/plans/addons`

Delete Zoom API URL: `/accounts/{accountId}/plans/addons`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `put /accounts/{accountId}/plans/addons`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putAccountsByAccountIdPlansAddons`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
accountId	string	<input checked="" type="checkbox"/>		The account ID
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `putAccountsByAccountIdPlansAddons` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 4.1.11 putAccountsByAccountIdPlansBase

Update a base plan for a sub account  
Update a base plan for a sub account <aside>Can only update a base plan for the sub account which is a paid account and paid by master account</aside>

Catalog: Zoom

Schema: Billing

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/accounts/{accountId}/plans/base`

Insert Zoom API URL: `/accounts/{accountId}/plans/base`

Update Zoom API URL: `/accounts/{accountId}/plans/base`

Delete Zoom API URL: `/accounts/{accountId}/plans/base`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `put /accounts/{accountId}/plans/base`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putAccountsByIdPlansBase`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>accountId</code>	string	<input checked="" type="checkbox"/>		The account ID
<code>body</code>	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `putAccountsByIdPlansBase` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

## 5 Schema: CloudRecording

### 5.1 Tables

#### 5.1.1 deleteMeetingsByMeetingIdRecordings

Delete a meeting's recordingsDelete a meeting's recordings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings`

Insert Zoom API URL: `/meetings/{meetingId}/recordings`

Update Zoom API URL: `/meetings/{meetingId}/recordings`

Delete Zoom API URL: `/meetings/{meetingId}/recordings`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /meetings/{meetingId}/recordings`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteMeetingsByMeetingIdRecordings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
action	string	<input type="checkbox"/>	trash	The recording delete action (Values: trash, delete)
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

### Table Function Columns

The columns of the table function `deleteMeetingsByMeetingIdRecordings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 5.1.2 deleteMeetingsByMeetingIdRecordingsRecordingId

Delete one meeting recording file Delete one meeting recording file

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}`

Insert Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}`

Update Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}`

Delete Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /meetings/{meetingId}/recordings/{recordingId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteMeetingsByMeetingIdRecordingsRecordingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
action	string	<input type="checkbox"/>	trash	The recording delete action (Values: trash, delete)
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
recordingId	string	<input checked="" type="checkbox"/>		The recording ID

## Table Function Columns

The columns of the table function `deleteMeetingsByMeetingIdRecordingsRecordingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 5.1.3 MeetingsByMeetingIdRecordings

Retrieve a meeting's all recordings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings`

Insert Zoom API URL: `/meetings/{meetingId}/recordings`

Update Zoom API URL: `/meetings/{meetingId}/recordings`

Delete Zoom API URL: `/meetings/{meetingId}/recordings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /meetings/{meetingId}/recordings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingIdRecordings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `MeetingsByMeetingIdRecordings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>account_id</code>	string		<input type="checkbox"/>	ID of the user account
<code>duration</code>	int64		<input type="checkbox"/>	Meeting duration
<code>host_id</code>	string		<input type="checkbox"/>	ID of the user set as host of meeting
<code>id</code>	string		<input type="checkbox"/>	Meeting ID, also know as meeting number
<code>recording_count</code>	string		<input type="checkbox"/>	Recording count
<code>start_time</code>	datetime		<input type="checkbox"/>	Meeting start time
<code>topic</code>	string		<input type="checkbox"/>	Meeting topic
<code>total_size</code>	string		<input type="checkbox"/>	Total size
<code>uuid</code>	string		<input type="checkbox"/>	Meeting unique ID

### 5.1.4 MeetingsByMeetingIdRecordingsSettings

Retrieve a meeting recording's settings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Insert Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Update Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Delete Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /meetings/{meetingId}/recordings/settings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingIdRecordingsSettings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `MeetingsByMeetingIdRecordingsSettings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
approval_type	int64		<input type="checkbox"/>	Approval type
on_demand	boolean		<input type="checkbox"/>	Registration required
password	string		<input type="checkbox"/>	Password protect
send_email_to_host	boolean		<input type="checkbox"/>	Send an email to host when someone registers
share_recording	string		<input type="checkbox"/>	Determine if the meeting recording is shared
show_social_share_buttons	boolean		<input type="checkbox"/>	Show social share buttons on registration page
viewer_download	boolean		<input type="checkbox"/>	Host video

### 5.1.5 patchMeetingsByMeetingIdRecordingsSettings

Update a meeting recording's settings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Insert Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Update Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Delete Zoom API URL: `/meetings/{meetingId}/recordings/settings`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /meetings/{meetingId}/recordings/settings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchMeetingsByMeetingIdRecordingsSettings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting recording Settings
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `patchMeetingsByMeetingIdRecordingsSettings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 5.1.6 putMeetingsByMeetingIdRecordingsRecordingIdStatus

Recover a single recording

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}/status`

Insert Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}/status`

Update Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}/status`

Delete Zoom API URL: `/meetings/{meetingId}/recordings/{recordingId}/status`



Field Selection Method: NotRequired

Select Zoom API Operation: `put /meetings/{meetingId}/recordings/{recordingId}/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putMeetingsByMeetingIdRecordingsRecordingIdStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
recordingId	string	<input checked="" type="checkbox"/>		The recording ID

## Table Function Columns

The columns of the table function `putMeetingsByMeetingIdRecordingsRecordingIdStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 5.1.7 putMeetingsByMeetingIdRecordingsStatus

Recover a meeting's recordings Recover a meeting's recordings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/recordings/status`

Insert Zoom API URL: `/meetings/{meetingId}/recordings/status`

Update Zoom API URL: `/meetings/{meetingId}/recordings/status`

Delete Zoom API URL: `/meetings/{meetingId}/recordings/status`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /meetings/{meetingId}/recordings/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putMeetingsByMeetingIdRecordingsStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `putMeetingsByMeetingIdRecordingsStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 5.1.8 UsersByUserId\_MeetingsRecordings

List all the recordingsList all the recordings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/recordings`

Insert Zoom API URL: `/users/{userId}/recordings`

Update Zoom API URL: /users/{userId}/recordings

Delete Zoom API URL: /users/{userId}/recordings

Field Selection Method: NotRequired

Base Path: meetings[\*]

Select Zoom API Operation: get /users/{userId}/recordings

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserId_MeetingsRecordings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
mc	string	<input type="checkbox"/>	false	Query mc
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
trash	boolean	<input type="checkbox"/>	False	Query trash
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserId_MeetingsRecordings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
account_id	string		<input type="checkbox"/>	ID of the user account
duration	int64		<input type="checkbox"/>	Meeting duration
from	datetime		<input type="checkbox"/>	Start Date,

Name	Data Type	Label	Required	Documentation
host_id	string		<input type="checkbox"/>	ID of the user set as host of meeting
id	string		<input type="checkbox"/>	Meeting ID, also know as meeting number
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
recording_count	string		<input type="checkbox"/>	Recording count
start_time	datetime		<input type="checkbox"/>	Meeting start time
to	datetime		<input type="checkbox"/>	End Date
topic	string		<input type="checkbox"/>	Meeting topic
total_records	int64		<input type="checkbox"/>	The number of all records available across pages
total_size	string		<input type="checkbox"/>	Total size
uuid	string		<input type="checkbox"/>	Meeting unique ID

### 5.1.9 UsersByUserIdRecordings

List all the recordings

Catalog: Zoom

Schema: CloudRecording

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/recordings`

Insert Zoom API URL: `/users/{userId}/recordings`

Update Zoom API URL: `/users/{userId}/recordings`

Delete Zoom API URL: `/users/{userId}/recordings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /users/{userId}/recordings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdRecordings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a

pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
mc	string	<input type="checkbox"/>	false	Query mc
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
trash	boolean	<input type="checkbox"/>	False	Query trash
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdRecordings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start Date,
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
to	datetime		<input type="checkbox"/>	End Date
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

## 6 Schema: Dashboards

### 6.1 Tables

#### 6.1.1 Metrics\_Crc\_ports\_usageCrc\_ports\_hour\_usageCrc

Retrieve CRC Port UsageGet CRC Port usage hour by hour for a specified time period <aside class='notice'>We will report a maximum of one month. For example, if "from" is set to "2017-08-05" and "to" is "2017-10-10" we will adjust "from" to "2017-09-10"</aside>.

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/crc`

Insert Zoom API URL: `/metrics/crc`

Update Zoom API URL: `/metrics/crc`

Delete Zoom API URL: `/metrics/crc`

Field Selection Method: `NotRequired`

Base Path: `crc_ports_usage[*].crc_ports_hour_usage[*]`

Select Zoom API Operation: `get /metrics/crc`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metrics_Crc_ports_usageCrc_ports_hour_usageCrc`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
to	datetime	<input checked="" type="checkbox"/>		End Date

### Table Function Columns

The columns of the table function `Metrics_Crc_ports_usageCrc_ports_hour_usageCrc` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
date_time	string		<input type="checkbox"/>	Date and time
hour	string		<input type="checkbox"/>	
max_usage	int64		<input type="checkbox"/>	
total_usage	int64		<input type="checkbox"/>	

### 6.1.2 Metrics\_MeetingsMeetings

List meetings List live meetings or past meetings for a specified period

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings`

Insert Zoom API URL: `/metrics/meetings`

Update Zoom API URL: `/metrics/meetings`

Delete Zoom API URL: `/metrics/meetings`

Field Selection Method: NotRequired

Base Path: `meetings[*]`

Select Zoom API Operation: `get /metrics/meetings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric_s_MeetingsMeetings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the cur-

Name	Data Type	Required	Default Value	Documentation
				rent page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, pastOne, live)

## Table Function Columns

The columns of the table function `Metrics_MeetingsMeetings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	string		<input type="checkbox"/>	Meeting duration
email	string		<input type="checkbox"/>	User email
end_time	datetime		<input type="checkbox"/>	Meeting end time
from	datetime		<input type="checkbox"/>	Start date for this report
has_3rd_party_audio	boolean		<input type="checkbox"/>	
has_pstn	boolean		<input type="checkbox"/>	
has_recording	boolean		<input type="checkbox"/>	
has_screen_share	boolean		<input type="checkbox"/>	
has_sip	boolean		<input type="checkbox"/>	
has_video	boolean		<input type="checkbox"/>	
has_voip	boolean		<input type="checkbox"/>	
host	string		<input type="checkbox"/>	User display name
id	int64		<input type="checkbox"/>	Meeting ID
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
participants	int64		<input type="checkbox"/>	Meeting participant count
start_time	datetime		<input type="checkbox"/>	Meeting start time
to	datetime		<input type="checkbox"/>	End date for this report
topic	string		<input type="checkbox"/>	Meeting topic
total_records	int64		<input type="checkbox"/>	The number of all records available across pages



Name	Data Type	Label	Required	Documentation
user_type	string		<input type="checkbox"/>	User type
uuid	guid		<input type="checkbox"/>	Meeting UUID

### 6.1.3 Metrics\_UsersIm

Retrieve IMRetrieve metrics of Zoom IM

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/im`

Insert Zoom API URL: `/metrics/im`

Update Zoom API URL: `/metrics/im`

Delete Zoom API URL: `/metrics/im`

Field Selection Method: `NotRequired`

Base Path: `users[*]`

Select Zoom API Operation: `get /metrics/im`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric-UsersIm`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.

Name	Data Type	Required	Default Value	Documentation
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `Metrics_UsersIm` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
calls_receive	int64		<input type="checkbox"/>	
calls_send	int64		<input type="checkbox"/>	
email	string		<input type="checkbox"/>	User email
emoji_receive	int64		<input type="checkbox"/>	
emoji_send	int64		<input type="checkbox"/>	
files_receive	int64		<input type="checkbox"/>	
files_send	int64		<input type="checkbox"/>	
from	datetime		<input type="checkbox"/>	Start date for this report
group_receive	int64		<input type="checkbox"/>	
group_send	int64		<input type="checkbox"/>	
images_receive	int64		<input type="checkbox"/>	
images_send	int64		<input type="checkbox"/>	
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
to	datetime		<input type="checkbox"/>	End date for this report
total_receive	int64		<input type="checkbox"/>	
total_records	int64		<input type="checkbox"/>	The number of all records available across pages
total_send	int64		<input type="checkbox"/>	
user_id	string		<input type="checkbox"/>	User ID
user_name	string		<input type="checkbox"/>	User display name
videos_receive	int64		<input type="checkbox"/>	
videos_send	int64		<input type="checkbox"/>	
voice_receive	int64		<input type="checkbox"/>	
voice_send	int64		<input type="checkbox"/>	

### 6.1.4 Metrics\_WebinarsWebinars

List webinars List live webinars or past webinars for a specified period

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars`

Insert Zoom API URL: `/metrics/webinars`

Update Zoom API URL: `/metrics/webinars`

Delete Zoom API URL: `/metrics/webinars`

Field Selection Method: `NotRequired`

Base Path: `webinars[*]`

Select Zoom API Operation: `get /metrics/webinars`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric- s_WebinarsWebinars`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>from</code>	<code>datetime</code>	<input checked="" type="checkbox"/>		Start Date
<code>next_page_token</code>	<code>string</code>	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
<code>page_size</code>	<code>int64</code>	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>to</code>	<code>datetime</code>	<input checked="" type="checkbox"/>		End Date

Name	Data Type	Required	Default Value	Documentation
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, pastOne, live)

## Table Function Columns

The columns of the table function `Metrics_WebinarsWebinars` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	string		<input type="checkbox"/>	Webinar duration
email	string		<input type="checkbox"/>	User email
end_time	datetime		<input type="checkbox"/>	Webinar end time
from	datetime		<input type="checkbox"/>	Start date for this report
has_3rd_party_audio	boolean		<input type="checkbox"/>	
has_pstn	boolean		<input type="checkbox"/>	
has_recording	boolean		<input type="checkbox"/>	
has_screen_share	boolean		<input type="checkbox"/>	
has_sip	boolean		<input type="checkbox"/>	
has_video	boolean		<input type="checkbox"/>	
has_voip	boolean		<input type="checkbox"/>	
host	string		<input type="checkbox"/>	User display name
id	int64		<input type="checkbox"/>	Webinar ID
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
participants	int64		<input type="checkbox"/>	Webinar participant count
start_time	datetime		<input type="checkbox"/>	Webinar start time
to	datetime		<input type="checkbox"/>	End date for this report
topic	string		<input type="checkbox"/>	Webinar topic
total_records	int64		<input type="checkbox"/>	The number of all records available across pages
user_type	string		<input type="checkbox"/>	User type
uuid	guid		<input type="checkbox"/>	Webinar UUID

### 6.1.5 MetricsCrc

Retrieve CRC Port UsageGet CRC Port usage hour by hour for a specified time period  
 <aside class='notice'>We will report a maximum of one month. For example, if "from" is set to "2017-08-05" and "to" is "2017-10-10" we will adjust "from" to "2017-09-10"</aside>.

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the In-  
 vative SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/crc`

Insert Zoom API URL: `/metrics/crc`

Update Zoom API URL: `/metrics/crc`

Delete Zoom API URL: `/metrics/crc`

Field Selection Method: `NotRequired`

Base Path: `crc_ports_usage[*]`

Select Zoom API Operation: `get /metrics/crc`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsCrc`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `MetricsCrc` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
date_time	string		<input type="checkbox"/>	Date and time

### 6.1.6 MetricsIm

Retrieve IMRetrieve metrics of Zoom IM

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/im`

Insert Zoom API URL: `/metrics/im`

Update Zoom API URL: `/metrics/im`

Delete Zoom API URL: `/metrics/im`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /metrics/im`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric-slm`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>from</code>	<code>datetime</code>	<input checked="" type="checkbox"/>		Start Date
<code>next_page_token</code>	<code>string</code>	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
<code>page_size</code>	<code>int64</code>	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>to</code>	<code>datetime</code>	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `MetricsIm` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date for this report
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
to	datetime		<input type="checkbox"/>	End date for this report
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 6.1.7 MetricsMeetings

List meetings List live meetings or past meetings for a specified period

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings`

Insert Zoom API URL: `/metrics/meetings`

Update Zoom API URL: `/metrics/meetings`

Delete Zoom API URL: `/metrics/meetings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /metrics/meetings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsMeetings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four

parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, pastOne, live)

## Table Function Columns

The columns of the table function `MetricsMeetings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date for this report
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
to	datetime		<input type="checkbox"/>	End date for this report
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 6.1.8 MetricsMeetingsByMeetingId

Retrieve meeting detail Retrieve live or past meetings detail

Catalog: Zoom



## Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings/{meetingId}`

Insert Zoom API URL: `/metrics/meetings/{meetingId}`

Update Zoom API URL: `/metrics/meetings/{meetingId}`

Delete Zoom API URL: `/metrics/meetings/{meetingId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /metrics/meetings/{meetingId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsMeetingsByMeetingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>meetingId</code>	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
<code>type</code>	string	<input type="checkbox"/>	live	The meeting type (Values: past, pastOne, live)

## Table Function Columns

The columns of the table function `MetricsMeetingsByMeetingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>duration</code>	string		<input type="checkbox"/>	Meeting duration
<code>email</code>	string		<input type="checkbox"/>	User email
<code>end_time</code>	datetime		<input type="checkbox"/>	Meeting end time
<code>has_3rd_party_audio</code>	boolean		<input type="checkbox"/>	
<code>has_pstn</code>	boolean		<input type="checkbox"/>	
<code>has_recording</code>	boolean		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
has_screen_share	boolean		<input type="checkbox"/>	
has_sip	boolean		<input type="checkbox"/>	
has_video	boolean		<input type="checkbox"/>	
has_voip	boolean		<input type="checkbox"/>	
host	string		<input type="checkbox"/>	User display name
id	int64		<input type="checkbox"/>	Meeting ID
participants	int64		<input type="checkbox"/>	Meeting participant count
start_time	datetime		<input type="checkbox"/>	Meeting start time
topic	string		<input type="checkbox"/>	Meeting topic
user_type	string		<input type="checkbox"/>	User type
uuid	guid		<input type="checkbox"/>	Meeting UUID

### 6.1.9 MetricsMeetingsByMeetingIdParticipants

Retrieve meeting participants Retrieve live or past meetings participants

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings/{meetingId}/participants`

Insert Zoom API URL: `/metrics/meetings/{meetingId}/participants`

Update Zoom API URL: `/metrics/meetings/{meetingId}/participants`

Delete Zoom API URL: `/metrics/meetings/{meetingId}/participants`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /metrics/meetings/{meetingId}/participants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric-sMeetingsByMeetingIdParticipants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the

default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, pastOne, live)

## Table Function Columns

The columns of the table function `MetricsMeetingsByMeetingIdParticipants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
connection_type	string		<input type="checkbox"/>	Participant connection type
data_center	string		<input type="checkbox"/>	Participant data center
device	string		<input type="checkbox"/>	Participant device
domain	string		<input type="checkbox"/>	Participant domain
harddisk_id	string		<input type="checkbox"/>	Participant hard disk id
id	guid		<input type="checkbox"/>	Participant UUID
ip_address	string		<input type="checkbox"/>	Participant IP Address
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
location	string		<input type="checkbox"/>	Participant location
mac_addr	string		<input type="checkbox"/>	Participant MAC Address
microphone	string		<input type="checkbox"/>	Participant microphone
network_type	string		<input type="checkbox"/>	Participant network type
pc_name	string		<input type="checkbox"/>	Participant PC name
recording	boolean		<input type="checkbox"/>	Participant record
share_application	boolean		<input type="checkbox"/>	Did participant share application
share_desktop	boolean		<input type="checkbox"/>	Did participant share desktop
share_whiteboard	boolean		<input type="checkbox"/>	Did participant share whiteboard
speaker	string		<input type="checkbox"/>	Participant speaker
user_id	string		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name

Name	Data Type	Label	Required	Documentation
version	string		<input type="checkbox"/>	Participant version

### 6.1.10 MetricsMeetingsByMeetingIdParticipants\_ParticipantsDetailsSharing

Retrieve sharing/recording details of meeting participant Retrieve sharing/recording details of live or past meetings participant

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table NativePlatformScalarRequests to upload data to the Zoom API.

Select Zoom API URL: /metrics/meetings/{meetingId}/participants/sharing

Insert Zoom API URL: /metrics/meetings/{meetingId}/participants/sharing

Update Zoom API URL: /metrics/meetings/{meetingId}/participants/sharing

Delete Zoom API URL: /metrics/meetings/{meetingId}/participants/sharing

Field Selection Method: NotRequired

Base Path: participants[\*].details[\*]

Select Zoom API Operation: get /metrics/meetings/{meetingId}/participants/sharing

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function MetricsMeetingsByMeetingIdParticipants\_ParticipantsDetailsSharing. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be re-

Name	Data Type	Required	Default Value	Documentation
				turned w henever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned w ithin a single API call
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, live)

## Table Function Columns

The columns of the table function `MetricsMeetingsByMeetingIdParticipants_s_ParticipantsDetailsSharing` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
content	string		<input type="checkbox"/>	Type of content shared
end_time	string		<input type="checkbox"/>	End time of sharing
id	string		<input type="checkbox"/>	Participant UUID
start_time	string		<input type="checkbox"/>	Start time of sharing
user_id	string		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name

### 6.1.11 MetricsMeetingsByMeetingIdParticipantsParticipantIdQos

Retrieve meeting participant QOSRetrieve live or past meetings participant quality of service

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings/{meetingId}/participants/{participantId}/qos`

Insert Zoom API URL: `/metrics/meetings/{meetingId}/participants/{participantId}/qos`

Update Zoom API URL: `/metrics/meetings/{meetingId}/participants/{participantId}/qos`

Delete Zoom API URL: `/metrics/meetings/{meetingId}/participants/{participantId}/qos`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /metrics/meetings/{meetingId}/participants/{participantId}/qos`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsMeetingsByMeetingIdParticipantsParticipantIdQos`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
participantId	string	<input checked="" type="checkbox"/>		Participant ID
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, live)

## Table Function Columns

The columns of the table function `MetricsMeetingsByMeetingIdParticipantsParticipantIdQos` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
device	string		<input type="checkbox"/>	Participant device
domain	string		<input type="checkbox"/>	Participant domain
harddisk_id	string		<input type="checkbox"/>	Participant hard disk id
ip_address	string		<input type="checkbox"/>	Participant IP Address
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
location	string		<input type="checkbox"/>	Participant location
mac_addr	string		<input type="checkbox"/>	Participant MAC Address
pc_name	string		<input type="checkbox"/>	Participant PC name
user_id	guid		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name
user_qos_audio_input_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_input_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_input_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_input_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_input_max_loss	string		<input type="checkbox"/>	Max Loss

Name	Data Type	Label	Required	Documentation
user_qos_audio_output_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_output_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_output_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_output_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_output_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_cpu_usage_system_max_cpu_usage	string		<input type="checkbox"/>	System Maximum CPU Usage
user_qos_cpu_usage_zoom_avg_cpu_usage	string		<input type="checkbox"/>	Zoom Average CPU Usage
user_qos_cpu_usage_zoom_max_cpu_usage	string		<input type="checkbox"/>	Zoom Maximum CPU Usage
user_qos_cpu_usage_zoom_min_cpu_usage	string		<input type="checkbox"/>	Zoom Minimum CPU Usage
user_qos_date_time	datetime		<input type="checkbox"/>	Datetime of QOS
version	string		<input type="checkbox"/>	Participant version

### 6.1.12 MetricsMeetingsByMeetingIdParticipantsQos

List meeting participants QOS Retrieve list of live or past meetings participants quality of service

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings/{meetingId}/participants/qos`

Insert Zoom API URL: `/metrics/meetings/{meetingId}/participants/qos`

Update Zoom API URL: `/metrics/meetings/{meetingId}/participants/qos`

Delete Zoom API URL: `/metrics/meetings/{meetingId}/participants/qos`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /metrics/meetings/{meetingId}/participants/qos`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricMeetingsByMeetingIdParticipantsQos`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four

parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	1	Number of items returned per page
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, live)

## Table Function Columns

The columns of the table function `MetricsMeetingsByMeetingIdParticipantsQos` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
device	string		<input type="checkbox"/>	Participant device
domain	string		<input type="checkbox"/>	Participant domain
harddisk_id	string		<input type="checkbox"/>	Participant hard disk id
ip_address	string		<input type="checkbox"/>	Participant IP Address
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
location	string		<input type="checkbox"/>	Participant location
mac_addr	string		<input type="checkbox"/>	Participant MAC Address
pc_name	string		<input type="checkbox"/>	Participant PC name
user_id	guid		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name
user_qos_audio_input_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_input_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_input_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_input_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_input_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_audio_output_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_output_bitrate	string		<input type="checkbox"/>	Bitrate



Name	Data Type	Label	Required	Documentation
user_qos_audio_output_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_output_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_output_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_cpu_usage_system_max_cpu_usage	string		<input type="checkbox"/>	System Maximum CPU Usage
user_qos_cpu_usage_zoom_avg_cpu_usage	string		<input type="checkbox"/>	Zoom Average CPU Usage
user_qos_cpu_usage_zoom_max_cpu_usage	string		<input type="checkbox"/>	Zoom Maximum CPU Usage
user_qos_cpu_usage_zoom_min_cpu_usage	string		<input type="checkbox"/>	Zoom Minimum CPU Usage
user_qos_date_time	datetime		<input type="checkbox"/>	Datetime of QOS
version	string		<input type="checkbox"/>	Participant version

### 6.1.13 MetricsMeetingsByMeetingIdParticipantsSharing

Retrieve sharing/recording details of meeting participant  
Retrieve sharing/recording details of live or past meetings participant

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/meetings/{meetingId}/participants/sharing`

Insert Zoom API URL: `/metrics/meetings/{meetingId}/participants/sharing`

Update Zoom API URL: `/metrics/meetings/{meetingId}/participants/sharing`

Delete Zoom API URL: `/metrics/meetings/{meetingId}/participants/sharing`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /metrics/meetings/{meetingId}/participants/sharing`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric-sMeetingsByMeetingIdParticipantsSharing`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four

parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
type	string	<input type="checkbox"/>	live	The meeting type (Values: past, live)

## Table Function Columns

The columns of the table function `MetricsMeetingsByMeetingIdParticipantSharing` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Participant UUID
user_id	string		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name

### 6.1.14 MetricsWebinars

List webinars List live webinars or past webinars for a specified period

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars`

Insert Zoom API URL: `/metrics/webinars`

Update Zoom API URL: `/metrics/webinars`

Delete Zoom API URL: `/metrics/webinars`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /metrics/webinars`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsWebinars`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, pastOne, live)

## Table Function Columns

The columns of the table function `MetricsWebinars` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date for this report
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page

Name	Data Type	Label	Required	Documentation
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
to	datetime		<input type="checkbox"/>	End date for this report
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 6.1.15 MetricsWebinarsByWebinarId

Retrieve webinar detail Retrieve live or past webinars detail

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars/{webinarId}`

Insert Zoom API URL: `/metrics/webinars/{webinarId}`

Update Zoom API URL: `/metrics/webinars/{webinarId}`

Delete Zoom API URL: `/metrics/webinars/{webinarId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /metrics/webinars/{webinarId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric-sWebinarsByWebinarId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, live)
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `MetricsWebinarsByWebinarId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	string		<input type="checkbox"/>	Webinar duration
email	string		<input type="checkbox"/>	User email
end_time	datetime		<input type="checkbox"/>	Webinar end time
has_3rd_party_audio	boolean		<input type="checkbox"/>	
has_pstn	boolean		<input type="checkbox"/>	
has_recording	boolean		<input type="checkbox"/>	
has_screen_share	boolean		<input type="checkbox"/>	
has_sip	boolean		<input type="checkbox"/>	
has_video	boolean		<input type="checkbox"/>	
has_voip	boolean		<input type="checkbox"/>	
host	string		<input type="checkbox"/>	User display name
id	int64		<input type="checkbox"/>	Webinar ID
participants	int64		<input type="checkbox"/>	Webinar participant count
start_time	datetime		<input type="checkbox"/>	Webinar start time
topic	string		<input type="checkbox"/>	Webinar topic
user_type	string		<input type="checkbox"/>	User type
uuid	guid		<input type="checkbox"/>	Webinar UUID

### 6.1.16 MetricsWebinarsByWebinarIdParticipants

Retrieve webinar participants Retrieve live or past webinar participants

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars/{webinarId}/participants`

Insert Zoom API URL: `/metrics/webinars/{webinarId}/participants`

Update Zoom API URL: `/metrics/webinars/{webinarId}/participants`

Delete Zoom API URL: `/metrics/webinars/{webinarId}/participants`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /metrics/webinars/{webinarId}/participants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsWebinarsByWebinarIdParticipants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, live)
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `MetricsWebinarsByWebinarIdParticipants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
connection_type	string		<input type="checkbox"/>	Participant connection type
data_center	string		<input type="checkbox"/>	Participant data center
device	string		<input type="checkbox"/>	Participant device
domain	string		<input type="checkbox"/>	Participant domain
harddisk_id	string		<input type="checkbox"/>	Participant hard disk id
id	guid		<input type="checkbox"/>	Participant UUID
ip_address	string		<input type="checkbox"/>	Participant IP Address
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
location	string		<input type="checkbox"/>	Participant location
mac_addr	string		<input type="checkbox"/>	Participant MAC Address
microphone	string		<input type="checkbox"/>	Participant microphone

Name	Data Type	Label	Required	Documentation
network_type	string		<input type="checkbox"/>	Participant network type
pc_name	string		<input type="checkbox"/>	Participant PC name
recording	boolean		<input type="checkbox"/>	Participant record
share_application	boolean		<input type="checkbox"/>	Did participant share application
share_desktop	boolean		<input type="checkbox"/>	Did participant share desktop
share_whiteboard	boolean		<input type="checkbox"/>	Did participant share whiteboard
speaker	string		<input type="checkbox"/>	Participant speaker
user_id	string		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name
version	string		<input type="checkbox"/>	Participant version

### 6.1.17 MetricsWebinarsByWebinarIdParticipants\_ParticipantsDetailsSharing

Retrieve sharing/recording details of webinar participant Retrieve sharing/recording details of live or past webinar participant

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Insert Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Update Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Delete Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Field Selection Method: NotRequired

Base Path: `participants[*].details[*]`

Select Zoom API Operation: `get /metrics/webinars/{webinarId}/participants/sharing`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Metric-sWebinarsByWebinarIdParticipants_ParticipantsDetailsSharing`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, live)
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `MetricsWebinarsByWebinarIdParticipants_s_ParticipantsDetailsSharing` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
content	string		<input type="checkbox"/>	Type of content shared
end_time	string		<input type="checkbox"/>	End time of sharing
id	string		<input type="checkbox"/>	Participant UUID
start_time	string		<input type="checkbox"/>	Start time of sharing
user_id	string		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name

### 6.1.18 MetricsWebinarsByWebinarIdParticipantsParticipantIdQos

Retrieve webinar participant QOS Retrieve live or past webinar participant quality of service

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars/{webinarId}/participants/{participantId}/qos`

Insert Zoom API URL: `/metrics/webinars/{webinarId}/participants/{participantId}/qos`



Update Zoom API URL: `/metrics/webinars/{webinarId}/participants/{participantId}/qos`

Delete Zoom API URL: `/metrics/webinars/{webinarId}/participants/{participantId}/qos`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /metrics/webinars/{webinarId}/participants/{participantId}/qos`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsWebinarsByWebinarIdParticipantsParticipantIdQos`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
participantId	string	<input checked="" type="checkbox"/>		Participant ID
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, live)
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `MetricsWebinarsByWebinarIdParticipantsParticipantIdQos` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
device	string		<input type="checkbox"/>	Participant device
domain	string		<input type="checkbox"/>	Participant domain
harddisk_id	string		<input type="checkbox"/>	Participant hard disk id
ip_address	string		<input type="checkbox"/>	Participant IP Address
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
location	string		<input type="checkbox"/>	Participant location
mac_addr	string		<input type="checkbox"/>	Participant MAC Address
pc_name	string		<input type="checkbox"/>	Participant PC name

Name	Data Type	Label	Required	Documentation
user_id	guid		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name
user_qos_audio_input_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_input_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_input_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_input_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_input_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_audio_output_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_output_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_output_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_output_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_output_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_cpu_usage_system_max_cpu_usage	string		<input type="checkbox"/>	System Maximum CPU Usage
user_qos_cpu_usage_zoom_avg_cpu_usage	string		<input type="checkbox"/>	Zoom Average CPU Usage
user_qos_cpu_usage_zoom_max_cpu_usage	string		<input type="checkbox"/>	Zoom Maximum CPU Usage
user_qos_cpu_usage_zoom_min_cpu_usage	string		<input type="checkbox"/>	Zoom Minimum CPU Usage
user_qos_date_time	datetime		<input type="checkbox"/>	Datetime of QOS
version	string		<input type="checkbox"/>	Participant version

### 6.1.19 MetricsWebinarsByWebinarIdParticipantsQos

List webinar participant QOS Retrieve list of live or past webinar participants quality of service

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars/{webinarId}/participants/qos`

Insert Zoom API URL: `/metrics/webinars/{webinarId}/participants/qos`

Update Zoom API URL: `/metrics/webinars/{webinarId}/participants/qos`

Delete Zoom API URL: `/metrics/webinars/{webinarId}/participants/qos`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /metrics/webinars/{webinarId}/participants/qos`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsWebinarsByWebinarIdParticipantsQos`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	1	Number of items returned per page
type	string	<input type="checkbox"/>	live	The webinar type (Values: past, live)
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `MetricsWebinarsByWebinarIdParticipantsQos` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
device	string		<input type="checkbox"/>	Participant device
domain	string		<input type="checkbox"/>	Participant domain
harddisk_id	string		<input type="checkbox"/>	Participant hard disk id
ip_address	string		<input type="checkbox"/>	Participant IP Address
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
location	string		<input type="checkbox"/>	Participant location
mac_addr	string		<input type="checkbox"/>	Participant MAC Address
pc_name	string		<input type="checkbox"/>	Participant PC name
user_id	guid		<input type="checkbox"/>	Participant ID
user_name	string		<input type="checkbox"/>	Participant display name
user_qos_audio_input_avg_loss	string		<input type="checkbox"/>	Average Loss

Name	Data Type	Label	Required	Documentation
user_qos_audio_input_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_input_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_input_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_input_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_audio_output_avg_loss	string		<input type="checkbox"/>	Average Loss
user_qos_audio_output_bitrate	string		<input type="checkbox"/>	Bitrate
user_qos_audio_output_jitter	string		<input type="checkbox"/>	Jitter
user_qos_audio_output_latency	string		<input type="checkbox"/>	Latency
user_qos_audio_output_max_loss	string		<input type="checkbox"/>	Max Loss
user_qos_cpu_usage_system_max_cpu_usage	string		<input type="checkbox"/>	System Maximum CPU Usage
user_qos_cpu_usage_zoom_avg_cpu_usage	string		<input type="checkbox"/>	Zoom Average CPU Usage
user_qos_cpu_usage_zoom_max_cpu_usage	string		<input type="checkbox"/>	Zoom Maximum CPU Usage
user_qos_cpu_usage_zoom_min_cpu_usage	string		<input type="checkbox"/>	Zoom Minimum CPU Usage
user_qos_date_time	datetime		<input type="checkbox"/>	Datetime of QOS
version	string		<input type="checkbox"/>	Participant version

### 6.1.20 MetricsWebinarsByWebinarIdParticipantsSharing

Retrieve sharing/recording details of webinar participant  
Retrieve sharing/recording details of live or past webinar participant

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Insert Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Update Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Delete Zoom API URL: `/metrics/webinars/{webinarId}/participants/sharing`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /metrics/webinars/{webinarId}/participants/sharing`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsWebinarsByWebinarIdParticipantsSharing`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>next_page_token</code>	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
<code>page_size</code>	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>type</code>	string	<input type="checkbox"/>	live	The webinar type (Values: past, live)
<code>webinarId</code>	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `MetricsWebinarsByWebinarIdParticipantSharing` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>id</code>	string		<input type="checkbox"/>	Participant UUID
<code>user_id</code>	string		<input type="checkbox"/>	Participant ID
<code>user_name</code>	string		<input type="checkbox"/>	Participant display name

### 6.1.21 MetricsZoomrooms

List Zoom Rooms List all zoom rooms on account

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/zoomrooms`

Insert Zoom API URL: `/metrics/zoomrooms`

Update Zoom API URL: `/metrics/zoomrooms`

Delete Zoom API URL: `/metrics/zoomrooms`

Field Selection Method: `NotRequired`

Base Path: `zoom_rooms[*]`

Select Zoom API Operation: `get /metrics/zoomrooms`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricsZoomrooms`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>page_number</code>	<code>int64</code>	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	<code>int64</code>	<input type="checkbox"/>	30	The number of records returned within a single API call

## Table Function Columns

The columns of the table function `MetricsZoomrooms` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>account_type</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room email type
<code>calender_name</code>	<code>string</code>		<input type="checkbox"/>	Zoom Calendar name
<code>camera</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room camera
<code>device_ip</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room device IP
<code>email</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room email
<code>id</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room ID
<code>last_start_time</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room last start time
<code>microphone</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room microphone
<code>room_name</code>	<code>string</code>		<input type="checkbox"/>	Zoom Room name

Name	Data Type	Label	Required	Documentation
speaker	string		<input type="checkbox"/>	Zoom Room speaker
status	string		<input type="checkbox"/>	Zoom Room status

### 6.1.22 MetricsZoomrooms\_Past\_meetingsMeetingsByZoomroomId

Retrieve Zoom RoomRetrieve zoom room on account

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the In-variantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Insert Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Update Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Delete Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Field Selection Method: NotRequired

Base Path: `past_meetings.meetings[*]`

Select Zoom API Operation: `get /metrics/zoomrooms/{zoomroomId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricZoomrooms_Past_meetingsMeetingsByZoomroomId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
zoomroomId	string	<input checked="" type="checkbox"/>		The Zoom Room ID

## Table Function Columns

The columns of the table function `MetricsZoomrooms_Past_meetingsMeeting-sByZoomroomId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
account_type	string		<input type="checkbox"/>	Zoom Room email type
calender_name	string		<input type="checkbox"/>	Zoom Calendar name
camera	string		<input type="checkbox"/>	Zoom Room camera
device_ip	string		<input type="checkbox"/>	Zoom Room device IP
duration	string		<input type="checkbox"/>	Meeting duration
email_1	string		<input type="checkbox"/>	User email
email	string		<input type="checkbox"/>	Zoom Room email
end_time	datetime		<input type="checkbox"/>	Meeting end time
from	datetime		<input type="checkbox"/>	Start date for this report
has_3rd_party_audio	boolean		<input type="checkbox"/>	
has_pstn	boolean		<input type="checkbox"/>	
has_recording	boolean		<input type="checkbox"/>	
has_screen_share	boolean		<input type="checkbox"/>	
has_sip	boolean		<input type="checkbox"/>	
has_video	boolean		<input type="checkbox"/>	
has_voip	boolean		<input type="checkbox"/>	
host	string		<input type="checkbox"/>	User display name
id_1	int64		<input type="checkbox"/>	Meeting ID
id	string		<input type="checkbox"/>	Zoom Room ID
last_start_time	string		<input type="checkbox"/>	Zoom Room last start time
live_meeting_duration	string		<input type="checkbox"/>	Meeting duration
live_meeting_email	string		<input type="checkbox"/>	User email
live_meeting_end_time	datetime		<input type="checkbox"/>	Meeting end time
live_meeting_has_3rd_party_audio	boolean		<input type="checkbox"/>	
live_meeting_has_pstn	boolean		<input type="checkbox"/>	
live_meeting_has_recording	boolean		<input type="checkbox"/>	
live_meeting_has_screen_share	boolean		<input type="checkbox"/>	
live_meeting_has_sip	boolean		<input type="checkbox"/>	
live_meeting_has_video	boolean		<input type="checkbox"/>	
live_meeting_has_voip	boolean		<input type="checkbox"/>	
live_meeting_host	string		<input type="checkbox"/>	User display name
live_meeting_id	int64		<input type="checkbox"/>	Meeting ID
live_meeting_participants	int64		<input type="checkbox"/>	Meeting participant count
live_meeting_start_time	datetime		<input type="checkbox"/>	Meeting start time
live_meeting_topic	string		<input type="checkbox"/>	Meeting topic
live_meeting_user_type	string		<input type="checkbox"/>	User type



Name	Data Type	Label	Required	Documentation
live_meeting_uuid	guid		<input type="checkbox"/>	Meeting UUID
microphone	string		<input type="checkbox"/>	Zoom Room microphone
participants	int64		<input type="checkbox"/>	Meeting participant count
room_name	string		<input type="checkbox"/>	Zoom Room name
speaker	string		<input type="checkbox"/>	Zoom Room speaker
start_time	datetime		<input type="checkbox"/>	Meeting start time
status	string		<input type="checkbox"/>	Zoom Room status
to	datetime		<input type="checkbox"/>	End date for this report
topic	string		<input type="checkbox"/>	Meeting topic
user_type	string		<input type="checkbox"/>	User type
uuid	guid		<input type="checkbox"/>	Meeting UUID

### 6.1.23 MetricsZoomroomsByZoomroomId

Retrieve Zoom Room Retrieve zoom room on account

Catalog: Zoom

Schema: Dashboards

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Insert Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Update Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Delete Zoom API URL: `/metrics/zoomrooms/{zoomroomId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /metrics/zoomrooms/{zoomroomId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MetricZoomroomsByZoomroomId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
zoomroomld	string	<input checked="" type="checkbox"/>		The Zoom Room ID

## Table Function Columns

The columns of the table function `MetricsZoomroomsByZoomroomId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
account_type	string		<input type="checkbox"/>	Zoom Room email type
calender_name	string		<input type="checkbox"/>	Zoom Calendar name
camera	string		<input type="checkbox"/>	Zoom Room camera
device_ip	string		<input type="checkbox"/>	Zoom Room device IP
email	string		<input type="checkbox"/>	Zoom Room email
id	string		<input type="checkbox"/>	Zoom Room ID
last_start_time	string		<input type="checkbox"/>	Zoom Room last start time
live_meeting_duration	string		<input type="checkbox"/>	Meeting duration
live_meeting_email	string		<input type="checkbox"/>	User email
live_meeting_end_time	datetime		<input type="checkbox"/>	Meeting end time
live_meeting_has_3rd_party_audio	boolean		<input type="checkbox"/>	
live_meeting_has_pstn	boolean		<input type="checkbox"/>	
live_meeting_has_recording	boolean		<input type="checkbox"/>	
live_meeting_has_screen_share	boolean		<input type="checkbox"/>	
live_meeting_has_sip	boolean		<input type="checkbox"/>	
live_meeting_has_video	boolean		<input type="checkbox"/>	
live_meeting_has_voip	boolean		<input type="checkbox"/>	
live_meeting_host	string		<input type="checkbox"/>	User display name
live_meeting_id	int64		<input type="checkbox"/>	Meeting ID
live_meeting_participants	int64		<input type="checkbox"/>	Meeting participant count
live_meeting_start_time	datetime		<input type="checkbox"/>	Meeting start time
live_meeting_topic	string		<input type="checkbox"/>	Meeting topic
live_meeting_user_type	string		<input type="checkbox"/>	User type
live_meeting_uuid	guid		<input type="checkbox"/>	Meeting UUID
microphone	string		<input type="checkbox"/>	Zoom Room microphone
room_name	string		<input type="checkbox"/>	Zoom Room name
speaker	string		<input type="checkbox"/>	Zoom Room speaker
status	string		<input type="checkbox"/>	Zoom Room status

## 7 Schema: Devices

### 7.1 Tables

#### 7.1.1 deleteH323DevicesByDeviceId

Delete a H.323/SIP DeviceDelete a H.323/SIP Device on your Zoom account

Catalog: Zoom

Schema: Devices

This is a read-only table function. The Zoom API may not support changing the data or the In-variantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/h323/devices/{deviceId}`

Insert Zoom API URL: `/h323/devices/{deviceId}`

Update Zoom API URL: `/h323/devices/{deviceId}`

Delete Zoom API URL: `/h323/devices/{deviceId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /h323/devices/{deviceId}`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteH323DevicesByDeviceId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>deviceId</code>	string	<input checked="" type="checkbox"/>		The device ID

### Table Function Columns

The columns of the table function `deleteH323DevicesByDeviceId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 7.1.2 H323Devices

List H.323/SIP Devices. List H.323/SIP Devices on your Zoom account.

Catalog: Zoom

Schema: Devices

This is a read-only table. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/h323/devices`

Insert Zoom API URL: `/h323/devices`

Update Zoom API URL: `/h323/devices`

Delete Zoom API URL: `/h323/devices`

Field Selection Method: `NotRequired`

Base Path: `devices[*]`

Select Zoom API Operation: `get /h323/devices`

## Table Columns

The columns of the table `H323Devices` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
encryption	string		<input checked="" type="checkbox"/>	Device encryption
id	string		<input type="checkbox"/>	Device ID
ip	string		<input checked="" type="checkbox"/>	Device Ip
name	string(64)		<input checked="" type="checkbox"/>	Device name
protocol	string		<input checked="" type="checkbox"/>	Device protocol

### 7.1.3 patchH323DevicesByDeviceId

Update a H.323/SIP Device. Update a H.323/SIP Device on your Zoom account

Catalog: Zoom

Schema: Devices

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/h323/devices/{deviceId}`

Insert Zoom API URL: `/h323/devices/{deviceId}`

Update Zoom API URL: `/h323/devices/{deviceId}`

Delete Zoom API URL: `/h323/devices/{deviceId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /h323/devices/{deviceId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchH323DevicesByDeviceId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
deviceId	string	<input checked="" type="checkbox"/>		The device ID

## Table Function Columns

The columns of the table function `patchH323DevicesByDeviceId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 7.1.4 postH323Devices

Create a H.323/SIP DeviceCreate a H.323/SIP Device on your Zoom account

Catalog: Zoom

Schema: Devices

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/h323/devices`

Insert Zoom API URL: `/h323/devices`

Update Zoom API URL: `/h323/devices`

Delete Zoom API URL: `/h323/devices`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /h323/devices`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postH323Devices`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		H.323/SIP Device

## Table Function Columns

The columns of the table function `postH323Devices` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
encryption	string		<input checked="" type="checkbox"/>	Device encryption
id	string		<input type="checkbox"/>	Device ID
ip	string		<input checked="" type="checkbox"/>	Device Ip
name	string(64)		<input checked="" type="checkbox"/>	Device name
protocol	string		<input checked="" type="checkbox"/>	Device protocol
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

## 8 Schema: Groups

### 8.1 Tables

#### 8.1.1 `deleteGroupsByGroupId`

Delete a group Delete a group under your account

Catalog: Zoom

Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups/{groupId}`

Insert Zoom API URL: `/groups/{groupId}`

Update Zoom API URL: `/groups/{groupId}`

Delete Zoom API URL: `/groups/{groupId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /groups/{groupId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteGroupsByGroupId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>groupId</code>	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `deleteGroupsByGroupId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 8.1.2 `deleteGroupsByGroupIdMembersMemberId`

Delete a group member>Delete a member from a group under your account

Catalog: Zoom

Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups/{groupId}/members/{memberId}`

Insert Zoom API URL: `/groups/{groupId}/members/{memberId}`

Update Zoom API URL: `/groups/{groupId}/members/{memberId}`

Delete Zoom API URL: `/groups/{groupId}/members/{memberId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /groups/{groupId}/members/{memberId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteGroupsByGroupIdMembersMemberId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
groupId	string	<input checked="" type="checkbox"/>		The group ID
memberId	string	<input checked="" type="checkbox"/>		The member ID

## Table Function Columns

The columns of the table function `deleteGroupsByGroupIdMembersMemberId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 8.1.3 Groups

List groupsList groups under your account

Catalog: Zoom

Schema: Groups

This is a read-only table. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups`

Insert Zoom API URL: `/groups`

Update Zoom API URL: `/groups`



Delete Zoom API URL: /groups

Field Selection Method: NotRequired

Select Zoom API Operation: get /groups

## Table Columns

The columns of the table `Groups` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
total_records	int64		<input type="checkbox"/>	Total records

### 8.1.4 Groups\_Groups

List groupsList groups under your account

Catalog: Zoom

Schema: Groups

This is a read-only table. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: /groups

Insert Zoom API URL: /groups

Update Zoom API URL: /groups

Delete Zoom API URL: /groups

Field Selection Method: NotRequired

Base Path: groups [\*]

Select Zoom API Operation: get /groups

## Table Columns

The columns of the table `Groups_Groups` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Group ID
name	string		<input type="checkbox"/>	Group name
total_members	int64		<input type="checkbox"/>	Total number of members in this group
total_records	int64		<input type="checkbox"/>	Total records

### 8.1.5 GroupsByGroupId

Retrieve a groupRetrieve a group under your account

Catalog: Zoom

Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups/{groupId}`

Insert Zoom API URL: `/groups/{groupId}`

Update Zoom API URL: `/groups/{groupId}`

Delete Zoom API URL: `/groups/{groupId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /groups/{groupId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `GroupsByGroupId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>groupId</code>	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `GroupsByGroupId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>id</code>	string		<input type="checkbox"/>	Group ID
<code>name</code>	string		<input type="checkbox"/>	Group name
<code>total_members</code>	int64		<input type="checkbox"/>	Total number of members in this group

### 8.1.6 GroupsByGroupIdMembers

List a group's membersList a group's members under your account

Catalog: Zoom

## Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups/{groupId}/members`

Insert Zoom API URL: `/groups/{groupId}/members`

Update Zoom API URL: `/groups/{groupId}/members`

Delete Zoom API URL: `/groups/{groupId}/members`

Field Selection Method: `NotRequired`

Base Path: `members[*]`

Select Zoom API Operation: `get /groups/{groupId}/members`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `GroupsByGroupIdMembers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>groupId</code>	string	<input checked="" type="checkbox"/>		The group ID
<code>page_number</code>	int64	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	int64	<input type="checkbox"/>	30	The number of records returned within a single API call

## Table Function Columns

The columns of the table function `GroupsByGroupIdMembers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>email</code>	string		<input type="checkbox"/>	User email
<code>first_name</code>	string		<input type="checkbox"/>	User first name
<code>id</code>	string		<input type="checkbox"/>	User ID
<code>last_name</code>	string		<input type="checkbox"/>	User last name

Name	Data Type	Label	Required	Documentation
type	int64		<input type="checkbox"/>	User type

### 8.1.7 patchGroupsByGroupId

Update a groupUpdate a group under your account

Catalog: Zoom

Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups/{groupId}`

Insert Zoom API URL: `/groups/{groupId}`

Update Zoom API URL: `/groups/{groupId}`

Delete Zoom API URL: `/groups/{groupId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /groups/{groupId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchGroupsByGroupId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
groupId	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `patchGroupsByGroupId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 8.1.8 postGroups

Create a groupCreate a group under your account

Catalog: Zoom

Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups`

Insert Zoom API URL: `/groups`

Update Zoom API URL: `/groups`

Delete Zoom API URL: `/groups`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /groups`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postGroups`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `postGroups` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Group ID
name	string		<input type="checkbox"/>	Group name
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
total_members	int64		<input type="checkbox"/>	Group member count

### 8.1.9 postGroupsByGroupIdMembers

Add group members Add members to a group under your account

Catalog: Zoom

Schema: Groups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/groups/{groupId}/members`

Insert Zoom API URL: `/groups/{groupId}/members`

Update Zoom API URL: `/groups/{groupId}/members`

Delete Zoom API URL: `/groups/{groupId}/members`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /groups/{groupId}/members`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postGroupsByGroupIdMembers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
groupId	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `postGroupsByGroupIdMembers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>added_at</code>	<code>datetime</code>		<input type="checkbox"/>	
<code>ids</code>	<code>string</code>		<input type="checkbox"/>	
RESULT	<code>string</code>		<input type="checkbox"/>	Outcome of operation as single plain text column.

## 9 Schema: IMChat

### 9.1 Tables

#### 9.1.1 ImChat\_SessionsSessions

Retrieve IM Chat sessions Retrieve IM Chat sessions for a specified period <aside>This API only supports oauth2.</aside>

Catalog: Zoom

Schema: IMChat

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/chat/sessions`

Insert Zoom API URL: `/im/chat/sessions`

Update Zoom API URL: `/im/chat/sessions`

Delete Zoom API URL: `/im/chat/sessions`

Field Selection Method: NotRequired

Base Path: `sessions[*]`

Select Zoom API Operation: `get /im/chat/sessions`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ImChat_SessionsSessions`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `ImChat_SessionsSessions` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date
last_message_sent_time	datetime		<input type="checkbox"/>	Last message sent time
name	string		<input type="checkbox"/>	Meeting topic
next_page_token	string		<input type="checkbox"/>	Next page token, used to paginate through large result sets. A next page token will be returned whenever the set of available result list exceeds page size. The expiration period is 15 minutes.
page_size	int64		<input type="checkbox"/>	The amount of records returns within a single API call.
session_id	string		<input type="checkbox"/>	IM Chat session ID
to	datetime		<input type="checkbox"/>	End date
type	string		<input type="checkbox"/>	IM Chat session type

### 9.1.2 ImChatSessions

Retrieve IM Chat sessions Retrieve IM Chat sessions for a specified period <aside>This API only supports oauth2.</aside>

Catalog: Zoom

Schema: IMChat

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/chat/sessions`

Insert Zoom API URL: `/im/chat/sessions`



Update Zoom API URL: `/im/chat/sessions`

Delete Zoom API URL: `/im/chat/sessions`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /im/chat/sessions`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ImChatSessions`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `ImChatSessions` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date
next_page_token	string		<input type="checkbox"/>	Next page token, used to paginate through large result sets. A next page token will be returned whenever the set of available result list exceeds page size. The expiration period is 15 minutes.
page_size	int64		<input type="checkbox"/>	The amount of records returns within a single API call.

Name	Data Type	Label	Required	Documentation
to	datetime		<input type="checkbox"/>	End date

### 9.1.3 ImChatSessions\_MessagesBySessionId

Retrieve IM Chat messages Retrieve IM Chat messages for a specified period <aside>This API only supports oauth2.</aside>

Catalog: Zoom

Schema: IMChat

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table NativePlatformScalarRequests to upload data to the Zoom API.

Select Zoom API URL: /im/chat/sessions/{sessionId}

Insert Zoom API URL: /im/chat/sessions/{sessionId}

Update Zoom API URL: /im/chat/sessions/{sessionId}

Delete Zoom API URL: /im/chat/sessions/{sessionId}

Field Selection Method: NotRequired

Base Path: messages [\*]

Select Zoom API Operation: get /im/chat/sessions/{sessionId}

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function ImChatSessions\_MessagesBySessionId. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.

Name	Data Type	Required	Default Value	Documentation
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
sessionId	string	<input checked="" type="checkbox"/>		IM Chat Session ID
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `ImChatSessions_MessagesBySessionId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
action_time	datetime		<input type="checkbox"/>	Action time
action	string		<input type="checkbox"/>	IM Chat message action
date_time	datetime		<input type="checkbox"/>	IM Chat message sent time
from	datetime		<input type="checkbox"/>	Start date
message	string		<input type="checkbox"/>	IM Chat message content
next_page_token	string		<input type="checkbox"/>	Next page token, used to paginate through large result sets. A next page token will be returned whenever the set of available result list exceeds page size. The expiration period is 15 minutes.
page_size	int64		<input type="checkbox"/>	The amount of records returns within a single API call.
sender	string		<input type="checkbox"/>	IM Chat message sender
session_id	string		<input type="checkbox"/>	IM Chat session ID
to	datetime		<input type="checkbox"/>	End date

### 9.1.4 ImChatSessionsBySessionId

Retrieve IM Chat messages Retrieve IM Chat messages for a specified period <aside>This API only supports oauth2.</aside>

Catalog: Zoom

Schema: IMChat

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/chat/sessions/{sessionId}`

Insert Zoom API URL: `/im/chat/sessions/{sessionId}`

Update Zoom API URL: `/im/chat/sessions/{sessionId}`

Delete Zoom API URL: `/im/chat/sessions/{sessionId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /im/chat/sessions/{sessionId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ImChatSessionsBySessionId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
sessionId	string	<input checked="" type="checkbox"/>		IM Chat Session ID
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `ImChatSessionsBySessionId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date
next_page_token	string		<input type="checkbox"/>	Next page token, used to paginate through large result sets. A next page token will be returned whenever the set of available result list exceeds page size. The expiration period is 15 minutes.
page_size	int64		<input type="checkbox"/>	The amount of records returns within a single API call.
session_id	string		<input type="checkbox"/>	IM Chat session ID
to	datetime		<input type="checkbox"/>	End date

## 10 Schema: IMGroups

### 10.1 Tables

#### 10.1.1 deleteImGroupsByGroupId

Delete an IM GroupDelete an IM Group under your account

Catalog: Zoom

Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups/{groupId}`

Insert Zoom API URL: `/im/groups/{groupId}`

Update Zoom API URL: `/im/groups/{groupId}`

Delete Zoom API URL: `/im/groups/{groupId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /im/groups/{groupId}`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteImGroupsByGroupId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>groupId</code>	string	<input checked="" type="checkbox"/>		The group ID

### Table Function Columns

The columns of the table function `deleteImGroupsByGroupId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 10.1.2 deleteImGroupsByGroupIdMembersMemberId

Delete an IM Group member Delete a member from an IM Group under your account

Catalog: Zoom

Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups/{groupId}/members/{memberId}`

Insert Zoom API URL: `/im/groups/{groupId}/members/{memberId}`

Update Zoom API URL: `/im/groups/{groupId}/members/{memberId}`

Delete Zoom API URL: `/im/groups/{groupId}/members/{memberId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `delete /im/groups/{groupId}/members/{memberId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteImGroupsByGroupIdMembersMemberId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>groupId</code>	string	<input checked="" type="checkbox"/>		The group ID
<code>memberId</code>	string	<input checked="" type="checkbox"/>		The member ID

## Table Function Columns

The columns of the table function `deleteImGroupsByGroupIdMembersMemberId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 10.1.3 ImGroups

List IM Groups List IM groups under your account

Catalog: Zoom

Schema: IMGroups

This is a read-only table. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups`

Insert Zoom API URL: `/im/groups`

Update Zoom API URL: `/im/groups`

Delete Zoom API URL: `/im/groups`

Field Selection Method: NotRequired

Base Path: `groups[*]`

Select Zoom API Operation: `get /im/groups`

## Table Columns

The columns of the table `ImGroups` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>id</code>	string		<input type="checkbox"/>	IM Group ID
<code>name</code>	string		<input type="checkbox"/>	Group name
<code>search_by_account</code>	boolean		<input type="checkbox"/>	Members can search others under same account
<code>search_by_domain</code>	boolean		<input type="checkbox"/>	Members can search others in the same email domain
<code>search_by_ma_account</code>	boolean		<input type="checkbox"/>	Members can search others under same master account, including all sub accounts
<code>total_members</code>	int64		<input type="checkbox"/>	Total number of members in this group
<code>type</code>	string		<input type="checkbox"/>	IM Group type

### 10.1.4 ImGroupsByGroupId

Retrieve an IM Group Retrieve an IM Group under your account

Catalog: Zoom

Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups/{groupId}`

Insert Zoom API URL: `/im/groups/{groupId}`

Update Zoom API URL: `/im/groups/{groupId}`

Delete Zoom API URL: `/im/groups/{groupId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /im/groups/{groupId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ImGroupsByGroupId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
groupId	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `ImGroupsByGroupId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Group ID
name	string		<input type="checkbox"/>	Group name
search_by_account	boolean		<input type="checkbox"/>	Members can search others under same account
search_by_domain	boolean		<input type="checkbox"/>	Members can search others in the same email domain
search_by_ma_account	boolean		<input type="checkbox"/>	Members can search others under same master account, including all sub accounts
total_members	int64		<input type="checkbox"/>	Total number of members in this group
type	string		<input type="checkbox"/>	IM Group type

### 10.1.5 ImGroupsByGroupIdMembers

List an IM Group's membersList an IM Group's members under your account

Catalog: Zoom



## Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups/{groupId}/members`

Insert Zoom API URL: `/im/groups/{groupId}/members`

Update Zoom API URL: `/im/groups/{groupId}/members`

Delete Zoom API URL: `/im/groups/{groupId}/members`

Field Selection Method: `NotRequired`

Base Path: `members[*]`

Select Zoom API Operation: `get /im/groups/{groupId}/members`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ImGroupsByGroupIdMembers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>groupId</code>	string	<input checked="" type="checkbox"/>		The group ID
<code>page_number</code>	int64	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	int64	<input type="checkbox"/>	30	The number of records returned within a single API call

## Table Function Columns

The columns of the table function `ImGroupsByGroupIdMembers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>email</code>	string		<input type="checkbox"/>	User email
<code>first_name</code>	string		<input type="checkbox"/>	User first name
<code>id</code>	string		<input type="checkbox"/>	User ID
<code>last_name</code>	string		<input type="checkbox"/>	User last name

Name	Data Type	Label	Required	Documentation
type	int64		<input type="checkbox"/>	User type

### 10.1.6 patchImGroupsByGroupId

Update an IM Group

Catalog: Zoom

Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups/{groupId}`

Insert Zoom API URL: `/im/groups/{groupId}`

Update Zoom API URL: `/im/groups/{groupId}`

Delete Zoom API URL: `/im/groups/{groupId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /im/groups/{groupId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchImGroupsByGroupId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
groupId	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `patchImGroupsByGroupId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 10.1.7 postImGroups

Create an IM GroupCreate a IM Group under your account

Catalog: Zoom

Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups`

Insert Zoom API URL: `/im/groups`

Update Zoom API URL: `/im/groups`

Delete Zoom API URL: `/im/groups`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /im/groups`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postImGroups`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `postImGroups` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Group ID
name	string		<input type="checkbox"/>	Group name
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
search_by_account	boolean		<input type="checkbox"/>	Members can search others under same account
search_by_domain	boolean		<input type="checkbox"/>	Members can search others in the same email domain
search_by_ma_account	boolean		<input type="checkbox"/>	Members can search others under same master account, including all sub accounts
total_members	int64		<input type="checkbox"/>	Group member count

### 10.1.8 postImGroupsByGroupIdMembers

Add IM Group members Add members to an IM Group under your account

Catalog: Zoom

Schema: IMGroups

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/im/groups/{groupId}/members`

Insert Zoom API URL: `/im/groups/{groupId}/members`

Update Zoom API URL: `/im/groups/{groupId}/members`

Delete Zoom API URL: `/im/groups/{groupId}/members`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /im/groups/{groupId}/members`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postImGroupsByGroupIdMembers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
groupid	string	<input checked="" type="checkbox"/>		The group ID

## Table Function Columns

The columns of the table function `postImGroupsByGroupIdMembers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
added_at	datetime		<input type="checkbox"/>	
ids	string		<input type="checkbox"/>	
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

## 11 Schema: Meetings

### 11.1 Tables

#### 11.1.1 deleteMeetingsByMeetingId

Delete a meetingDelete a meeting

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}`

Insert Zoom API URL: `/meetings/{meetingId}`

Update Zoom API URL: `/meetings/{meetingId}`

Delete Zoom API URL: `/meetings/{meetingId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /meetings/{meetingId}`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteMeetingsByMeetingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
occurrence_id	string	<input type="checkbox"/>		The meeting occurrence ID

## Table Function Columns

The columns of the table function `deleteMeetingsByMeetingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.2 deleteMeetingsByMeetingIdPollsPollId

Delete a meeting's Poll>Delete a meeting's Poll

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the In-variantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Insert Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Update Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Delete Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /meetings/{meetingId}/polls/{pollId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteMeetingsByMeetingIdPollsPollId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
pollId	string	<input checked="" type="checkbox"/>		The poll ID

## Table Function Columns

The columns of the table function `deleteMeetingsByMeetingIdPollsPollId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.3 MeetingsByMeetingId

Retrieve a meeting's details

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}`

Insert Zoom API URL: `/meetings/{meetingId}`

Update Zoom API URL: `/meetings/{meetingId}`

Delete Zoom API URL: `/meetings/{meetingId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /meetings/{meetingId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `MeetingsByMeetingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
agenda	string		<input type="checkbox"/>	Agenda
created_at	datetime		<input type="checkbox"/>	Create time
duration	int64		<input type="checkbox"/>	Meeting duration
h323_password	string		<input type="checkbox"/>	H.323/SIP room system password
host_id	string		<input type="checkbox"/>	ID of the user set as host of meeting
id	string		<input type="checkbox"/>	Meeting ID, also known as meeting number
join_url	string		<input type="checkbox"/>	Join url
password	string		<input type="checkbox"/>	Meeting password
settings_alternative_hosts	string		<input type="checkbox"/>	Alternative hosts emails or IDs. Multiple value separated by comma.
settings_approval_type	int64		<input type="checkbox"/>	
settings_audio	string		<input type="checkbox"/>	Determine how participants can join the audio portion of the meeting
settings_auto_recording	string		<input type="checkbox"/>	
settings_close_registration	boolean		<input type="checkbox"/>	Close registration after event date
settings_cn_meeting	boolean		<input type="checkbox"/>	Host meeting in China
settings_enforce_login_domains	string		<input type="checkbox"/>	Only signed-in users with specified domains can join meetings
settings_enforce_login	boolean		<input type="checkbox"/>	Only signed-in users can join this meeting
settings_host_video	boolean		<input type="checkbox"/>	Start video when host joins meeting
settings_in_meeting	boolean		<input type="checkbox"/>	Host meeting in India
settings_join_before_host	boolean		<input type="checkbox"/>	Allow participants to join the meeting before the host starts the meeting. Only used for scheduled or recurring meetings.
settings_mute_upon_entry	boolean		<input type="checkbox"/>	Mute participants upon entry



Name	Data Type	Label	Required	Documentation
settings_participant_video	boolean		<input type="checkbox"/>	Start video when participants join meeting
settings_registration_type	int64		<input type="checkbox"/>	Registration type. Used for recurring meeting with fixed time only.
settings_use_pmi	boolean		<input type="checkbox"/>	Use Personal Meeting ID. Only used for scheduled meetings and recurring meetings with no fixed time.
settings_waiting_room	boolean		<input type="checkbox"/>	Enable waiting room
settings_watermark	boolean		<input type="checkbox"/>	Add watermark when viewing shared screen
start_time	datetime		<input type="checkbox"/>	Meeting start time
start_url	string		<input type="checkbox"/>	Start url
timezone	string		<input type="checkbox"/>	Timezone to format start_time
topic	string		<input type="checkbox"/>	Meeting topic
type	int64		<input type="checkbox"/>	Meeting Type
uuid	string		<input type="checkbox"/>	Meeting unique ID

#### 11.1.4 MeetingsByMeetingIdInvitation

Retrieve meeting invitation Retrieve a meeting invitation

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/invitation`

Insert Zoom API URL: `/meetings/{meetingId}/invitation`

Update Zoom API URL: `/meetings/{meetingId}/invitation`

Delete Zoom API URL: `/meetings/{meetingId}/invitation`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /meetings/{meetingId}/invitation`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingIdInvitation`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four

parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `MeetingsByMeetingIdInvitation` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
invitation	string		<input type="checkbox"/>	Meeting invitation

### 11.1.5 MeetingsByMeetingIdPolls

List a meeting's polls List polls of a meeting

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/polls`

Insert Zoom API URL: `/meetings/{meetingId}/polls`

Update Zoom API URL: `/meetings/{meetingId}/polls`

Delete Zoom API URL: `/meetings/{meetingId}/polls`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /meetings/{meetingId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingIdPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `MeetingsByMeetingIdPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 11.1.6 MeetingsByMeetingIdPollsPollId

Retrieve a meeting's pollRetrieve a meeting's poll

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Insert Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Update Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Delete Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /meetings/{meetingId}/polls/{pollId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingIdPollsPollId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the

default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
pollId	string	<input checked="" type="checkbox"/>		The poll ID

## Table Function Columns

The columns of the table function `MeetingsByMeetingIdPollsPollId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Meeting Poll ID
status	string		<input type="checkbox"/>	Status of the Meeting Poll
title	string		<input type="checkbox"/>	Poll Title

### 11.1.7 MeetingsByMeetingIdRegistrants

List a meeting's registrants

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/registrants`

Insert Zoom API URL: `/meetings/{meetingId}/registrants`

Update Zoom API URL: `/meetings/{meetingId}/registrants`

Delete Zoom API URL: `/meetings/{meetingId}/registrants`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /meetings/{meetingId}/registrants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `MeetingsByMeetingIdRegistrants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
occurrence_id	string	<input type="checkbox"/>		The meeting occurrence ID
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
status	string	<input type="checkbox"/>	approved	The registrant status (Values: pending, approved, denied)

## Table Function Columns

The columns of the table function `MeetingsByMeetingIdRegistrants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_number	int64		<input type="checkbox"/>	The page number of current results
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 11.1.8 PastMeetingsByMeetingIdInstances

List of ended meeting instances

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/past_meetings/{meetingId}/instances`

Insert Zoom API URL: `/past_meetings/{meetingId}/instances`

Update Zoom API URL: `/past_meetings/{meetingId}/instances`

Delete Zoom API URL: `/past_meetings/{meetingId}/instances`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /past_meetings/{meetingId}/instances`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `PastMeetingsByMeetingIdInstances`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `PastMeetingsByMeetingIdInstances` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
DUMMY	string(1)		<input checked="" type="checkbox"/>	Default column added since the specification specifies that no data is returned.

### 11.1.9 PastMeetingsByMeetingUUID

Retrieve past meeting details Retrieve ended meeting details

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/past_meetings/{meetingUUID}`

Insert Zoom API URL: `/past_meetings/{meetingUUID}`

Update Zoom API URL: `/past_meetings/{meetingUUID}`

Delete Zoom API URL: `/past_meetings/{meetingUUID}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /past_meetings/{meetingUUID}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `PastMeetingsByMeetingUUID`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingUUID	string	<input checked="" type="checkbox"/>		The meeting UUID.

## Table Function Columns

The columns of the table function `PastMeetingsByMeetingUUID` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	int64		<input type="checkbox"/>	Meeting duration
end_time	datetime		<input type="checkbox"/>	Meeting end time
host_id	int64		<input type="checkbox"/>	Host ID
id	int64		<input type="checkbox"/>	Meeting ID
participants_count	int64		<input type="checkbox"/>	Number of meeting participants
start_time	datetime		<input type="checkbox"/>	Meeting start time
topic	string		<input type="checkbox"/>	Meeting topic
total_minutes	int64		<input type="checkbox"/>	Number of meeting minutes
type	int64		<input type="checkbox"/>	Meeting type
user_email	string		<input type="checkbox"/>	User email
user_name	string		<input type="checkbox"/>	User display name
uuid	guid		<input type="checkbox"/>	Meeting UUID

### 11.1.10 PastMeetingsByMeetingUUIDParticipants

Retrieve past meeting participants  
Retrieve ended meeting participants

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/past_meetings/{meetingUUID}/participants`

Insert Zoom API URL: `/past_meetings/{meetingUUID}/participants`

Update Zoom API URL: `/past_meetings/{meetingUUID}/participants`

Delete Zoom API URL: `/past_meetings/{meetingUUID}/participants`

Field Selection Method: `NotRequired`

Base Path: `participants[*]`

Select Zoom API Operation: `get /past_meetings/{meetingUUID}/participants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `PastMeetingsByMeetingUUIDParticipants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingUUID	string	<input checked="" type="checkbox"/>		The meeting UUID.
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call

## Table Function Columns

The columns of the table function `PastMeetingsByMeetingUUIDParticipants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	guid		<input type="checkbox"/>	Participant UUID



Name	Data Type	Label	Required	Documentation
name	string		<input type="checkbox"/>	Participant display name

### 11.1.11 patchMeetingsByMeetingId

Update a meetingUpdate a meeting's details

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}`

Insert Zoom API URL: `/meetings/{meetingId}`

Update Zoom API URL: `/meetings/{meetingId}`

Delete Zoom API URL: `/meetings/{meetingId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /meetings/{meetingId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchMeetingsByMeetingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `patchMeetingsByMeetingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.12 patchMeetingsByMeetingIdLivestream

Update a meeting live stream Update a meeting's live stream

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/livestream`

Insert Zoom API URL: `/meetings/{meetingId}/livestream`

Update Zoom API URL: `/meetings/{meetingId}/livestream`

Delete Zoom API URL: `/meetings/{meetingId}/livestream`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /meetings/{meetingId}/livestream`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchMeetingsByMeetingIdLivestream`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `patchMeetingsByMeetingIdLivestream` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.13 patchMeetingsByMeetingIdLivestreamStatus

Update a meeting live stream status Update a meeting's live stream status

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/livestream/status`

Insert Zoom API URL: `/meetings/{meetingId}/livestream/status`

Update Zoom API URL: `/meetings/{meetingId}/livestream/status`

Delete Zoom API URL: `/meetings/{meetingId}/livestream/status`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /meetings/{meetingId}/livestream/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchMeetingsByMeetingIdLivestreamStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `patchMeetingsByMeetingIdLivestreamStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

#### 11.1.14 postMeetingsByMeetingIdPolls

Create a meeting's poll>Create a poll for a meeting

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/polls`

Insert Zoom API URL: `/meetings/{meetingId}/polls`

Update Zoom API URL: `/meetings/{meetingId}/polls`

Delete Zoom API URL: `/meetings/{meetingId}/polls`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /meetings/{meetingId}/polls`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postMeetingsByMeetingIdPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting poll object
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

### Table Function Columns

The columns of the table function `postMeetingsByMeetingIdPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Meeting Poll ID
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
status	string		<input type="checkbox"/>	Status of the Meeting Poll
title	string		<input type="checkbox"/>	Poll Title

### 11.1.15 postMeetingsByMeetingIdRegistrants

Add a meeting registrantRegister a participant for a meeting

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/registrants`

Insert Zoom API URL: `/meetings/{meetingId}/registrants`

Update Zoom API URL: `/meetings/{meetingId}/registrants`

Delete Zoom API URL: `/meetings/{meetingId}/registrants`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /meetings/{meetingId}/registrants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postMeetingsByMeetingIdRegistrants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
occurrence_ids	string	<input type="checkbox"/>		Occurrence IDs. You can find these with the meeting get API. Multiple values separated by comma.

## Table Function Columns

The columns of the table function `postMeetingsByMeetingIdRegistrants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Registrant ID
join_url	string		<input type="checkbox"/>	Join URL for this registrant
registrant_id	string		<input type="checkbox"/>	Registrant ID
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
start_time	datetime		<input type="checkbox"/>	Start time
topic	string		<input type="checkbox"/>	Topic

### 11.1.16 postUsersByUserIdMeetings

Create a meetingCreate a meeting for a user <aside>The expiration time of start\_url is two hours. But for API users, the expiration time is 90 days.</aside>

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/meetings`

Insert Zoom API URL: `/users/{userId}/meetings`

Update Zoom API URL: `/users/{userId}/meetings`

Delete Zoom API URL: `/users/{userId}/meetings`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /users/{userId}/meetings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postUsersByUserIdMeetings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the

default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting object
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `postUsersByUserIdMeetings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
agenda	string		<input type="checkbox"/>	Agenda
created_at	datetime		<input type="checkbox"/>	Create time
duration	int64		<input type="checkbox"/>	Meeting duration
h323_password	string		<input type="checkbox"/>	H.323/SIP room system password
host_id	string		<input type="checkbox"/>	ID of the user set as host of meeting
id	string		<input type="checkbox"/>	Meeting ID, also known as meeting number
join_url	string		<input type="checkbox"/>	Join url
password	string		<input type="checkbox"/>	Meeting password
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
settings_alternative_hosts	string		<input type="checkbox"/>	Alternative hosts emails or IDs. Multiple value separated by comma.
settings_approval_type	int64		<input type="checkbox"/>	
settings_audio	string		<input type="checkbox"/>	Determine how participants can join the audio portion of the meeting
settings_auto_recording	string		<input type="checkbox"/>	
settings_close_registration	boolean		<input type="checkbox"/>	Close registration after event date
settings_cn_meeting	boolean		<input type="checkbox"/>	Host meeting in China
settings_enforce_login_domains	string		<input type="checkbox"/>	Only signed-in users with specified domains can join meetings
settings_enforce_login	boolean		<input type="checkbox"/>	Only signed-in users can join this meeting
settings_host_video	boolean		<input type="checkbox"/>	Start video when host joins meeting
settings_in_meeting	boolean		<input type="checkbox"/>	Host meeting in India
settings_join_before_host	boolean		<input type="checkbox"/>	Allow participants to join the meeting before the host starts the meeting. Only used for scheduled or recurring meetings.
settings_mute_upon_entry	boolean		<input type="checkbox"/>	Mute participants upon entry

Name	Data Type	Label	Required	Documentation
settings_participant_video	boolean		<input type="checkbox"/>	Start video when participants join meeting
settings_registration_type	int64		<input type="checkbox"/>	Registration type. Used for recurring meeting with fixed time only.
settings_use_pmi	boolean		<input type="checkbox"/>	Use Personal Meeting ID. Only used for scheduled meetings and recurring meetings with no fixed time.
settings_waiting_room	boolean		<input type="checkbox"/>	Enable waiting room
settings_watermark	boolean		<input type="checkbox"/>	Add watermark when viewing shared screen
start_time	datetime		<input type="checkbox"/>	Meeting start time
start_url	string		<input type="checkbox"/>	Start url
timezone	string		<input type="checkbox"/>	Timezone to format start_time
topic	string		<input type="checkbox"/>	Meeting topic
type	int64		<input type="checkbox"/>	Meeting Type
uuid	string		<input type="checkbox"/>	Meeting unique ID

### 11.1.17 putMeetingsByMeetingIdPollsPollId

Update a meeting's poll

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Insert Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Update Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Delete Zoom API URL: `/meetings/{meetingId}/polls/{pollId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /meetings/{meetingId}/polls/{pollId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putMeetingsByMeetingIdPollsPollId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four



parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Meeting Poll
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
pollId	string	<input checked="" type="checkbox"/>		The poll ID

## Table Function Columns

The columns of the table function `putMeetingsByMeetingIdPollsPollId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.18 putMeetingsByMeetingIdRegistrantsStatus

Update a meeting registrant's status

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/registrants/status`

Insert Zoom API URL: `/meetings/{meetingId}/registrants/status`

Update Zoom API URL: `/meetings/{meetingId}/registrants/status`

Delete Zoom API URL: `/meetings/{meetingId}/registrants/status`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /meetings/{meetingId}/registrants/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putMeetingsByMeetingIdRegistrantsStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID
occurrence_id	string	<input type="checkbox"/>		The meeting occurrence ID

## Table Function Columns

The columns of the table function `putMeetingsByMeetingIdRegistrantsStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.19 putMeetingsByMeetingIdStatus

Update a meeting's status

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/meetings/{meetingId}/status`

Insert Zoom API URL: `/meetings/{meetingId}/status`

Update Zoom API URL: `/meetings/{meetingId}/status`

Delete Zoom API URL: `/meetings/{meetingId}/status`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /meetings/{meetingId}/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putMeetingsByMeetingIdStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
meetingId	int64	<input checked="" type="checkbox"/>		The meeting ID

## Table Function Columns

The columns of the table function `putMeetingsByMeetingIdStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 11.1.20 UsersByUserIdMeetings

List meetings List meetings for a user

Catalog: Zoom

Schema: Meetings

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/meetings`

Insert Zoom API URL: `/users/{userId}/meetings`

Update Zoom API URL: `/users/{userId}/meetings`

Delete Zoom API URL: `/users/{userId}/meetings`

Field Selection Method: NotRequired

Base Path: `meetings[*]`

Select Zoom API Operation: `get /users/{userId}/meetings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdMeetings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
type	string	<input type="checkbox"/>	live	The meeting type (Values: scheduled, live, upcoming)
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdMeetings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
created_at	datetime		<input type="checkbox"/>	Create time
duration	int64		<input type="checkbox"/>	Meeting duration
host_id	string		<input type="checkbox"/>	ID of the user set as host of meeting
id	string		<input type="checkbox"/>	Meeting ID, also know as meeting number
join_url	string		<input type="checkbox"/>	Join url
start_time	datetime		<input type="checkbox"/>	Meeting start time
timezone	string		<input type="checkbox"/>	Timezone to format start_time
topic	string		<input type="checkbox"/>	Meeting topic
type	int64		<input type="checkbox"/>	Meeting Type
uuid	string		<input type="checkbox"/>	Meeting unique ID

## 12 Schema: Native

### 12.1 Tables

#### 12.1.1 NATIVEPLATFORMSCALARREQUESTS: Zoom Native Platform Scalar Requests

Direct access to native API.

Catalog: Zoom

Schema: Native

Alias: npt

Label: Native Platform Scalar Requests

Documentation:

The NativePlatformScalarRequests table provides direct access to the native API protocol over an established connection to the Zoom API server. It will contain a new row for every row inserted with a native API request in PAYLOAD\_TEXT with the results of unaltered forwarding of the payload to the Zoom API server.

Retrieve: true

Insert: true

Update: false

Delete: false

## View Columns

The columns of the view NATIVEPLATFORMSCALARREQUESTS are shown below. Each column has an SQL data type. A new non-null value must be provided for every required column at all times during insert.

Name	Data Type	Label	Required	Documentation
BLOB_PREFERRED	boolean	BLOB Preferred	<input checked="" type="checkbox"/>	Indicator whether a BLOB result is preferred over text.
BOL_RESPONSE_CACHE_MAX_AGE_SEC	int32	Response Cache Maximum Age (sec)	<input type="checkbox"/>	Maximum age in seconds of Bridge Online response cache entries to be used.
CONTENT_TYPE	string(240)	Content Type	<input type="checkbox"/>	
DATE_ENDED	datetime	End Date	<input checked="" type="checkbox"/>	
DATE_STARTED	datetime	Start Date	<input checked="" type="checkbox"/>	
DRY_RUN	boolean	Run without Actions	<input checked="" type="checkbox"/>	
DURATION_MS	int32	Duration (ms)	<input checked="" type="checkbox"/>	
ERROR_MESSAGE_CODE	string(30)	Error Message Code	<input type="checkbox"/>	
ERROR_MESSAGE_TEXT	string(32000)	Error Message Text	<input type="checkbox"/>	
FAIL_ON_ERROR	boolean	Fail on Error	<input checked="" type="checkbox"/>	Whether to raise an exception when processing the native request triggered an error from the provider.
HTTP_DISK_CACHE_MAX_AGE_SEC	int32	HTTP Disk Cache Maximum Age (sec)	<input type="checkbox"/>	Maximum age in seconds of HTTP disk cache entries to be used.
HTTP_DISK_CACHE_SAVE	boolean	Save HTTP Disk Cache	<input type="checkbox"/>	Whether results can be stored in HTTP disk cache.
HTTP_DISK_CACHE_USE	boolean	Use HTTP Disk Cache	<input type="checkbox"/>	Whether results can be fetched from HTTP disk cache.
HTTP_MEMORY_CACHE_MAX_AGE_SEC	int32	HTTP Memory Cache Maximum Age (sec)	<input type="checkbox"/>	Maximum age in seconds of HTTP memory cache entries to be used.
HTTP_MEMORY_CACHE_SAVE	boolean	Save HTTP Memory Cache	<input type="checkbox"/>	Whether results can be stored in HTTP memory cache.
HTTP_MEMORY_CACHE_USE	boolean	Use HTTP Memory Cache	<input type="checkbox"/>	Whether results can be fetched from HTTP memory cache.

Name	Data Type	Label	Required	Documentation
HTTP_METHOD	string(30)	HTTP Method	<input type="checkbox"/>	
HTTP_STATUS_CODE	int16	HTTP Status Code	<input type="checkbox"/>	
ORIG_SYSTEM_GROUP	string(4000)	Original System Group	<input type="checkbox"/>	
ORIG_SYSTEM_REFERENCE	string(4000)	Original System Reference	<input type="checkbox"/>	
PAYLOAD_TEXT	string	Payload	<input type="checkbox"/>	
RESULT_BLOB	byte[]	Result BLOB	<input type="checkbox"/>	
RESULT_DATE_TIME_UTC	datetime		<input type="checkbox"/>	
RESULT_NUMBER	decimal		<input type="checkbox"/>	
RESULT_TEXT	string	Result Text	<input type="checkbox"/>	
SUCCESSFUL	boolean	Successful	<input checked="" type="checkbox"/>	
TIMEOUT_SEC	int32	Timeout (sec)	<input type="checkbox"/>	Timeout in seconds.
TRANSACTION_ID	int32	Transaction ID	<input checked="" type="checkbox"/>	Incrementing ID of the transaction.
URL	string(4000)	URL	<input type="checkbox"/>	

## 13 Schema: PAC

### 13.1 Tables

#### 13.1.1 UsersByUserId\_Tsp\_accountsDedicated\_dial\_in\_numberPac

List user's PAC accounts

Catalog: Zoom

Schema: PAC

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/pac`

Insert Zoom API URL: `/users/{userId}/pac`

Update Zoom API URL: `/users/{userId}/pac`

Delete Zoom API URL: `/users/{userId}/pac`

Field Selection Method: NotRequired

Base Path: `tsp_accounts[*].dedicated_dial_in_number[*]`

Select Zoom API Operation: `get /users/{userId}/pac`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserId_Tsp_accountsDedicated_dial_in_numberPac`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserId_Tsp_accountsDedicated_dial_in_numberPac` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
conference_id	int64		<input type="checkbox"/>	Conference ID
country	string		<input type="checkbox"/>	Country Code
listen_only_password	string		<input type="checkbox"/>	Listen-Only Password, numeric value, length is less than 6
number	string		<input type="checkbox"/>	Dial-in number, length is less than 16
participant_password	string		<input type="checkbox"/>	Participant Password, numeric value, length is less than 6

### 13.1.2 UsersByUserId\_Tsp\_accountsGlobal\_dial\_in\_numbersPac

List user's PAC accounts

Catalog: Zoom

Schema: PAC

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/pac`

Insert Zoom API URL: `/users/{userId}/pac`

Update Zoom API URL: `/users/{userId}/pac`

Delete Zoom API URL: `/users/{userId}/pac`

Field Selection Method: NotRequired

Base Path: `tsp_accounts[*].global_dial_in_numbers[*]`

Select Zoom API Operation: `get /users/{userId}/pac`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserId_Tsp_accountsGlobal_dial_in_numbersPac`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserId_Tsp_accountsGlobal_dial_in_numbersPac` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
conference_id	int64		<input type="checkbox"/>	Conference ID
country	string		<input type="checkbox"/>	Country Code
listen_only_password	string		<input type="checkbox"/>	Listen-Only Password, numeric value, length is less than 6
number	string		<input type="checkbox"/>	Dial-in number, length is less than 16
participant_password	string		<input type="checkbox"/>	Participant Password, numeric value, length is less than 6

### 13.1.3 UsersByUserIdPac

List user's PAC accounts

Catalog: Zoom

Schema: PAC

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/pac`

Insert Zoom API URL: `/users/{userId}/pac`

Update Zoom API URL: `/users/{userId}/pac`

Delete Zoom API URL: `/users/{userId}/pac`

Field Selection Method: NotRequired



Base Path: `tsp_accounts[*]`

Select Zoom API Operation: `get /users/{userId}/pac`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdPac`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdPac` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
conference_id	int64		<input type="checkbox"/>	Conference ID
listen_only_password	string		<input type="checkbox"/>	Listen-Only Password, numeric value, length is less than 6
participant_password	string		<input type="checkbox"/>	Participant Password, numeric value, length is less than 6

## 14 Schema: Reports

### 14.1 Tables

#### 14.1.1 Report\_DatesDaily

Retrieve daily report Retrieve daily report for one month, can only get daily report for recent 6 months

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/daily`

Insert Zoom API URL: `/report/daily`

Update Zoom API URL: `/report/daily`

Delete Zoom API URL: `/report/daily`

Field Selection Method: `NotRequired`

Base Path: `dates[*]`

Select Zoom API Operation: `get /report/daily`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Report_DatesDaily`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
month	int64	<input type="checkbox"/>		Month for this report
year	int64	<input type="checkbox"/>		Year for this report

## Table Function Columns

The columns of the table function `Report_DatesDaily` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
date	datetime		<input type="checkbox"/>	Date for this object
meeting_minutes	int64		<input type="checkbox"/>	Number of meeting minutes on this date
meetings	int64		<input type="checkbox"/>	Number of meetings on this date
month	int64		<input type="checkbox"/>	Month for this report
new_users	int64		<input type="checkbox"/>	Number of new users on this date
participants	int64		<input type="checkbox"/>	Number of participants on this date
year	int64		<input type="checkbox"/>	Year for this report

### 14.1.2 Report\_Telephony\_usageTelephone

Retrieve telephone reportRetrieve telephone report for a specified period <aside>Toll Report option would be removed.</aside>.

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/telephone`

Insert Zoom API URL: `/report/telephone`

Update Zoom API URL: `/report/telephone`

Delete Zoom API URL: `/report/telephone`

Field Selection Method: NotRequired

Base Path: `telephony_usage[*]`

Select Zoom API Operation: `get /report/telephone`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Report_Telephony_usageTelephone`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
type	string	<input type="checkbox"/>	1	Audio type (Values: 1)

## Table Function Columns

The columns of the table function `Report_Telephony_usageTelephone` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
call_in_number	string		<input type="checkbox"/>	Call in number
country_name	string		<input type="checkbox"/>	Country Name
dept	string		<input type="checkbox"/>	User department
duration	int64		<input type="checkbox"/>	Meeting duration
end_time	datetime		<input type="checkbox"/>	Meeting end time
from	datetime		<input type="checkbox"/>	Start date for this report
host_email	string		<input type="checkbox"/>	User email
host_name	string		<input type="checkbox"/>	User display name
meeting_id	int64		<input type="checkbox"/>	Meeting ID
meeting_type	string		<input type="checkbox"/>	Meeting Type
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_number	int64		<input type="checkbox"/>	The page number of current results
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call
phone_number	string		<input type="checkbox"/>	Telephone Number
start_time	datetime		<input type="checkbox"/>	Meeting start time
to	datetime		<input type="checkbox"/>	End date for this report
total_records	int64		<input type="checkbox"/>	The number of all records available across pages
total	decimal		<input type="checkbox"/>	Total

### 14.1.3 Report\_UsersUsers

Retrieve hosts reportRetrieve active or inactive hosts report for a specified period

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/users`

Insert Zoom API URL: `/report/users`

Update Zoom API URL: `/report/users`

Delete Zoom API URL: `/report/users`

Field Selection Method: `NotRequired`

Base Path: `users[*]`

Select Zoom API Operation: `get /report/users`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Report_UsersUsers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
type	string	<input type="checkbox"/>		Active hosts or inactive hosts (Values: active, inactive)

## Table Function Columns

The columns of the table function `Report_UsersUsers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
dept	string		<input type="checkbox"/>	User department
email	string		<input type="checkbox"/>	User email
from	datetime		<input type="checkbox"/>	Start date for this report
id	guid		<input type="checkbox"/>	User ID
meeting_minutes	int64		<input type="checkbox"/>	Number of meeting minutes for user
meetings	int64		<input type="checkbox"/>	Number of meetings for user
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_number	int64		<input type="checkbox"/>	The page number of current results
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call
participants	int64		<input type="checkbox"/>	Number of participants in meetings for user
to	datetime		<input type="checkbox"/>	End date for this report

Name	Data Type	Label	Required	Documentation
total_meeting_minutes	int64		<input type="checkbox"/>	Number of meeting minutes for this range
total_meetings	int64		<input type="checkbox"/>	Number of meetings for this range
total_participants	int64		<input type="checkbox"/>	Number of participants for this range
total_records	int64		<input type="checkbox"/>	The number of all records available across pages
type	int64		<input type="checkbox"/>	User type
user_name	string		<input type="checkbox"/>	User display name

#### 14.1.4 ReportCloudRecording

Retrieve cloud recording usage report. Retrieve cloud recording usage report for a specified period. You can only get cloud recording reports for the most recent period of 6 months. The date gap between from and to dates should be smaller or equal to 30 days.

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/cloud_recording`

Insert Zoom API URL: `/report/cloud_recording`

Update Zoom API URL: `/report/cloud_recording`

Delete Zoom API URL: `/report/cloud_recording`

Field Selection Method: NotRequired

Base Path: `cloud_recording_storage[*]`

Select Zoom API Operation: `get /report/cloud_recording`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportCloudRecording`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
to	datetime	<input checked="" type="checkbox"/>		End Date

## Table Function Columns

The columns of the table function `ReportCloudRecording` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
date	datetime		<input type="checkbox"/>	
free_usage	string		<input type="checkbox"/>	
plan_usage	string		<input type="checkbox"/>	
usage	string		<input type="checkbox"/>	

### 14.1.5 ReportDaily

Retrieve daily reportRetrieve daily report for one month, can only get daily report for recent 6 months

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/daily`

Insert Zoom API URL: `/report/daily`

Update Zoom API URL: `/report/daily`

Delete Zoom API URL: `/report/daily`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /report/daily`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportDaily`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
month	int64	<input type="checkbox"/>		Month for this report
year	int64	<input type="checkbox"/>		Year for this report

## Table Function Columns

The columns of the table function `ReportDaily` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
month	int64		<input type="checkbox"/>	Month for this report
year	int64		<input type="checkbox"/>	Year for this report

### 14.1.6 ReportMeetings\_Tracking\_fieldsByMeetingId

Retrieve meeting details reportRetrieve ended meeting details report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/meetings/{meetingId}`

Insert Zoom API URL: `/report/meetings/{meetingId}`

Update Zoom API URL: `/report/meetings/{meetingId}`

Delete Zoom API URL: `/report/meetings/{meetingId}`

Field Selection Method: NotRequired

Base Path: `tracking_fields[*]`

Select Zoom API Operation: `get /report/meetings/{meetingId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportMeetings_Tracking_fieldsByMeetingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with `select \* from table(value1, value2, value3)` on a table with four



parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `ReportMeetings_Tracking_fieldsByMeetingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	int64		<input type="checkbox"/>	Meeting duration
end_time	datetime		<input type="checkbox"/>	Meeting end time
field	string		<input type="checkbox"/>	Tracking fields type
id	int64		<input type="checkbox"/>	Meeting ID
participants_count	int64		<input type="checkbox"/>	Number of meeting participants
start_time	datetime		<input type="checkbox"/>	Meeting start time
topic	string		<input type="checkbox"/>	Meeting topic
total_minutes	int64		<input type="checkbox"/>	Number of meeting minutes
type	int64		<input type="checkbox"/>	Meeting type
user_email	string		<input type="checkbox"/>	User email
user_name	string		<input type="checkbox"/>	User display name
uuid	guid		<input type="checkbox"/>	Meeting UUID
value	string		<input type="checkbox"/>	Tracking fields value

### 14.1.7 ReportMeetingsByMeetingId

Retrieve meeting details reportRetrieve ended meeting details report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/meetings/{meetingId}`

Insert Zoom API URL: `/report/meetings/{meetingId}`

Update Zoom API URL: `/report/meetings/{meetingId}`

Delete Zoom API URL: `/report/meetings/{meetingId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /report/meetings/{meetingId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportMeetingsByMeetingId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `ReportMeetingsByMeetingId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	int64		<input type="checkbox"/>	Meeting duration
end_time	datetime		<input type="checkbox"/>	Meeting end time
id	int64		<input type="checkbox"/>	Meeting ID
participants_count	int64		<input type="checkbox"/>	Number of meeting participants
start_time	datetime		<input type="checkbox"/>	Meeting start time
topic	string		<input type="checkbox"/>	Meeting topic
total_minutes	int64		<input type="checkbox"/>	Number of meeting minutes
type	int64		<input type="checkbox"/>	Meeting type
user_email	string		<input type="checkbox"/>	User email
user_name	string		<input type="checkbox"/>	User display name
uuid	guid		<input type="checkbox"/>	Meeting UUID

### 14.1.8 ReportMeetingsByMeetingId\_QuestionsPolls

Retrieve meeting polls reportRetrieve ended meeting polls report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/meetings/{meetingId}/polls`

Insert Zoom API URL: `/report/meetings/{meetingId}/polls`

Update Zoom API URL: `/report/meetings/{meetingId}/polls`

Delete Zoom API URL: `/report/meetings/{meetingId}/polls`

Field Selection Method: NotRequired

Base Path: `questions[*]`

Select Zoom API Operation: `get /report/meetings/{meetingId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportMeetingsByMeetingId_QuestionsPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `ReportMeetingsByMeetingId_QuestionsPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email	string		<input type="checkbox"/>	Participant email
id	int64		<input type="checkbox"/>	Meeting ID
name	string		<input type="checkbox"/>	Participant display name

Name	Data Type	Label	Required	Documentation
start_time	datetime		<input type="checkbox"/>	Meeting start time
uuid	guid		<input type="checkbox"/>	Meeting UUID

#### 14.1.9 ReportMeetingsByMeetingId\_QuestionsQuestion\_detailsPolls

Retrieve meeting polls reportRetrieve ended meeting polls report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/meetings/{meetingId}/polls`

Insert Zoom API URL: `/report/meetings/{meetingId}/polls`

Update Zoom API URL: `/report/meetings/{meetingId}/polls`

Delete Zoom API URL: `/report/meetings/{meetingId}/polls`

Field Selection Method: NotRequired

Base Path: `questions[*].question_details[*]`

Select Zoom API Operation: `get /report/meetings/{meetingId}/polls`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportMeetingsByMeetingId_QuestionsQuestion_detailsPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

### Table Function Columns

The columns of the table function `ReportMeetingsByMeetingId_QuestionsQuestion_detailsPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
answer	string		<input type="checkbox"/>	Given answer
email	string		<input type="checkbox"/>	Participant email
id	int64		<input type="checkbox"/>	Meeting ID
name	string		<input type="checkbox"/>	Participant display name
question	string		<input type="checkbox"/>	Asked question
start_time	datetime		<input type="checkbox"/>	Meeting start time
uuid	guid		<input type="checkbox"/>	Meeting UUID

#### 14.1.10 ReportMeetingsByMeetingIdParticipants

Retrieve meeting participants report Retrieve ended meeting participants report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/meetings/{meetingId}/participants`

Insert Zoom API URL: `/report/meetings/{meetingId}/participants`

Update Zoom API URL: `/report/meetings/{meetingId}/participants`

Delete Zoom API URL: `/report/meetings/{meetingId}/participants`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /report/meetings/{meetingId}/participants`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportMeetingsByMeetingIdParticipants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call

## Table Function Columns

The columns of the table function `ReportMeetingsByMeetingIdParticipants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
attentiveness_score	int64		<input type="checkbox"/>	Participant attentiveness score
duration	int64		<input type="checkbox"/>	Participant duration
id	guid		<input type="checkbox"/>	Participant UUID
join_time	datetime		<input type="checkbox"/>	Participant join time
leave_time	datetime		<input type="checkbox"/>	Participant leave time
name	string		<input type="checkbox"/>	Participant display name
user_email	string		<input type="checkbox"/>	Participant email
user_id	string		<input type="checkbox"/>	Participant ID

### 14.1.11 ReportMeetingsByMeetingIdPolls

Retrieve meeting polls `reportRetrieve ended meeting polls report`

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/meetings/{meetingId}/polls`

Insert Zoom API URL: `/report/meetings/{meetingId}/polls`

Update Zoom API URL: `/report/meetings/{meetingId}/polls`

Delete Zoom API URL: `/report/meetings/{meetingId}/polls`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /report/meetings/{meetingId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportMeetingsByMeetingIdPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
meetingId	string	<input checked="" type="checkbox"/>		The meeting ID or meeting UUID. If given meeting ID, will take the last meeting instance.

## Table Function Columns

The columns of the table function `ReportMeetingsByMeetingIdPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	int64		<input type="checkbox"/>	Meeting ID
start_time	datetime		<input type="checkbox"/>	Meeting start time
uuid	guid		<input type="checkbox"/>	Meeting UUID

### 14.1.12 ReportTelephone

Retrieve telephone report Retrieve telephone report for a specified period <aside>Toll Report option would be removed.</aside>.

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/telephone`

Insert Zoom API URL: `/report/telephone`

Update Zoom API URL: `/report/telephone`

Delete Zoom API URL: `/report/telephone`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /report/telephone`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportTelephone`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
type	string	<input type="checkbox"/>	1	Audio type (Values: 1)

## Table Function Columns

The columns of the table function `ReportTelephone` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date for this report
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_number	int64		<input type="checkbox"/>	The page number of current results
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call
to	datetime		<input type="checkbox"/>	End date for this report
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 14.1.13 ReportUsers

Retrieve hosts report  
Retrieve active or inactive hosts report for a specified period

Catalog: Zoom



## Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/users`

Insert Zoom API URL: `/report/users`

Update Zoom API URL: `/report/users`

Delete Zoom API URL: `/report/users`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /report/users`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportUsers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>from</code>	<code>datetime</code>	<input checked="" type="checkbox"/>		Start Date
<code>page_number</code>	<code>int64</code>	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	<code>int64</code>	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>to</code>	<code>datetime</code>	<input checked="" type="checkbox"/>		End Date
<code>type</code>	<code>string</code>	<input type="checkbox"/>		Active hosts or inactive hosts (Values: active, inactive)

## Table Function Columns

The columns of the table function `ReportUsers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>from</code>	<code>datetime</code>		<input type="checkbox"/>	Start date for this report
<code>page_count</code>	<code>int64</code>		<input type="checkbox"/>	The number of items returned on this page

Name	Data Type	Label	Required	Documentation
page_number	int64		<input type="checkbox"/>	The page number of current results
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call
to	datetime		<input type="checkbox"/>	End date for this report
total_meeting_minutes	int64		<input type="checkbox"/>	Number of meeting minutes for this range
total_meetings	int64		<input type="checkbox"/>	Number of meetings for this range
total_participants	int64		<input type="checkbox"/>	Number of participants for this range
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

#### 14.1.14 ReportUsersByUserId\_MeetingsMeetings

Retrieve meetings report Retrieve ended meetings report for a specified period

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/users/{userId}/meetings`

Insert Zoom API URL: `/report/users/{userId}/meetings`

Update Zoom API URL: `/report/users/{userId}/meetings`

Delete Zoom API URL: `/report/users/{userId}/meetings`

Field Selection Method: NotRequired

Base Path: `meetings[*]`

Select Zoom API Operation: `get /report/users/{userId}/meetings`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportUsersByUserId_MeetingsMeetings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the

default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
to	datetime	<input checked="" type="checkbox"/>		End Date
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `ReportUsersByUserId_MeetingsMeetings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	int64		<input type="checkbox"/>	Meeting duration
end_time	datetime		<input type="checkbox"/>	Meeting end time
from	datetime		<input type="checkbox"/>	Start date for this report
id	int64		<input type="checkbox"/>	Meeting ID
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
participants_count	int64		<input type="checkbox"/>	Number of meeting participants
start_time	datetime		<input type="checkbox"/>	Meeting start time
to	datetime		<input type="checkbox"/>	End date for this report
topic	string		<input type="checkbox"/>	Meeting topic
total_minutes	int64		<input type="checkbox"/>	Number of meeting minutes
total_records	int64		<input type="checkbox"/>	The number of all records available across pages
type	int64		<input type="checkbox"/>	Meeting type
user_email	string		<input type="checkbox"/>	User email
user_name	string		<input type="checkbox"/>	User display name

Name	Data Type	Label	Required	Documentation
uuid	guid		<input type="checkbox"/>	Meeting UUID

#### 14.1.15 ReportUsersByUserIdMeetings

Retrieve meetings reportRetrieve ended meetings report for a specified period

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/users/{userId}/meetings`

Insert Zoom API URL: `/report/users/{userId}/meetings`

Update Zoom API URL: `/report/users/{userId}/meetings`

Delete Zoom API URL: `/report/users/{userId}/meetings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /report/users/{userId}/meetings`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportUsersByUserIdMeetings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
from	datetime	<input checked="" type="checkbox"/>		Start Date
next_page_token	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call

Name	Data Type	Required	Default Value	Documentation
to	datetime	<input checked="" type="checkbox"/>		End Date
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `ReportUsersByUserIdMeetings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
from	datetime		<input type="checkbox"/>	Start date for this report
next_page_token	string		<input type="checkbox"/>	Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
page_count	int64		<input type="checkbox"/>	The number of items returned on this page
page_size	int64		<input type="checkbox"/>	The number of records returned within a single API call.
to	datetime		<input type="checkbox"/>	End date for this report
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 14.1.16 ReportWebinars\_Tracking\_fieldsByWebinarId

Retrieve webinar details report Retrieve ended webinar details report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}`

Insert Zoom API URL: `/report/webinars/{webinarId}`

Update Zoom API URL: `/report/webinars/{webinarId}`

Delete Zoom API URL: `/report/webinars/{webinarId}`

Field Selection Method: NotRequired

Base Path: `tracking_fields[*]`

Select Zoom API Operation: `get /report/webinars/{webinarId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinars_Tracking_fieldsByWebinarId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinars_Tracking_fieldsByWebinarId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
duration	int64		<input type="checkbox"/>	Meeting duration
end_time	datetime		<input type="checkbox"/>	Meeting end time
field	string		<input type="checkbox"/>	Tracking fields type
id	int64		<input type="checkbox"/>	Meeting ID
participants_count	int64		<input type="checkbox"/>	Number of meeting participants
start_time	datetime		<input type="checkbox"/>	Meeting start time
topic	string		<input type="checkbox"/>	Meeting topic
total_minutes	int64		<input type="checkbox"/>	Number of meeting minutes
type	int64		<input type="checkbox"/>	Meeting type
user_email	string		<input type="checkbox"/>	User email
user_name	string		<input type="checkbox"/>	User display name
uuid	guid		<input type="checkbox"/>	Meeting UUID
value	string		<input type="checkbox"/>	Tracking fields value

### 14.1.17 ReportWebinarsByWebinarId

Retrieve webinar details report Retrieve ended webinar details report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}`

Insert Zoom API URL: `/report/webinars/{webinarId}`

Update Zoom API URL: `/report/webinars/{webinarId}`

Delete Zoom API URL: `/report/webinars/{webinarId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /report/webinars/{webinarId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>webinarId</code>	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, w ill take the last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinarsByWebinarId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>duration</code>	int64		<input type="checkbox"/>	Meeting duration
<code>end_time</code>	datetime		<input type="checkbox"/>	Meeting end time
<code>id</code>	int64		<input type="checkbox"/>	Meeting ID
<code>participants_count</code>	int64		<input type="checkbox"/>	Number of meeting participants
<code>start_time</code>	datetime		<input type="checkbox"/>	Meeting start time
<code>topic</code>	string		<input type="checkbox"/>	Meeting topic
<code>total_minutes</code>	int64		<input type="checkbox"/>	Number of meeting minutes
<code>type</code>	int64		<input type="checkbox"/>	Meeting type

Name	Data Type	Label	Required	Documentation
user_email	string		<input type="checkbox"/>	User email
user_name	string		<input type="checkbox"/>	User display name
uuid	guid		<input type="checkbox"/>	Meeting UUID

#### 14.1.18 ReportWebinarsByWebinarId\_QuestionsPolls

Retrieve webinar polls reportRetrieve ended webinar polls report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/polls`

Insert Zoom API URL: `/report/webinars/{webinarId}/polls`

Update Zoom API URL: `/report/webinars/{webinarId}/polls`

Delete Zoom API URL: `/report/webinars/{webinarId}/polls`

Field Selection Method: NotRequired

Base Path: `questions[*]`

Select Zoom API Operation: `get /report/webinars/{webinarId}/polls`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarId_QuestionsPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

### Table Function Columns



The columns of the table function `ReportWebinarsByWebinarId_QuestionsPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email	string		<input type="checkbox"/>	Participant email
id	int64		<input type="checkbox"/>	Webinar ID
name	string		<input type="checkbox"/>	Participant display name
start_time	datetime		<input type="checkbox"/>	Webinar start time
uuid	guid		<input type="checkbox"/>	Webinar UUID

#### 14.1.19 ReportWebinarsByWebinarId\_QuestionsQa

Retrieve webinar Q&A report  
Retrieve ended webinar Q&A report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/qa`

Insert Zoom API URL: `/report/webinars/{webinarId}/qa`

Update Zoom API URL: `/report/webinars/{webinarId}/qa`

Delete Zoom API URL: `/report/webinars/{webinarId}/qa`

Field Selection Method: NotRequired

Base Path: `questions[*]`

Select Zoom API Operation: `get /report/webinars/{webinarId}/qa`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarId_QuestionsQa`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the

Name	Data Type	Required	Default Value	Documentation
				last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinarsByWebinarId_QuestionsQa` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email	string		<input type="checkbox"/>	Participant email
id	int64		<input type="checkbox"/>	Webinar ID
name	string		<input type="checkbox"/>	Participant display name
start_time	datetime		<input type="checkbox"/>	Webinar start time
uuid	guid		<input type="checkbox"/>	Webinar UUID

### 14.1.20 ReportWebinarsByWebinarId\_QuestionsQuestion\_detailsPolls

Retrieve webinar polls reportRetrieve ended webinar polls report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/polls`

Insert Zoom API URL: `/report/webinars/{webinarId}/polls`

Update Zoom API URL: `/report/webinars/{webinarId}/polls`

Delete Zoom API URL: `/report/webinars/{webinarId}/polls`

Field Selection Method: NotRequired

Base Path: `questions[*].question_details[*]`

Select Zoom API Operation: `get /report/webinars/{webinarId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarId_QuestionsQuestion_detailsPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with `select \* from table(name1 => value1, name3 => value3)` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinarsByWebinarId_QuestionsQuestion_detailsPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
answer	string		<input type="checkbox"/>	Given answer
email	string		<input type="checkbox"/>	Participant email
id	int64		<input type="checkbox"/>	Webinar ID
name	string		<input type="checkbox"/>	Participant display name
question	string		<input type="checkbox"/>	Asked question
start_time	datetime		<input type="checkbox"/>	Webinar start time
uuid	guid		<input type="checkbox"/>	Webinar UUID

### 14.1.21 ReportWebinarsByWebinarId\_QuestionsQuestion\_detailsQa

Retrieve webinar Q&A report Retrieve ended webinar Q&A report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/qa`

Insert Zoom API URL: `/report/webinars/{webinarId}/qa`

Update Zoom API URL: `/report/webinars/{webinarId}/qa`

Delete Zoom API URL: `/report/webinars/{webinarId}/qa`

Field Selection Method: NotRequired

Base Path: `questions[*].question_details[*]`

Select Zoom API Operation: `get /report/webinars/{webinarId}/qa`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarId_QuestionsQuestion_detailsQa`. A value must be provided at all

times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinarsByWebinarId_QuestionsQuestion_detailsQa` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
answer	string		<input type="checkbox"/>	Given answer
email	string		<input type="checkbox"/>	Participant email
id	int64		<input type="checkbox"/>	Webinar ID
name	string		<input type="checkbox"/>	Participant display name
question	string		<input type="checkbox"/>	Asked question
start_time	datetime		<input type="checkbox"/>	Webinar start time
uuid	guid		<input type="checkbox"/>	Webinar UUID

### 14.1.22 ReportWebinarsByWebinarIdParticipants

Retrieve webinar participants report Retrieve ended webinar participants report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/participants`

Insert Zoom API URL: `/report/webinars/{webinarId}/participants`

Update Zoom API URL: `/report/webinars/{webinarId}/participants`

Delete Zoom API URL: `/report/webinars/{webinarId}/participants`

Field Selection Method: NotRequired

Base Path: `participants[*]`

Select Zoom API Operation: `get /report/webinars/{webinarId}/participants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarIdParticipants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>next_page_token</code>	string	<input type="checkbox"/>		Next page token is used to paginate through large result sets. A next page token will be returned whenever the set of available results exceed the current page size. The expiration period for this token is 15 minutes.
<code>page_size</code>	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>webinarId</code>	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinarsByWebinarIdParticipants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>attentiveness_score</code>	string		<input type="checkbox"/>	Participant attentiveness score
<code>duration</code>	int64		<input type="checkbox"/>	Participant duration
<code>id</code>	guid		<input type="checkbox"/>	Participant UUID
<code>join_time</code>	datetime		<input type="checkbox"/>	Participant join time
<code>leave_time</code>	datetime		<input type="checkbox"/>	Participant leave time
<code>name</code>	string		<input type="checkbox"/>	Participant display name
<code>user_email</code>	string		<input type="checkbox"/>	Participant email
<code>user_id</code>	string		<input type="checkbox"/>	Participant ID

### 14.1.23 ReportWebinarsByWebinarIdPolls

Retrieve webinar polls report Retrieve ended webinar polls report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/polls`

Insert Zoom API URL: `/report/webinars/{webinarId}/polls`

Update Zoom API URL: `/report/webinars/{webinarId}/polls`

Delete Zoom API URL: `/report/webinars/{webinarId}/polls`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /report/webinars/{webinarId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarIdPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

## Table Function Columns

The columns of the table function `ReportWebinarsByWebinarIdPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	int64		<input type="checkbox"/>	Webinar ID
start_time	datetime		<input type="checkbox"/>	Webinar start time

Name	Data Type	Label	Required	Documentation
uuid	guid		<input type="checkbox"/>	Webinar UUID

#### 14.1.24 ReportWebinarsByWebinarIdQa

Retrieve webinar Q&A report  
Retrieve ended webinar Q&A report

Catalog: Zoom

Schema: Reports

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/report/webinars/{webinarId}/qa`

Insert Zoom API URL: `/report/webinars/{webinarId}/qa`

Update Zoom API URL: `/report/webinars/{webinarId}/qa`

Delete Zoom API URL: `/report/webinars/{webinarId}/qa`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /report/webinars/{webinarId}/qa`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `ReportWebinarsByWebinarIdQa`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	string	<input checked="" type="checkbox"/>		The webinar ID or webinar UUID. If given webinar ID, will take the last webinar instance.

### Table Function Columns

The columns of the table function `ReportWebinarsByWebinarIdQa` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	int64		<input type="checkbox"/>	Webinar ID
start_time	datetime		<input type="checkbox"/>	Webinar start time
uuid	guid		<input type="checkbox"/>	Webinar UUID

## 15 Schema: TrackingField

### 15.1 Tables

#### 15.1.1 deleteV2TrackingFieldsByFieldId

Delete a Tracking FieldDelete a Tracking Field on your Zoom account

Catalog: Zoom

Schema: TrackingField

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/v2/tracking_fields/{fieldId}`

Insert Zoom API URL: `/v2/tracking_fields/{fieldId}`

Update Zoom API URL: `/v2/tracking_fields/{fieldId}`

Delete Zoom API URL: `/v2/tracking_fields/{fieldId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /v2/tracking_fields/{fieldId}`

### Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteV2TrackingFieldsByFieldId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
fieldId	string	<input checked="" type="checkbox"/>		The Tracking Field ID

### Table Function Columns



The columns of the table function `deleteV2TrackingFieldsByFieldId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 15.1.2 patchV2TrackingFieldsByFieldId

Update a Tracking FieldUpdate a Tracking Field on your Zoom account

Catalog: Zoom

Schema: TrackingField

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/v2/tracking_fields/{fieldId}`

Insert Zoom API URL: `/v2/tracking_fields/{fieldId}`

Update Zoom API URL: `/v2/tracking_fields/{fieldId}`

Delete Zoom API URL: `/v2/tracking_fields/{fieldId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /v2/tracking_fields/{fieldId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchV2TrackingFieldsByFieldId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
fieldId	string	<input checked="" type="checkbox"/>		The Tracking Field ID

## Table Function Columns

The columns of the table function `patchV2TrackingFieldsByFieldId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 15.1.3 postV2TrackingFields

Create a Tracking Field on your Zoom account

Catalog: Zoom

Schema: TrackingField

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/v2/tracking_fields`

Insert Zoom API URL: `/v2/tracking_fields`

Update Zoom API URL: `/v2/tracking_fields`

Delete Zoom API URL: `/v2/tracking_fields`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /v2/tracking_fields`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postV2TrackingFields`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Tracking Field

## Table Function Columns

The columns of the table function `postV2TrackingFields` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
field	string		<input type="checkbox"/>	Tracking Field Name
id	string		<input type="checkbox"/>	Tracking Field ID
required	boolean		<input type="checkbox"/>	Tracking Field Required
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
visible	boolean		<input type="checkbox"/>	Tracking Field Visible

#### 15.1.4 V2TrackingFields

List Tracking Fields. List Tracking Fields on your Zoom account.

Catalog: Zoom

Schema: TrackingField

This is a read-only table. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/v2/tracking_fields`

Insert Zoom API URL: `/v2/tracking_fields`

Update Zoom API URL: `/v2/tracking_fields`

Delete Zoom API URL: `/v2/tracking_fields`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /v2/tracking_fields`

### Table Columns

The columns of the table `V2TrackingFields` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

#### 15.1.5 V2TrackingFieldsByFieldId

Retrieve a tracking field Retrieve a tracking field

Catalog: Zoom

Schema: TrackingField

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/v2/tracking_fields/{fieldId}`

Insert Zoom API URL: `/v2/tracking_fields/{fieldId}`

Update Zoom API URL: `/v2/tracking_fields/{fieldId}`

Delete Zoom API URL: `/v2/tracking_fields/{fieldId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /v2/tracking_fields/{fieldId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `V2TrackingFieldsByFieldId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
fieldId	string	<input checked="" type="checkbox"/>		The Tracking Field ID

## Table Function Columns

The columns of the table function `V2TrackingFieldsByFieldId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
field	string		<input type="checkbox"/>	Tracking Field Name
id	string		<input type="checkbox"/>	Tracking Field ID
required	boolean		<input type="checkbox"/>	Tracking Field Required
visible	boolean		<input type="checkbox"/>	Tracking Field Visible

## 16 Schema: TSP

### 16.1 Tables

#### 16.1.1 deleteUsersByUserIdTspTspId

Delete a user's TSP accountDelete a user's TSP account

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp/{tspId}`

Insert Zoom API URL: `/users/{userId}/tsp/{tspId}`

Update Zoom API URL: `/users/{userId}/tsp/{tspId}`

Delete Zoom API URL: `/users/{userId}/tsp/{tspId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /users/{userId}/tsp/{tspId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersByIdTspTspId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>tspId</code>	string	<input checked="" type="checkbox"/>		TSP account index
<code>userId</code>	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersByIdTspTspId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 16.1.2 patchTsp

Update account's TSP informationUpdate TSP information on account level

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/tsp`

Insert Zoom API URL: `/tsp`

Update Zoom API URL: `/tsp`

Delete Zoom API URL: `/tsp`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `patch /tsp`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchTsp`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		TSP Account

## Table Function Columns

The columns of the table function `patchTsp` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 16.1.3 patchUsersByUserIdTspTspId

Update a TSP accountUpdate a user's TSP account

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp/{tspId}`

Insert Zoom API URL: `/users/{userId}/tsp/{tspId}`

Update Zoom API URL: `/users/{userId}/tsp/{tspId}`

Delete Zoom API URL: `/users/{userId}/tsp/{tspId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /users/{userId}/tsp/{tspId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchUsersByUserIdTspTspId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		TSP Account
tspId	string	<input checked="" type="checkbox"/>		TSP account index
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `patchUsersByUserIdTspTspId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 16.1.4 postUsersByUserId\_Dial\_in\_numbersTsp

Add a user's TSP accountAdd a user's TSP account

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp`

Insert Zoom API URL: `/users/{userId}/tsp`

Update Zoom API URL: `/users/{userId}/tsp`

Delete Zoom API URL: `/users/{userId}/tsp`

Field Selection Method: NotRequired

Base Path: `dial_in_numbers[*]`

Select Zoom API Operation: `post /users/{userId}/tsp`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postUsersByUserId_Dial_in_numbersTsp`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		TSP Account
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `postUsersByUserId_Dial_in_numbersTsp` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
code	string(6)		<input type="checkbox"/>	Country Code
conference_code	string(16)		<input checked="" type="checkbox"/>	Conference code, numeric value, length is less than 16.
country_label	string(10)		<input type="checkbox"/>	Country Label, if passed, will display in place of code.
leader_pin	string(16)		<input checked="" type="checkbox"/>	Leader PIN, numeric value, length is less than 16.
number	string(16)		<input type="checkbox"/>	Dial-in number, length is less than 16.
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
type	string		<input type="checkbox"/>	Dial-in number type.



### 16.1.5 postUsersByUserIdTsp

Add a user's TSP accountAdd a user's TSP account

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp`

Insert Zoom API URL: `/users/{userId}/tsp`

Update Zoom API URL: `/users/{userId}/tsp`

Delete Zoom API URL: `/users/{userId}/tsp`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /users/{userId}/tsp`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postUsersByUserIdTsp`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		TSP Account
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `postUsersByUserIdTsp` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
conference_code	string(16)		<input checked="" type="checkbox"/>	Conference code, numeric value, length is less than 16.
leader_pin	string(16)		<input checked="" type="checkbox"/>	Leader PIN, numeric value, length is less than 16.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 16.1.6 Tsp

Retrieve account's TSP information Retrieve TSP information on account level

Catalog: Zoom

Schema: TSP

This is a read-only table. The Zoom API may not support changing the data or the Invantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/tsp`

Insert Zoom API URL: `/tsp`

Update Zoom API URL: `/tsp`

Delete Zoom API URL: `/tsp`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /tsp`

## Table Columns

The columns of the table `Tsp` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
enable	boolean		<input type="checkbox"/>	Enable 3rd party audio conferencing for account users
tsp_provider	string		<input type="checkbox"/>	3rd party audio conferencing provider

### 16.1.7 Tsp\_Dial\_in\_numbers

Retrieve account's TSP information Retrieve TSP information on account level

Catalog: Zoom

Schema: TSP

This is a read-only table. The Zoom API may not support changing the data or the Invantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/tsp`

Insert Zoom API URL: `/tsp`

Update Zoom API URL: `/tsp`

Delete Zoom API URL: `/tsp`

Field Selection Method: NotRequired

Base Path: `dial_in_numbers[*]`

Select Zoom API Operation: `get /tsp`

## Table Columns

The columns of the table `Tsp_Dial_in_numbers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
code	string		<input type="checkbox"/>	Country Code
enable	boolean		<input type="checkbox"/>	Enable 3rd party audio conferencing for account users
number	string		<input type="checkbox"/>	Dial-in number, length is less than 16
tsp_provider	string		<input type="checkbox"/>	3rd party audio conferencing provider
type	string		<input type="checkbox"/>	

### 16.1.8 UsersByUserId\_Tsp\_accountsDial\_in\_numbersTsp

List user's TSP accountsList user's TSP accounts

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp`

Insert Zoom API URL: `/users/{userId}/tsp`

Update Zoom API URL: `/users/{userId}/tsp`

Delete Zoom API URL: `/users/{userId}/tsp`

Field Selection Method: NotRequired

Base Path: `tsp_accounts[*].dial_in_numbers[*]`

Select Zoom API Operation: `get /users/{userId}/tsp`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserId_Tsp_accountsDial_in_numbersTsp`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four

parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserId_Tsp_accountsDial_in_numbersTsp` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
code	string(6)		<input type="checkbox"/>	Country Code
conference_code	string(16)		<input checked="" type="checkbox"/>	Conference code, numeric value, length is less than 16.
country_label	string(10)		<input type="checkbox"/>	Country Label, if passed, will display in place of code.
leader_pin	string(16)		<input checked="" type="checkbox"/>	Leader PIN, numeric value, length is less than 16.
number	string(16)		<input type="checkbox"/>	Dial-in number, length is less than 16.
type	string		<input type="checkbox"/>	Dial-in number type.

### 16.1.9 UsersByUserIdTsp

List user's TSP accounts

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp`

Insert Zoom API URL: `/users/{userId}/tsp`

Update Zoom API URL: `/users/{userId}/tsp`

Delete Zoom API URL: `/users/{userId}/tsp`

Field Selection Method: NotRequired

Base Path: `tsp_accounts[*]`

Select Zoom API Operation: `get /users/{userId}/tsp`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdTsp`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdTsp` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
conference_code	string(16)		<input checked="" type="checkbox"/>	Conference code, numeric value, length is less than 16.
leader_pin	string(16)		<input checked="" type="checkbox"/>	Leader PIN, numeric value, length is less than 16.

### 16.1.10 UsersByUserIdTsp\_Dial\_in\_numbersTspId

Retrieve a user's TSP account

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp/{tspId}`

Insert Zoom API URL: `/users/{userId}/tsp/{tspId}`

Update Zoom API URL: `/users/{userId}/tsp/{tspId}`

Delete Zoom API URL: `/users/{userId}/tsp/{tspId}`

Field Selection Method: NotRequired

Base Path: `dial_in_numbers[*]`

Select Zoom API Operation: `get /users/{userId}/tsp/{tspId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdTsp_Dial_in_numbersTspId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
tspId	string	<input checked="" type="checkbox"/>		TSP account index
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdTsp_Dial_in_numbersTspId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
code	string(6)		<input type="checkbox"/>	Country Code
conference_code	string(16)		<input checked="" type="checkbox"/>	Conference code, numeric value, length is less than 16.
country_label	string(10)		<input type="checkbox"/>	Country Label, if passed, will display in place of code.
leader_pin	string(16)		<input checked="" type="checkbox"/>	Leader PIN, numeric value, length is less than 16.
number	string(16)		<input type="checkbox"/>	Dial-in number, length is less than 16.
type	string		<input type="checkbox"/>	Dial-in number type.

### 16.1.11 UsersByUserIdTspTspId

Retrieve a user's TSP account

Catalog: Zoom

Schema: TSP

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/tsp/{tspId}`

Insert Zoom API URL: `/users/{userId}/tsp/{tspId}`

Update Zoom API URL: `/users/{userId}/tsp/{tspId}`

Delete Zoom API URL: `/users/{userId}/tsp/{tspId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /users/{userId}/tsp/{tspId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdTspTspId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>tspId</code>	string	<input checked="" type="checkbox"/>		TSP account index
<code>userId</code>	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdTspTspId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>conference_code</code>	string(16)		<input checked="" type="checkbox"/>	Conference code, numeric value, length is less than 16.
<code>leader_pin</code>	string(16)		<input checked="" type="checkbox"/>	Leader PIN, numeric value, length is less than 16.

## 17 Schema: Users

### 17.1 Tables

#### 17.1.1 `deleteUsersByUserId`

Delete a userDelete a user on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}`

Insert Zoom API URL: `/users/{userId}`

Update Zoom API URL: `/users/{userId}`

Delete Zoom API URL: `/users/{userId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `delete /users/{userId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersById`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
action	string	<input type="checkbox"/>	disassociate	Delete action type (Values: disassociate, delete)
transfer_email	string	<input type="checkbox"/>		Transfer email
transfer_meeting	boolean	<input type="checkbox"/>		Transfer meeting
transfer_recording	boolean	<input type="checkbox"/>		Transfer recording
transfer_webinar	boolean	<input type="checkbox"/>		Transfer webinar
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersById` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.



### 17.1.2 deleteUsersByUserIdAssistants

Delete a user's assistants>Delete all of a user's assistants

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/assistants`

Insert Zoom API URL: `/users/{userId}/assistants`

Update Zoom API URL: `/users/{userId}/assistants`

Delete Zoom API URL: `/users/{userId}/assistants`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /users/{userId}/assistants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersByUserIdAssistants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersByUserIdAssistants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.3 deleteUsersByUserIdAssistantsAssistantId

Delete a user's assistant>Delete one of a user's assistants

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/assistants/{assistantId}`

Insert Zoom API URL: `/users/{userId}/assistants/{assistantId}`

Update Zoom API URL: `/users/{userId}/assistants/{assistantId}`

Delete Zoom API URL: `/users/{userId}/assistants/{assistantId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /users/{userId}/assistants/{assistantId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersByUserIdAssistantsAssistantId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
assistantId	string	<input checked="" type="checkbox"/>		Assistant's ID
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersByUserIdAssistantsAssistantId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.4 deleteUsersByUserIdSchedulers

Delete a user's schedulersDelete all of a user'schedulers

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/schedulers`

Insert Zoom API URL: `/users/{userId}/schedulers`

Update Zoom API URL: `/users/{userId}/schedulers`

Delete Zoom API URL: `/users/{userId}/schedulers`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /users/{userId}/schedulers`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersByUserIdSchedulers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersByUserIdSchedulers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.5 deleteUsersByUserIdSchedulersSchedulerId

Delete a user's schedulerDelete one of a user's schedulers

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/schedulers/{schedulerId}`

Insert Zoom API URL: `/users/{userId}/schedulers/{schedulerId}`

Update Zoom API URL: `/users/{userId}/schedulers/{schedulerId}`

Delete Zoom API URL: `/users/{userId}/schedulers/{schedulerId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /users/{userId}/schedulers/{schedulerId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersByUserIdSchedulersSchedulerId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>schedulerId</code>	string	<input checked="" type="checkbox"/>		Scheduler's ID
<code>userId</code>	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersByUserIdSchedulersSchedulerId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.6 deleteUsersByUserIdToken

Revoke a user's SSO token

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/token`

Insert Zoom API URL: `/users/{userId}/token`

Update Zoom API URL: `/users/{userId}/token`

Delete Zoom API URL: `/users/{userId}/token`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /users/{userId}/token`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteUsersByUserIdToken`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `deleteUsersByUserIdToken` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.7 patchUsersByUserId

Update a userUpdate a user on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}`

Insert Zoom API URL: `/users/{userId}`

Update Zoom API URL: `/users/{userId}`

Delete Zoom API URL: `/users/{userId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /users/{userId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchUsersByUserId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		User
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `patchUsersByUserId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.8 patchUsersByUserIdSettings

Update a user's settings Update a user's settings

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/settings`

Insert Zoom API URL: `/users/{userId}/settings`

Update Zoom API URL: `/users/{userId}/settings`

Delete Zoom API URL: `/users/{userId}/settings`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /users/{userId}/settings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchUsersByUserIdSettings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		User Settings
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `patchUsersByUserIdSettings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

**17.1.9 postUsers**

Create a user  
Create a user on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users`

Insert Zoom API URL: `/users`

Update Zoom API URL: `/users`

Delete Zoom API URL: `/users`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /users`

**Table Function Columns**

The columns of the table function `postUsers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email	string		<input type="checkbox"/>	User's email address
first_name	string(64)		<input type="checkbox"/>	User's first name
id	string		<input type="checkbox"/>	User ID
last_name	string(64)		<input type="checkbox"/>	User's last name
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
type	int64		<input type="checkbox"/>	User's type

**17.1.10 postUsersByUserIdAssistants**

Add assistants  
Add assistants to a user

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/assistants`

Insert Zoom API URL: `/users/{userId}/assistants`

Update Zoom API URL: `/users/{userId}/assistants`

Delete Zoom API URL: `/users/{userId}/assistants`

Field Selection Method: NotRequired



Select Zoom API Operation: `post /users/{userId}/assistants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postUsersByUserIdAssistants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		User assistant
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `postUsersByUserIdAssistants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
add_at	datetime		<input type="checkbox"/>	
ids	string		<input type="checkbox"/>	User ID
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.11 `postUsersByUserIdPicture`

Upload a user's pictureUpload a user's profile picture

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/picture`

Insert Zoom API URL: `/users/{userId}/picture`

Update Zoom API URL: `/users/{userId}/picture`

Delete Zoom API URL: `/users/{userId}/picture`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /users/{userId}/picture`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postUsersByIdPicture`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
pic_file	byte[]	<input checked="" type="checkbox"/>		User picture file, must be a jpg/jpeg file
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `postUsersByIdPicture` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.12 putUsersByEmail

Update a user's email

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/email`

Insert Zoom API URL: `/users/{userId}/email`

Update Zoom API URL: `/users/{userId}/email`

Delete Zoom API URL: `/users/{userId}/email`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /users/{userId}/email`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putUsersByUserIdEmail`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `putUsersByUserIdEmail` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.13 putUsersByUserIdPassword

Update a user's passwordUpdate a user's password

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/password`

Insert Zoom API URL: `/users/{userId}/password`

Update Zoom API URL: `/users/{userId}/password`

Delete Zoom API URL: `/users/{userId}/password`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /users/{userId}/password`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putUsersByUserIdPassword`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `putUsersByUserIdPassword` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.14 putUsersByUserIdStatus

Update a user's statusUpdate a user's status

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/status`

Insert Zoom API URL: `/users/{userId}/status`

Update Zoom API URL: `/users/{userId}/status`

Delete Zoom API URL: `/users/{userId}/status`

Field Selection Method: NotRequired

Select Zoom API Operation: `put /users/{userId}/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putUsersByUserIdStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `putUsersByUserIdStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 17.1.15 Users

List UsersList users on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users`

Insert Zoom API URL: `/users`

Update Zoom API URL: `/users`

Delete Zoom API URL: `/users`

Field Selection Method: `NotRequired`

Base Path: `users[*]`

Select Zoom API Operation: `get /users`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Users`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
status	string	<input type="checkbox"/>	active	User status (Values: active, inactive, pending)

## Table Function Columns

The columns of the table function `Users` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
created_at	datetime		<input type="checkbox"/>	User create time
dept	string		<input type="checkbox"/>	Department
email	string		<input checked="" type="checkbox"/>	User's email address
first_name	string(64)		<input type="checkbox"/>	User's first name
id	string		<input type="checkbox"/>	User ID
last_client_version	string		<input type="checkbox"/>	User last login client version
last_login_time	datetime		<input type="checkbox"/>	User last login time
last_name	string(64)		<input type="checkbox"/>	User's last name
pmi	string		<input type="checkbox"/>	Personal Meeting ID
timezone	string		<input type="checkbox"/>	Time Zone
type	int64		<input checked="" type="checkbox"/>	User's type

### 17.1.16 Users\_Group\_idsByUserId

Retrieve a user  
Retrieve a user on your account

Catalog: Zoom

## Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}`

Insert Zoom API URL: `/users/{userId}`

Update Zoom API URL: `/users/{userId}`

Delete Zoom API URL: `/users/{userId}`

Field Selection Method: `NotRequired`

Base Path: `group_ids[*]`

Select Zoom API Operation: `get /users/{userId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Users_Group_idsByUserId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>login_type</code>	string	<input type="checkbox"/>		Values: 0, 1, 99, 100, 101.
<code>userId</code>	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `Users_Group_idsByUserId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>account_id</code>	string		<input type="checkbox"/>	
<code>cms_user_id</code>	string		<input type="checkbox"/>	
<code>created_at</code>	datetime		<input type="checkbox"/>	User create time
<code>dept</code>	string		<input type="checkbox"/>	Department
<code>email</code>	string		<input checked="" type="checkbox"/>	User's email address
<code>first_name</code>	string(64)		<input type="checkbox"/>	User's first name

Name	Data Type	Label	Required	Documentation
host_key	string		<input type="checkbox"/>	
id	string		<input type="checkbox"/>	User ID
language	string		<input type="checkbox"/>	
last_client_version	string		<input type="checkbox"/>	User last login client version
last_login_time	datetime		<input type="checkbox"/>	User last login time
last_name	string(64)		<input type="checkbox"/>	User's last name
personal_meeting_url	string		<input type="checkbox"/>	
pic_url	string		<input type="checkbox"/>	
pmi	string		<input type="checkbox"/>	Personal Meeting ID
timezone	string		<input type="checkbox"/>	Time Zone
type	int64		<input checked="" type="checkbox"/>	User's type
use_pmi	boolean		<input type="checkbox"/>	
value	string		<input type="checkbox"/>	
vanity_url	string		<input type="checkbox"/>	
verified	int64		<input type="checkbox"/>	

### 17.1.17 Users\_Im\_group\_idsByUserId

Retrieve a userRetrieve a user on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}`

Insert Zoom API URL: `/users/{userId}`

Update Zoom API URL: `/users/{userId}`

Delete Zoom API URL: `/users/{userId}`

Field Selection Method: NotRequired

Base Path: `im_group_ids[*]`

Select Zoom API Operation: `get /users/{userId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Users_Im_group_idsByUserId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four



parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
login_type	string	<input type="checkbox"/>		Values: 0, 1, 99, 100, 101.
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `Users_Im_group_idsByUserId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
account_id	string		<input type="checkbox"/>	
cms_user_id	string		<input type="checkbox"/>	
created_at	datetime		<input type="checkbox"/>	User create time
dept	string		<input type="checkbox"/>	Department
email	string		<input checked="" type="checkbox"/>	User's email address
first_name	string(64)		<input type="checkbox"/>	User's first name
host_key	string		<input type="checkbox"/>	
id	string		<input type="checkbox"/>	User ID
language	string		<input type="checkbox"/>	
last_client_version	string		<input type="checkbox"/>	User last login client version
last_login_time	datetime		<input type="checkbox"/>	User last login time
last_name	string(64)		<input type="checkbox"/>	User's last name
personal_meeting_url	string		<input type="checkbox"/>	
pic_url	string		<input type="checkbox"/>	
pmi	string		<input type="checkbox"/>	Personal Meeting ID
timezone	string		<input type="checkbox"/>	Time Zone
type	int64		<input checked="" type="checkbox"/>	User's type
use_pmi	boolean		<input type="checkbox"/>	
value	string		<input type="checkbox"/>	
vanity_url	string		<input type="checkbox"/>	
verified	int64		<input type="checkbox"/>	

### 17.1.18 Users\_UsersGroup\_ids

List UsersList users on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users`

Insert Zoom API URL: `/users`

Update Zoom API URL: `/users`

Delete Zoom API URL: `/users`

Field Selection Method: `NotRequired`

Base Path: `users[*].group_ids[*]`

Select Zoom API Operation: `get /users`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Users_UsersGroup_ids`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>page_number</code>	<code>int64</code>	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	<code>int64</code>	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>status</code>	<code>string</code>	<input type="checkbox"/>	<code>active</code>	User status (Values: <code>active</code> , <code>inactive</code> , <code>pending</code> )

## Table Function Columns

The columns of the table function `Users_UsersGroup_ids` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>created_at</code>	<code>datetime</code>		<input type="checkbox"/>	User create time
<code>dept</code>	<code>string</code>		<input type="checkbox"/>	Department
<code>email</code>	<code>string</code>		<input checked="" type="checkbox"/>	User's email address
<code>first_name</code>	<code>string(64)</code>		<input type="checkbox"/>	User's first name

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	User ID
last_client_version	string		<input type="checkbox"/>	User last login client version
last_login_time	datetime		<input type="checkbox"/>	User last login time
last_name	string(64)		<input type="checkbox"/>	User's last name
pmi	string		<input type="checkbox"/>	Personal Meeting ID
timezone	string		<input type="checkbox"/>	Time Zone
type	int64		<input checked="" type="checkbox"/>	User's type
value	string		<input type="checkbox"/>	

### 17.1.19 Users\_UsersIm\_group\_ids

List UsersList users on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users`

Insert Zoom API URL: `/users`

Update Zoom API URL: `/users`

Delete Zoom API URL: `/users`

Field Selection Method: `NotRequired`

Base Path: `users[*].im_group_ids[*]`

Select Zoom API Operation: `get /users`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `Users_UsersIm_group_ids`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
status	string	<input type="checkbox"/>	active	User status (Values: active, inactive, pending)

## Table Function Columns

The columns of the table function `Users_UsersIm_group_ids` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
created_at	datetime		<input type="checkbox"/>	User create time
dept	string		<input type="checkbox"/>	Department
email	string		<input checked="" type="checkbox"/>	User's email address
first_name	string(64)		<input type="checkbox"/>	User's first name
id	string		<input type="checkbox"/>	User ID
last_client_version	string		<input type="checkbox"/>	User last login client version
last_login_time	datetime		<input type="checkbox"/>	User last login time
last_name	string(64)		<input type="checkbox"/>	User's last name
pmi	string		<input type="checkbox"/>	Personal Meeting ID
timezone	string		<input type="checkbox"/>	Time Zone
type	int64		<input checked="" type="checkbox"/>	User's type
value	string		<input type="checkbox"/>	

### 17.1.20 UsersByUserId

Retrieve a user on your account

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}`

Insert Zoom API URL: `/users/{userId}`

Update Zoom API URL: `/users/{userId}`

Delete Zoom API URL: `/users/{userId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /users/{userId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
login_type	string	<input type="checkbox"/>		Values: 0, 1, 99, 100, 101.
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
account_id	string		<input type="checkbox"/>	
cms_user_id	string		<input type="checkbox"/>	
created_at	datetime		<input type="checkbox"/>	User create time
dept	string		<input type="checkbox"/>	Department
email	string		<input checked="" type="checkbox"/>	User's email address
first_name	string(64)		<input type="checkbox"/>	User's first name
host_key	string		<input type="checkbox"/>	
id	string		<input type="checkbox"/>	User ID
language	string		<input type="checkbox"/>	
last_client_version	string		<input type="checkbox"/>	User last login client version
last_login_time	datetime		<input type="checkbox"/>	User last login time
last_name	string(64)		<input type="checkbox"/>	User's last name
personal_meeting_url	string		<input type="checkbox"/>	
pic_url	string		<input type="checkbox"/>	
pmi	string		<input type="checkbox"/>	Personal Meeting ID
timezone	string		<input type="checkbox"/>	Time Zone
type	int64		<input checked="" type="checkbox"/>	User's type
use_pmi	boolean		<input type="checkbox"/>	
vanity_url	string		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
verified	int64		<input type="checkbox"/>	

### 17.1.21 UsersByUserIdAssistants

List a user's assistants

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/assistants`

Insert Zoom API URL: `/users/{userId}/assistants`

Update Zoom API URL: `/users/{userId}/assistants`

Delete Zoom API URL: `/users/{userId}/assistants`

Field Selection Method: NotRequired

Base Path: `assistants[*]`

Select Zoom API Operation: `get /users/{userId}/assistants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdAssistants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdAssistants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email	string		<input type="checkbox"/>	User email address. Must have id or email, if given id, the email is ignored.
id	string		<input type="checkbox"/>	User ID

### 17.1.22 UsersByUserIdPermissions

Retrieve a user's permissions

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/permissions`

Insert Zoom API URL: `/users/{userId}/permissions`

Update Zoom API URL: `/users/{userId}/permissions`

Delete Zoom API URL: `/users/{userId}/permissions`

Field Selection Method: NotRequired

Base Path: `permissions[*]`

Select Zoom API Operation: `get /users/{userId}/permissions`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdPermissions`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdPermissions` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
TEXT	string		<input type="checkbox"/>	

### 17.1.23 UsersByUserIdSchedulers

List a user's schedulers

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/schedulers`

Insert Zoom API URL: `/users/{userId}/schedulers`

Update Zoom API URL: `/users/{userId}/schedulers`

Delete Zoom API URL: `/users/{userId}/schedulers`

Field Selection Method: NotRequired

Base Path: `assistants[*]`

Select Zoom API Operation: `get /users/{userId}/schedulers`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdSchedulers`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns



The columns of the table function `UsersByUserIdSchedulers` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
email	string		<input type="checkbox"/>	User email address.
id	string		<input type="checkbox"/>	User ID

#### 17.1.24 UsersByUserIdSettings

Retrieve a user's settings

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/settings`

Insert Zoom API URL: `/users/{userId}/settings`

Update Zoom API URL: `/users/{userId}/settings`

Delete Zoom API URL: `/users/{userId}/settings`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /users/{userId}/settings`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdSettings`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
login_type	string	<input type="checkbox"/>		Values: 0, 1, 99, 100, 101.
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdSettings` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>email_notification_alternative_host_reminder</code>	boolean		<input type="checkbox"/>	When an alternative host is set or removed from a meeting
<code>email_notification_cancel_meeting_reminder</code>	boolean		<input type="checkbox"/>	When a meeting is cancelled
<code>email_notification_jbh_reminder</code>	boolean		<input type="checkbox"/>	When attendees join meeting before host
<code>feature_cn_meeting</code>	boolean		<input type="checkbox"/>	CN meeting feature
<code>feature_in_meeting</code>	boolean		<input type="checkbox"/>	IN meeting feature
<code>feature_large_meeting_capacity</code>	int64		<input type="checkbox"/>	Large meeting capacity, can be 100, 200, 300 or 500, depends on if having related large meeting capacity plan subscription or not
<code>feature_large_meeting</code>	boolean		<input type="checkbox"/>	Large meeting feature
<code>feature_meeting_capacity</code>	int64		<input type="checkbox"/>	User's meeting capacity
<code>feature_webinar_capacity</code>	int64		<input type="checkbox"/>	Webinar capacity, can be 100, 500, 1000, 3000, 5000 or 10000, depends on if having related webinar capacity plan subscription or not
<code>feature_webinar</code>	boolean		<input type="checkbox"/>	Webinar feature
<code>in_meeting_allow_live_streaming</code>	boolean		<input type="checkbox"/>	Allow live streaming
<code>in_meeting_annotation</code>	boolean		<input type="checkbox"/>	Annotation
<code>in_meeting_attendee_on_hold</code>	boolean		<input type="checkbox"/>	Allow host to put attendee on hold
<code>in_meeting_attention_tracking</code>	boolean		<input type="checkbox"/>	Attention tracking
<code>in_meeting_auto_saving_chat</code>	boolean		<input type="checkbox"/>	Auto saving chats
<code>in_meeting_breakout_room</code>	boolean		<input type="checkbox"/>	Breakout room
<code>in_meeting_chat</code>	boolean		<input type="checkbox"/>	Chat
<code>in_meeting_closed_caption</code>	boolean		<input type="checkbox"/>	Closed caption
<code>in_meeting_co_host</code>	boolean		<input type="checkbox"/>	Co-host
<code>in_meeting_custom_live_streaming</code>	boolean		<input type="checkbox"/>	Custom live streaming
<code>in_meeting_custom_service_instructions</code>	string		<input type="checkbox"/>	Custom service instructions
<code>in_meeting_e2e_encryption</code>	boolean		<input type="checkbox"/>	End-to-end encryption
<code>in_meeting_entry_exit_chime</code>	string		<input type="checkbox"/>	Play sound on join/leave
<code>in_meeting_far_end_camera_control</code>	boolean		<input type="checkbox"/>	Far end camera control
<code>in_meeting_feedback</code>	boolean		<input type="checkbox"/>	Feedback to Zoom
<code>in_meeting_file_transfer</code>	boolean		<input type="checkbox"/>	File transfer
<code>in_meeting_group_hd</code>	boolean		<input type="checkbox"/>	Group HD video
<code>in_meeting_non_verbal_feedback</code>	boolean		<input type="checkbox"/>	Non-verbal feedback
<code>in_meeting_polling</code>	boolean		<input type="checkbox"/>	Polling
<code>in_meeting_private_chat</code>	boolean		<input type="checkbox"/>	Private chat

Name	Data Type	Label	Required	Documentation
in_meeting_record_play_voice	boolean		<input type="checkbox"/>	Record and play their own voice
in_meeting_remote_control	boolean		<input type="checkbox"/>	Remote control
in_meeting_remote_support	boolean		<input type="checkbox"/>	Remote support
in_meeting_share_dual_camera	boolean		<input type="checkbox"/>	Share dual camera (Deprecated)
in_meeting_virtual_background	boolean		<input type="checkbox"/>	Virtual background
in_meeting_waiting_room	boolean		<input type="checkbox"/>	Waiting room
in_meeting_workplace_by_facebook	boolean		<input type="checkbox"/>	Workplace by facebook
recording_auto_delete_cmnr_days	int64		<input type="checkbox"/>	A specified number of days of auto delete cloud recordings
recording_auto_delete_cmnr	boolean		<input type="checkbox"/>	Auto delete cloud recordings
recording_auto_recording	string		<input type="checkbox"/>	Automatic recording
recording_cloud_recording	boolean		<input type="checkbox"/>	Cloud recording
recording_local_recording	boolean		<input type="checkbox"/>	Local recording
recording_record_audio_file	boolean		<input type="checkbox"/>	Record an audio only file
recording_record_gallery_view	boolean		<input type="checkbox"/>	Record the gallery view
recording_record_speaker_view	boolean		<input type="checkbox"/>	Record the active speaker view
recording_recording_audio_transcript	boolean		<input type="checkbox"/>	Audio transcript
recording_save_chat_text	boolean		<input type="checkbox"/>	Save chat text from the meeting
recording_show_timestamp	boolean		<input type="checkbox"/>	Show timestamp on video
schedule_meeting_audio_type	string		<input type="checkbox"/>	Determine how participants can join the audio portion of the meeting
schedule_meeting_force_pmi_jbh_password	boolean		<input type="checkbox"/>	Require a password for Personal Meetings if attendees can join before host
schedule_meeting_host_video	boolean		<input type="checkbox"/>	Host video
schedule_meeting_join_before_host	boolean		<input type="checkbox"/>	Join before host
schedule_meeting_participants_video	boolean		<input type="checkbox"/>	Participants video
schedule_meeting_pstn_password_protected	boolean		<input type="checkbox"/>	Generate and require password for participants joining by phone
telephony_audio_conference_info	string(2048)		<input type="checkbox"/>	3rd party audio conference info
telephony_show_international_numbers_link	boolean		<input type="checkbox"/>	Show international numbers link on the invitation email
telephony_third_party_audio	boolean		<input type="checkbox"/>	3rd party audio conference

### 17.1.25 UsersByUserIdToken

Retrieve a user's token

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/token`

Insert Zoom API URL: `/users/{userId}/token`

Update Zoom API URL: `/users/{userId}/token`

Delete Zoom API URL: `/users/{userId}/token`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /users/{userId}/token`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByIdToken`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>type</code>	string	<input type="checkbox"/>		User token type (Values: token, zpk, zak)
<code>userId</code>	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByIdToken` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>token</code>	string		<input type="checkbox"/>	User ID

### 17.1.26 UsersEmail

Check a user's emailCheck if the user email exists

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/email`

Insert Zoom API URL: `/users/email`

Update Zoom API URL: `/users/email`

Delete Zoom API URL: `/users/email`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /users/email`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersEmail`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
email	string	<input checked="" type="checkbox"/>		Zoom work email

## Table Function Columns

The columns of the table function `UsersEmail` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
existed_email	boolean		<input type="checkbox"/>	

### 17.1.27 UsersVanityName

Check a user's personal meeting room name  
Check if the user's personal meeting room name exists

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/vanity_name`

Insert Zoom API URL: `/users/vanity_name`

Update Zoom API URL: `/users/vanity_name`

Delete Zoom API URL: `/users/vanity_name`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /users/vanity_name`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersVanityName`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>vanity_name</code>	string	<input checked="" type="checkbox"/>		Personal meeting room name

## Table Function Columns

The columns of the table function `UsersVanityName` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>existed</code>	boolean		<input type="checkbox"/>	

### 17.1.28 UsersZpk

Verify a user's zpk (DeprecatedCheck if the zpk is expired. The zpk is used to authenticate a user.

Catalog: Zoom

Schema: Users

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/zpk`

Insert Zoom API URL: `/users/zpk`

Update Zoom API URL: `/users/zpk`

Delete Zoom API URL: `/users/zpk`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /users/zpk`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersZpk`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
zpk	string	<input checked="" type="checkbox"/>		User zpk

## Table Function Columns

The columns of the table function `UsersZpk` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
expire_in	int64		<input type="checkbox"/>	

## 18 Schema: Webhooks

### 18.1 Tables

#### 18.1.1 `deleteWebhooksByWebhookId`

Delete a webhookDelete a webhook

Catalog: Zoom

Schema: Webhooks

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webhooks/{webhookId}`

Insert Zoom API URL: `/webhooks/{webhookId}`

Update Zoom API URL: `/webhooks/{webhookId}`

Delete Zoom API URL: `/webhooks/{webhookId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `delete /webhooks/{webhookId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteWebhooksByWebhookId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>webhookId</code>	string	<input checked="" type="checkbox"/>		The webhook ID

## Table Function Columns

The columns of the table function `deleteWebhooksByWebhookId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 18.1.2 `patchWebhooksByWebhookId`

Update a webhookUpdate a webhook

Catalog: Zoom

Schema: Webhooks



This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webhooks/{webhookId}`

Insert Zoom API URL: `/webhooks/{webhookId}`

Update Zoom API URL: `/webhooks/{webhookId}`

Delete Zoom API URL: `/webhooks/{webhookId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `patch /webhooks/{webhookId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchWebhooksByWebhookId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Webhook
webhookId	string	<input checked="" type="checkbox"/>		The webhook ID

## Table Function Columns

The columns of the table function `patchWebhooksByWebhookId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 18.1.3 patchWebhooksOptions

Switch webhook versionSwitch webhook version

Catalog: Zoom

Schema: Webhooks

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webhooks/options`

Insert Zoom API URL: `/webhooks/options`

Update Zoom API URL: `/webhooks/options`

Delete Zoom API URL: `/webhooks/options`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `patch /webhooks/options`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchWebhooksOptions`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		

## Table Function Columns

The columns of the table function `patchWebhooksOptions` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 18.1.4 postWebhooks

Create a webhookCreate a webhook for a account

Catalog: Zoom

Schema: Webhooks

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webhooks`

Insert Zoom API URL: `/webhooks`

Update Zoom API URL: `/webhooks`

Delete Zoom API URL: `/webhooks`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `post /webhooks`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postWebhooks`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Webhook

## Table Function Columns

The columns of the table function `postWebhooks` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
auth_password	string(64)		<input checked="" type="checkbox"/>	Webhook auth password
auth_user	string(128)		<input checked="" type="checkbox"/>	Webhook auth user name
created_at	datetime		<input type="checkbox"/>	Webhook create time
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
url	string(256)		<input checked="" type="checkbox"/>	Webhook endpoint
webhook_id	string		<input type="checkbox"/>	Webhook Id

### 18.1.5 Webhooks

List webhooks List webhooks for a account

Catalog: Zoom

Schema: Webhooks

This is a read-only table. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webhooks`

Insert Zoom API URL: `/webhooks`

Update Zoom API URL: `/webhooks`

Delete Zoom API URL: `/webhooks`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /webhooks`

## Table Columns

The columns of the table `webhooks` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>total_records</code>	<code>int64</code>		<input type="checkbox"/>	The number of all records available across pages

### 18.1.6 WebhooksByWebhookId

Retrieve a webhook Retrieve a webhook

Catalog: Zoom

Schema: Webhooks

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webhooks/{webhookId}`

Insert Zoom API URL: `/webhooks/{webhookId}`

Update Zoom API URL: `/webhooks/{webhookId}`

Delete Zoom API URL: `/webhooks/{webhookId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /webhooks/{webhookId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `WebhooksByWebhookId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a

pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webhookId	string	<input checked="" type="checkbox"/>		The w ebhook ID

## Table Function Columns

The columns of the table function `WebhooksByWebhookId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
auth_passw ord	string(64)		<input checked="" type="checkbox"/>	Webhook auth passw ord
auth_user	string(128)		<input checked="" type="checkbox"/>	Webhook auth user name
created_at	datetime		<input type="checkbox"/>	Webhook create time
url	string(256)		<input checked="" type="checkbox"/>	Webhook endpoint
webhook_id	string		<input type="checkbox"/>	Webhook Id

## 19 Schema: Webinars

### 19.1 Tables

#### 19.1.1 deleteWebinarsByWebinarId

Delete a webinarDelete a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}`

Insert Zoom API URL: `/webinars/{webinarId}`

Update Zoom API URL: `/webinars/{webinarId}`

Delete Zoom API URL: `/webinars/{webinarId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /webinars/{webinarId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteWebinarsByWebinarId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>occurrence_id</code>	string	<input type="checkbox"/>		The meeting occurrence ID
<code>webinarId</code>	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `deleteWebinarsByWebinarId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.2 deleteWebinarsByWebinarIdPanelists

Remove a webinar's panelists Remove all panelists from a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/panelists`

Insert Zoom API URL: `/webinars/{webinarId}/panelists`

Update Zoom API URL: `/webinars/{webinarId}/panelists`

Delete Zoom API URL: `/webinars/{webinarId}/panelists`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /webinars/{webinarId}/panelists`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteWebinarsByWebinarIdPanelists`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `deleteWebinarsByWebinarIdPanelists` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.3 deleteWebinarsByWebinarIdPanelistsPanelistId

Remove a webinar panelist Remove a panelist from a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/panelists/{panelistId}`

Insert Zoom API URL: `/webinars/{webinarId}/panelists/{panelistId}`

Update Zoom API URL: `/webinars/{webinarId}/panelists/{panelistId}`

Delete Zoom API URL: `/webinars/{webinarId}/panelists/{panelistId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /webinars/{webinarId}/panelists/{panelistId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteWebinarsByWebinarIdPanelistsPanelistId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
panelistId	int64	<input checked="" type="checkbox"/>		The panelist ID
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `deleteWebinarsByWebinarIdPanelistsPanelistId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.4 deleteWebinarsByWebinarIdPollsPollId

Delete a webinar's Poll>Delete a webinar's Poll

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Insert Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Update Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Delete Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `delete /webinars/{webinarId}/polls/{pollId}`



## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `deleteWebinarsByWebinarIdPollsPollId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
pollId	string	<input checked="" type="checkbox"/>		The poll ID
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `deleteWebinarsByWebinarIdPollsPollId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.5 PastWebinarsByWebinarIdInstances

List of ended webinar instances

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/past_webinars/{webinarId}/instances`

Insert Zoom API URL: `/past_webinars/{webinarId}/instances`

Update Zoom API URL: `/past_webinars/{webinarId}/instances`

Delete Zoom API URL: `/past_webinars/{webinarId}/instances`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /past_webinars/{webinarId}/instances`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `PastWebinarsByWebinarIdInstances`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `PastWebinarsByWebinarIdInstances` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
DUMMY	string(1)		<input checked="" type="checkbox"/>	Default column added since the specification specifies that no data is returned.

### 19.1.6 patchWebinarsByWebinarId

Update a webinarUpdate a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}`

Insert Zoom API URL: `/webinars/{webinarId}`

Update Zoom API URL: `/webinars/{webinarId}`

Delete Zoom API URL: `/webinars/{webinarId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `patch /webinars/{webinarId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `patchWebinarsByWebinarId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Webinar
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `patchWebinarsByWebinarId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.7 postUsersByUserIdWebinars

Create a webinar  
 Create a webinar for a user <aside>The expiration time of start\_url is two hours. But for API users, the expiration time is 90 days.</aside>

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/webinars`

Insert Zoom API URL: `/users/{userId}/webinars`

Update Zoom API URL: `/users/{userId}/webinars`

Delete Zoom API URL: `/users/{userId}/webinars`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /users/{userId}/webinars`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postUsersByUserIdWebinars`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		User
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `postUsersByUserIdWebinars` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
agenda	string		<input type="checkbox"/>	Webinar agenda
created_at	datetime		<input type="checkbox"/>	Create time
duration	int64		<input type="checkbox"/>	Webinar duration
host_id	string		<input type="checkbox"/>	ID of the user set as host of Webinar
id	string		<input type="checkbox"/>	Webinar ID, also known as Webinar number
join_url	string		<input type="checkbox"/>	Join url
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
settings_allow_multiple_devices	boolean		<input type="checkbox"/>	Allow attendees to join from multiple devices
settings_alternative_hosts	string		<input type="checkbox"/>	Alternative hosts emails or IDs. Multiple values separated by comma.
settings_approval_type	int64		<input type="checkbox"/>	
settings_audio	string		<input type="checkbox"/>	Determine how participants can join the audio portion of the meeting
settings_auto_recording	string		<input type="checkbox"/>	
settings_close_registration	boolean		<input type="checkbox"/>	Close registration after event date

Name	Data Type	Label	Required	Documentation
settings_enforce_login_domains	string		<input type="checkbox"/>	Only signed-in users with specified domains can join meetings
settings_enforce_login	boolean		<input type="checkbox"/>	Only signed-in users can join this meeting
settings_hd_video	boolean		<input type="checkbox"/>	Default to HD Video
settings_host_video	boolean		<input type="checkbox"/>	Start video when host joins webinar
settings_on_demand	boolean		<input type="checkbox"/>	Make the webinar on-demand
settings_panelists_video	boolean		<input type="checkbox"/>	Start video when panelists join webinar
settings_practice_session	boolean		<input type="checkbox"/>	Enable Practice Session
settings_registration_type	int64		<input type="checkbox"/>	Registration type. Used for recurring webinar with fixed time only.
settings_show_share_button	boolean		<input type="checkbox"/>	Show social share buttons on registration page
start_time	datetime		<input type="checkbox"/>	Webinar start time
start_url	string		<input type="checkbox"/>	Start url
timezone	string		<input type="checkbox"/>	Timezone to format start_time
topic	string		<input type="checkbox"/>	Webinar topic
type	int64		<input type="checkbox"/>	Webinar Type
uuid	string		<input type="checkbox"/>	Webinar unique ID

### 19.1.8 postWebinarsByWebinarIdPanelists

Add a webinar panelistAdd panelist to webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/panelists`

Insert Zoom API URL: `/webinars/{webinarId}/panelists`

Update Zoom API URL: `/webinars/{webinarId}/panelists`

Delete Zoom API URL: `/webinars/{webinarId}/panelists`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /webinars/{webinarId}/panelists`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postWebinarsByWebinarIdPanelists`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default

to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `postWebinarsByWebinarIdPanelists` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Panelist ID
join_url	string		<input type="checkbox"/>	Join URL for this panelist
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.9 postWebinarsByWebinarIdPolls

Create a webinar's poll  
Create a poll for a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/polls`

Insert Zoom API URL: `/webinars/{webinarId}/polls`

Update Zoom API URL: `/webinars/{webinarId}/polls`

Delete Zoom API URL: `/webinars/{webinarId}/polls`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /webinars/{webinarId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postWebinarsByWebinarIdPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Webinar poll object
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `postWebinarsByWebinarIdPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Webinar Poll ID
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
status	string		<input type="checkbox"/>	Status of the Webinar Poll
title	string		<input type="checkbox"/>	Poll Title

### 19.1.10 postWebinarsByWebinarIdRegistrants

Add a webinar registrantAdd a registrant for a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invasive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/registrants`

Insert Zoom API URL: `/webinars/{webinarId}/registrants`

Update Zoom API URL: `/webinars/{webinarId}/registrants`

Delete Zoom API URL: `/webinars/{webinarId}/registrants`

Field Selection Method: NotRequired

Select Zoom API Operation: `post /webinars/{webinarId}/registrants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `postWebinarsByWebinarIdRegistrants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
occurrence_ids	string	<input type="checkbox"/>		Occurrence IDs, could get this value from Webinar Get API. Multiple value separated by comma.
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `postWebinarsByWebinarIdRegistrants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
id	string		<input type="checkbox"/>	Registrant ID
join_url	string		<input type="checkbox"/>	Join URL for this registrant
registrant_id	string		<input type="checkbox"/>	Registrant ID
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.
start_time	datetime		<input type="checkbox"/>	Start time
topic	string		<input type="checkbox"/>	Topic

### 19.1.11 putWebinarsByWebinarIdPollsPollId

Update a webinar's pollUpdate a webinar's poll

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`



Insert Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Update Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Delete Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `put /webinars/{webinarId}/polls/{pollId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putWebinarsByWebinarIdPollsPollId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		Webinar Poll
pollId	string	<input checked="" type="checkbox"/>		The poll ID
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `putWebinarsByWebinarIdPollsPollId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.12 putWebinarsByWebinarIdRegistrantsStatus

Update a webinar registrant's statusUpdate a webinar registrant's status

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/registrants/status`

Insert Zoom API URL: `/webinars/{webinarId}/registrants/status`

Update Zoom API URL: `/webinars/{webinarId}/registrants/status`

Delete Zoom API URL: `/webinars/{webinarId}/registrants/status`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `put /webinars/{webinarId}/registrants/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putWebinarsByWebinarIdRegistrantsStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
occurrence_id	string	<input type="checkbox"/>		The meeting occurrence ID
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `putWebinarsByWebinarIdRegistrantsStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.13 putWebinarsByWebinarIdStatus

Update a webinar's status

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/status`

Insert Zoom API URL: `/webinars/{webinarId}/status`

Update Zoom API URL: `/webinars/{webinarId}/status`

Delete Zoom API URL: `/webinars/{webinarId}/status`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `put /webinars/{webinarId}/status`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `putWebinarsByWebinarIdStatus`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
body	string	<input checked="" type="checkbox"/>		
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `putWebinarsByWebinarIdStatus` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
RESULT	string		<input type="checkbox"/>	Outcome of operation as single plain text column.

### 19.1.14 UsersByUserIdWebinars

List webinarsList webinars for a user

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/users/{userId}/webinars`

Insert Zoom API URL: `/users/{userId}/webinars`

Update Zoom API URL: `/users/{userId}/webinars`

Delete Zoom API URL: `/users/{userId}/webinars`

Field Selection Method: `NotRequired`

Base Path: `webinars[*]`

Select Zoom API Operation: `get /users/{userId}/webinars`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `UsersByUserIdWebinars`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
page_number	int64	<input type="checkbox"/>	1	Current page number of returned records
page_size	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
userId	string	<input checked="" type="checkbox"/>		The user ID or email address

## Table Function Columns

The columns of the table function `UsersByUserIdWebinars` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
created_at	datetime		<input type="checkbox"/>	Create time
duration	int64		<input type="checkbox"/>	Meeting duration
host_id	string		<input type="checkbox"/>	ID of the user set as host of webinar
id	string		<input type="checkbox"/>	Webinar ID, also known as webinar number
join_url	string		<input type="checkbox"/>	Join url
timezone	string		<input type="checkbox"/>	Timezone to format start_time
topic	string		<input type="checkbox"/>	Meeting topic
type	int64		<input type="checkbox"/>	Meeting Type
uuid	string		<input type="checkbox"/>	Webinar unique ID

### 19.1.15 WebinarsByWebinarId

Retrieve a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}`

Insert Zoom API URL: `/webinars/{webinarId}`

Update Zoom API URL: `/webinars/{webinarId}`

Delete Zoom API URL: `/webinars/{webinarId}`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /webinars/{webinarId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `WebinarsByWebinarId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>webinarId</code>	<code>int64</code>	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `WebinarsByWebinarId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>agenda</code>	<code>string</code>		<input type="checkbox"/>	Webinar agenda
<code>created_at</code>	<code>datetime</code>		<input type="checkbox"/>	Create time
<code>duration</code>	<code>int64</code>		<input type="checkbox"/>	Webinar duration

Name	Data Type	Label	Required	Documentation
host_id	string		<input type="checkbox"/>	ID of the user set as host of webinar
id	string		<input type="checkbox"/>	Webinar ID, also know as webinar number
join_url	string		<input type="checkbox"/>	Join url
settings_allow_multiple_devices	boolean		<input type="checkbox"/>	Allow attendees to join from multiple devices
settings_alternative_hosts	string		<input type="checkbox"/>	Alternative hosts emails or IDs. Multiple values separated by comma.
settings_approval_type	int64		<input type="checkbox"/>	
settings_audio	string		<input type="checkbox"/>	Determine how participants can join the audio portion of the meeting
settings_auto_recording	string		<input type="checkbox"/>	
settings_close_registration	boolean		<input type="checkbox"/>	Close registration after event date
settings_enforce_login_domains	string		<input type="checkbox"/>	Only signed-in users with specified domains can join meetings
settings_enforce_login	boolean		<input type="checkbox"/>	Only signed-in users can join this meeting
settings_hd_video	boolean		<input type="checkbox"/>	Default to HD Video
settings_host_video	boolean		<input type="checkbox"/>	Start video when host joins webinar
settings_on_demand	boolean		<input type="checkbox"/>	Make the webinar on-demand
settings_panelists_video	boolean		<input type="checkbox"/>	Start video when panelists join webinar
settings_practice_session	boolean		<input type="checkbox"/>	Enable Practice Session
settings_registration_type	int64		<input type="checkbox"/>	Registration type. Used for recurring webinar with fixed time only.
settings_show_share_button	boolean		<input type="checkbox"/>	Show social share buttons on registration page
start_time	datetime		<input type="checkbox"/>	Webinar start time
start_url	string		<input type="checkbox"/>	Start url
timezone	string		<input type="checkbox"/>	Timezone to format start_time
topic	string		<input type="checkbox"/>	Webinar topic
type	int64		<input type="checkbox"/>	Webinar Type
uuid	string		<input type="checkbox"/>	Webinar unique ID

### 19.1.16 WebinarsByWebinarIdPanelists

List a webinar's panelists  
List panelists for a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/panelists`

Insert Zoom API URL: `/webinars/{webinarId}/panelists`

Update Zoom API URL: `/webinars/{webinarId}/panelists`

Delete Zoom API URL: `/webinars/{webinarId}/panelists`

Field Selection Method: `NotRequired`

Select Zoom API Operation: `get /webinars/{webinarId}/panelists`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `WebinarsByWebinarIdPanelists`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>webinarId</code>	<code>int64</code>	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `WebinarsByWebinarIdPanelists` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>total_records</code>	<code>int64</code>		<input type="checkbox"/>	Total records

### 19.1.17 WebinarsByWebinarIdPolls

List a webinar's polls List polls of a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the In-vantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/polls`

Insert Zoom API URL: `/webinars/{webinarId}/polls`

Update Zoom API URL: `/webinars/{webinarId}/polls`

Delete Zoom API URL: `/webinars/{webinarId}/polls`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /webinars/{webinarId}/polls`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `WebinarsByWebinarIdPolls`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
webinarId	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `WebinarsByWebinarIdPolls` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
total_records	int64		<input type="checkbox"/>	The number of all records available across pages

### 19.1.18 WebinarsByWebinarIdPollsPollId

Retrieve a webinar's poll

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`



Insert Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Update Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Delete Zoom API URL: `/webinars/{webinarId}/polls/{pollId}`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /webinars/{webinarId}/polls/{pollId}`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `WebinarsByWebinarIdPollsPollId`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>pollId</code>	string	<input checked="" type="checkbox"/>		The poll ID
<code>webinarId</code>	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `WebinarsByWebinarIdPollsPollId` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>id</code>	string		<input type="checkbox"/>	Webinar Poll ID
<code>status</code>	string		<input type="checkbox"/>	Status of the Webinar Poll
<code>title</code>	string		<input type="checkbox"/>	Poll Title

### 19.1.19 WebinarsByWebinarIdRegistrants

List a webinar's registrantsList registrants for a webinar

Catalog: Zoom

Schema: Webinars

This is a read-only table function. The Zoom API may not support changing the data or the Invariantive SQL driver for Zoom does not cover it. In the latter case, please use the table `NativePlatformScalarRequests` to upload data to the Zoom API.

Select Zoom API URL: `/webinars/{webinarId}/registrants`

Insert Zoom API URL: `/webinars/{webinarId}/registrants`

Update Zoom API URL: `/webinars/{webinarId}/registrants`

Delete Zoom API URL: `/webinars/{webinarId}/registrants`

Field Selection Method: NotRequired

Select Zoom API Operation: `get /webinars/{webinarId}/registrants`

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `WebinarsByWebinarIdRegistrants`. A value must be provided at all times for required parameters, but optional parameters in general do not need to have a value and the execution will default to a pre-defined behaviour. Values can be specified by position and by name. In both cases, all parameters not specified will be treated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example with ``select * from table(value1, value2, value3)`` on a table with four parameters will use the default value for the fourth parameter and the specified values for the first three.

Value specification by name is done by listing all values that require a value. For example with ``select * from table(name1 => value1, name3 => value3)`` on the same table will use the default values for the second and fourth parameters and the specified values for the first and third.

Name	Data Type	Required	Default Value	Documentation
<code>occurrence_id</code>	string	<input type="checkbox"/>		The meeting occurrence ID
<code>page_number</code>	int64	<input type="checkbox"/>	1	Current page number of returned records
<code>page_size</code>	int64	<input type="checkbox"/>	30	The number of records returned within a single API call
<code>status</code>	string	<input type="checkbox"/>	approved	The registrant status (Values: pending, approved, denied)
<code>webinarId</code>	int64	<input checked="" type="checkbox"/>		The webinar ID

## Table Function Columns

The columns of the table function `WebinarsByWebinarIdRegistrants` are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
<code>page_count</code>	int64		<input type="checkbox"/>	The number of items returned on this page
<code>page_number</code>	int64		<input type="checkbox"/>	The page number of current results
<code>page_size</code>	int64		<input type="checkbox"/>	The number of records returned within a single API call
<code>total_records</code>	int64		<input type="checkbox"/>	The number of all records available across pages



# Index

## - A -

account\_id 46, 51, 207, 209, 213  
 account\_name 15  
 account\_type 15, 86, 88, 90  
 accountId 16, 17, 18, 19, 24, 25, 26, 28, 30, 32, 33,  
 34, 35, 37, 39, 40, 41, 42  
 Accounts 15  
 AccountsByAccountId 16  
 AccountsByAccountId\_DomainsManagedDomains 17  
 AccountsByAccountId\_Plan\_large\_meetingPlans 28  
 AccountsByAccountId\_Plan\_webinarPlans 30  
 AccountsByAccountIdBilling 32  
 AccountsByAccountIdManagedDomains 18  
 AccountsByAccountIdPlans 33  
 AccountsByAccountIdSettings 19  
 action 44, 45, 107, 192  
 action\_time 107  
 add\_at 201  
 added\_at 103, 117  
 add-odata-mandatory-filters 2  
 address 32  
 agenda 120, 135, 236, 246  
 analysis-enforce-row-uniqueness 2  
 answer 157, 171, 172  
 api-access-token 2  
 api-client-id 2  
 api-client-secret 2  
 api-pre-expiry-refresh-sec 2  
 api-redirect-url 2  
 api-refresh-token 2  
 api-scope 2  
 api-token-url 2  
 api-url 2  
 approval\_type 47  
 apt 32  
 assistantId 195  
 attentiveness\_score 158, 173  
 auth\_password 227, 229  
 auth\_user 227, 229

## - B -

BLOB Preferred 141  
 BLOB\_PREFERRED 141

body 25, 26, 27, 34, 35, 37, 39, 40, 41, 42, 48, 49,  
 50, 93, 94, 101, 102, 103, 115, 116, 117, 130, 131,  
 132, 133, 134, 135, 137, 138, 139, 178, 179, 182, 183,  
 184, 186, 199, 200, 201, 225, 226, 227, 235, 236, 238,  
 239, 240, 241, 242, 243  
 BOL\_RESPONSE\_CACHE\_MAX\_AGE\_SEC 141  
 bulk-delete-page-size-rows 2  
 bulk-insert-page-size-bytes 2  
 bulk-insert-page-size-rows 2

## - C -

calender\_name 86, 88, 90  
 call\_in\_number 148  
 calls\_receive 58  
 calls\_send 58  
 camera 86, 88, 90  
 city 32  
 cms\_user\_id 207, 209, 213  
 conference\_code 184, 186, 188, 189, 190, 191  
 conference\_id 143, 144, 145  
 connection\_type 67, 78  
 content 69, 80  
 Content Type 141  
 CONTENT\_TYPE 141  
 country 32, 143, 144  
 country\_label 184, 188, 190  
 country\_name 148  
 created\_at 15, 16, 27, 120, 135, 140, 206, 207,  
 209, 210, 212, 213, 227, 229, 236, 244, 246

## - D -

data\_center 67, 78  
 date 146, 151  
 DATE\_ENDED 141  
 DATE\_STARTED 141  
 date\_time 55, 62, 107  
 deleteAccountsByAccountId 24  
 deleteGroupsByGroupId 95  
 deleteGroupsByGroupIdMembersMemberId 96  
 deleteH323DevicesByDeviceId 92  
 deleteImGroupsByGroupId 110  
 deleteImGroupsByGroupIdMembersMemberId 111  
 deleteMeetingsByMeetingId 118  
 deleteMeetingsByMeetingIdPollsPollId 119  
 deleteMeetingsByMeetingIdRecordings 44  
 deleteMeetingsByMeetingIdRecordingsRecordingId 45  
 deleteUsersByUserId 192  
 deleteUsersByUserIdAssistants 194

deleteUsersByUserIdAssistantsAssistantId	195	download-error-504-gateway-timeout-sleep-initial-ms	2
deleteUsersByUserIdSchedulers	196	download-error-504-gateway-timeout-sleep-max-ms	2
deleteUsersByUserIdSchedulersSchedulerId	197	download-error-504-gateway-timeout-sleep-multiplicato	r 2
deleteUsersByUserIdToken	198	download-error-590-network-connect-timeout-max-tries	2
deleteUsersByUserIdTspTspId	181	download-error-590-network-connect-timeout-sleep-initi	al-ms 2
deleteV2TrackingFieldsByFieldId	177	download-error-590-network-connect-timeout-sleep-ma	x-ms 2
deleteWebhooksByWebhookId	224	download-error-590-network-connect-timeout-sleep-mu	ltiplicator 2
deleteWebinarsByWebinarId	230	download-error-599-network-connect-timeout-max-tries	2
deleteWebinarsByWebinarIdPanelists	231	download-error-599-network-connect-timeout-sleep-initi	al-ms 2
deleteWebinarsByWebinarIdPanelistsPanelistId	232	download-error-599-network-connect-timeout-sleep-ma	x-ms 2
deleteWebinarsByWebinarIdPollsPollId	233	download-error-599-network-connect-timeout-sleep-mu	ltiplicator 2
dept	148, 149, 206, 207, 209, 210, 212, 213	download-error-599-network-connect-timeout-max-tries	2
device	67, 70, 72, 78, 81, 83	download-error-599-network-connect-timeout-sleep-initi	al-ms 2
device_ip	86, 88, 90	download-error-599-network-connect-timeout-sleep-ma	x-ms 2
deviceId	92, 93	download-error-599-network-connect-timeout-sleep-mu	ltiplicator 2
download-error-400-bad-request-max-tries	2	download-error-argument-exception-max-tries	2
download-error-400-bad-request-sleep-initial-ms	2	download-error-argument-exception-sleep-initial-ms	2
download-error-400-bad-request-sleep-max-ms	2	download-error-argument-exception-sleep-max-ms	2
download-error-400-bad-request-sleep-multiplicato	r 2	download-error-argument-exception-sleep-multiplicato	r 2
download-error-408-request-timeout-max-tries	2	download-error-internet-down-max-tries	2
download-error-408-request-timeout-sleep-initial-ms	2	download-error-internet-down-sleep-initial-ms	2
download-error-408-request-timeout-sleep-max-ms	2	download-error-internet-down-sleep-max-ms	2
download-error-408-request-timeout-sleep-multiplicato	r 2	download-error-internet-down-sleep-multiplicator	2
download-error-422-bad-request-max-tries	2	download-error-io-exception-max-tries	2
download-error-422-bad-request-sleep-initial-ms	2	download-error-io-exception-sleep-initial-ms	2
download-error-422-bad-request-sleep-max-ms	2	download-error-io-exception-sleep-max-ms	2
download-error-422-bad-request-sleep-multiplicato	r 2	download-error-io-exception-sleep-multiplicator	2
download-error-429-too-many-requests-max-tries	2	download-error-json-exception-max-tries	2
download-error-429-too-many-requests-sleep-initial-ms	2	download-error-json-exception-sleep-initial-ms	2
download-error-429-too-many-requests-sleep-max-ms	2	download-error-json-exception-sleep-max-ms	2
download-error-429-too-many-requests-sleep-multiplicato	r 2	download-error-json-exception-sleep-multiplicator	2
download-error-502-server-unavailable-max-tries	2	download-error-other-exception-max-tries	2
download-error-502-server-unavailable-sleep-initial-ms	2	download-error-other-exception-sleep-initial-ms	2
download-error-502-server-unavailable-sleep-max-ms	2	download-error-other-exception-sleep-max-ms	2
download-error-502-server-unavailable-sleep-multiplicato	r 2	download-error-other-exception-sleep-multiplicator	2
download-error-503-server-unavailable-max-tries	2	download-error-socket-exception-max-tries	2
download-error-503-server-unavailable-sleep-initial-ms	2	download-error-socket-exception-sleep-initial-ms	2
download-error-503-server-unavailable-sleep-max-ms	2	download-error-socket-exception-sleep-max-ms	2
download-error-503-server-unavailable-sleep-multiplicato	r 2	download-error-socket-exception-sleep-multiplicator	2
download-error-504-gateway-timeout-max-tries	2	download-error-web-exception-max-tries	2
		download-error-web-exception-sleep-initial-ms	2
		download-error-web-exception-sleep-max-ms	2
		download-error-web-exception-sleep-multiplicator	2

download-error-web-not-implemented-max-tries 2 expire\_in 223  
download-error-web-not-implemented-sleep-initial-ms  
2  
download-error-web-not-implemented-sleep-max-ms 2  
2  
download-error-web-not-implemented-sleep-multiplicat  
or 2  
download-error-web-timeout-max-tries 2  
download-error-web-timeout-sleep-initial-ms 2  
download-error-web-timeout-sleep-max-ms 2  
download-error-web-timeout-sleep-multiplicator 2  
download-error-web-unauthorized-max-tries 2  
download-error-web-unauthorized-sleep-initial-ms  
download-error-web-unauthorized-sleep-max-ms 2  
download-error-web-unauthorized-sleep-multiplicator  
2  
Driver 1  
DRY\_RUN 141  
DUMMY 126, 234  
duration 46, 51, 56, 60, 65, 77, 88, 120, 127, 135,  
140, 148, 153, 154, 158, 163, 166, 167, 173, 236, 240,  
246  
Duration (ms) 141  
DURATION\_MS 141

**- F -**

Fail on Error 141  
FAIL\_ON\_ERROR 141  
feature\_cn\_meeting 218  
feature\_in\_meeting 218  
feature\_large\_meeting 218  
feature\_large\_meeting\_capacity 218  
feature\_meeting\_capacity 19, 218  
feature\_webinar 218  
feature\_webinar\_capacity 218  
field 153, 166, 179, 180  
fieldId 177, 178, 180  
files\_receive 58  
files\_send 58  
first\_name 32, 99, 113, 201, 206, 207, 209, 210,  
212, 213  
force-case-sensitive-identifiers 2  
forced-casing-identifiers 2  
free\_usage 151  
from 51, 53, 55, 56, 58, 60, 62, 63, 64, 75, 88, 90,  
104, 105, 107, 108, 148, 149, 151, 160, 161, 163, 165

**- E -**

email 32, 56, 58, 60, 65, 77, 86, 88, 90, 99, 113,  
149, 156, 157, 169, 170, 171, 172, 201, 206, 207, 209,  
210, 212, 213, 215, 217, 221  
email\_1 88  
email\_notification\_alternative\_host\_reminder 19,  
218  
email\_notification\_cancel\_meeting\_reminder 19,  
218  
email\_notification\_cloud\_recording\_avaliable\_reminder  
19  
email\_notification\_jbh\_reminder 19, 218  
email\_notification\_low\_host\_count\_reminder 19  
emoji\_receive 58  
emoji\_send 58  
enable 187  
encryption 93, 94  
End Date 141  
end\_time 56, 60, 65, 69, 77, 80, 88, 127, 148, 153,  
154, 163, 166, 167  
Error Message Code 141  
Error Message Text 141  
ERROR\_MESSAGE\_CODE 141  
ERROR\_MESSAGE\_TEXT 141  
existed 222  
existed\_email 221

**- G -**

group\_receive 58  
group\_send 58  
groupId 95, 96, 98, 99, 101, 103, 110, 111, 112,  
113, 115, 117  
Groups 97  
Groups\_Groups 98  
GroupsByGroupId 98  
GroupsByGroupIdMembers 99

**- H -**

h323\_password 120, 135  
H323Devices 93  
harddisk\_id 67, 70, 72, 78, 81, 83  
has\_3rd\_party\_audio 56, 60, 65, 77, 88  
has\_pstn 56, 60, 65, 77, 88  
has\_recording 56, 60, 65, 77, 88  
has\_screen\_share 56, 60, 65, 77, 88  
has\_sip 56, 60, 65, 77, 88  
has\_video 56, 60, 65, 77, 88  
has\_voip 56, 60, 65, 77, 88  
host 56, 60, 65, 77, 88  
host\_email 148

host\_id 46, 51, 120, 127, 135, 140, 236, 244, 246  
 host\_key 207, 209, 213  
 host\_name 148  
 hosts 28, 30, 35, 37  
 hour 55  
 HTTP Disk Cache Maximum Age (sec) 141  
 HTTP Memory Cache Maximum Age (sec) 141  
 HTTP Method 141  
 HTTP Status Code 141  
 HTTP\_DISK\_CACHE\_MAX\_AGE\_SEC 141  
 HTTP\_DISK\_CACHE\_SAVE 141  
 HTTP\_DISK\_CACHE\_USE 141  
 HTTP\_MEMORY\_CACHE\_MAX\_AGE\_SEC 141  
 HTTP\_MEMORY\_CACHE\_SAVE 141  
 HTTP\_MEMORY\_CACHE\_USE 141  
 HTTP\_METHOD 141  
 HTTP\_STATUS\_CODE 141  
 http-disk-cache-compression-level 2  
 http-disk-cache-directory 2  
 http-disk-cache-ignore-write-errors 2  
 http-disk-cache-max-age-sec 2  
 http-get-timeout-max-ms 2  
 http-get-timeout-ms 2  
 http-memory-cache-compression-level 2  
 http-memory-cache-max-age-sec 2  
 http-post-timeout-max-ms 2  
 http-post-timeout-ms 2  
  
**- | -**  
  
 id\_1 88  
 ids 103, 117, 201  
 ignore-http-400-errors 2  
 ignore-http-401-errors 2  
 ignore-http-402-errors 2  
 ignore-http-403-errors 2  
 ignore-http-404-errors 2  
 ignore-http-422-errors 2  
 ignore-http-429-errors 2  
 ignore-http-500-errors 2  
 ignore-http-502-errors 2  
 ignore-http-503-errors 2  
 ignore-unknown-path-type 2  
 ignore-values-unknown-path 2  
 images\_receive 58  
 images\_send 58  
 ImChat\_SessionsSessions 104  
 ImChatSessions 105  
 ImChatSessions\_MessagesBySessionId 107  
 ImChatSessionsBySessionId 108  
 ImGroups 112  
 ImGroupsByGroupId 112  
 ImGroupsByGroupIdMembers 113  
 in\_meeting\_alert\_guest\_join 19  
 in\_meeting\_allow\_live\_streaming 19, 218  
 in\_meeting\_allow\_show\_zoom\_windows 19  
 in\_meeting\_annotation 19, 218  
 in\_meeting\_anonymous\_question\_answer 19  
 in\_meeting\_attendee\_on\_hold 19, 218  
 in\_meeting\_attention\_tracking 19, 218  
 in\_meeting\_auto\_answer 19  
 in\_meeting\_auto\_saving\_chat 19, 218  
 in\_meeting\_breakout\_room 19, 218  
 in\_meeting\_chat 19, 218  
 in\_meeting\_closed\_caption 19, 218  
 in\_meeting\_co\_host 19, 218  
 in\_meeting\_custom\_live\_streaming 19, 218  
 in\_meeting\_custom\_service\_instructions 19, 218  
 in\_meeting\_dscp\_audio 19  
 in\_meeting\_dscp\_marking 19  
 in\_meeting\_dscp\_video 19  
 in\_meeting\_e2e\_encryption 19, 218  
 in\_meeting\_entry\_exit\_chime 218  
 in\_meeting\_far\_end\_camera\_control 19, 218  
 in\_meeting\_feedback 19, 218  
 in\_meeting\_file\_transfer 19, 218  
 in\_meeting\_group\_hd 19, 218  
 in\_meeting\_non\_verbal\_feedback 218  
 in\_meeting\_original\_audio 19  
 in\_meeting\_p2p\_connection 19  
 in\_meeting\_p2p\_ports 19  
 in\_meeting\_polling 19, 218  
 in\_meeting\_ports\_range 19  
 in\_meeting\_post\_meeting\_feedback 19  
 in\_meeting\_private\_chat 19, 218  
 in\_meeting\_record\_play\_voice 218  
 in\_meeting\_remote\_control 19, 218  
 in\_meeting\_remote\_support 218  
 in\_meeting\_screen\_sharing 19  
 in\_meeting\_sending\_default\_email\_invites 19  
 in\_meeting\_share\_dual\_camera 218  
 in\_meeting\_show\_meeting\_control\_toolbar 19  
 in\_meeting\_stereo\_audio 19  
 in\_meeting\_use\_html\_format\_email 19  
 in\_meeting\_virtual\_background 19, 218  
 in\_meeting\_waiting\_room 218  
 in\_meeting\_watermark 19  
 in\_meeting\_webinar\_question\_answer 19  
 in\_meeting\_whiteboard 19  
 in\_meeting\_workplace\_by\_facebook 19, 218  
 integration\_box 19  
 integration\_dropbox 19

integration\_google\_calendar 19  
 integration\_google\_drive 19  
 integration\_kubi 19  
 integration\_microsoft\_one\_drive 19  
 invalid-json-on-get-max-tries 2  
 invalid-json-on-get-sleep-initial-ms 2  
 invalid-json-on-get-sleep-max-ms 2  
 invalid-json-on-get-sleep-multiplicator 2  
 invalid-json-on-post-max-tries 2  
 invalid-json-on-post-sleep-initial-ms 2  
 invalid-json-on-post-sleep-max-ms 2  
 invalid-json-on-post-sleep-multiplicator 2  
 invantive-sql-compress-sparse-arrays 2  
 invantive-sql-correct-invalid-date 2  
 invantive-sql-forward-filters-to-data-containers 2  
 invantive-sql-share-byte-arrays 2  
 invantive-sql-share-strings 2  
 invantive-sql-shuffle-fetch-results-data-containers 2  
 invantive-use-cache 2  
 invitation 122  
 ip 93, 94  
 ip\_address 67, 70, 72, 78, 81, 83

## - J -

join\_time 67, 70, 72, 78, 81, 83, 158, 173  
 join\_url 120, 134, 135, 140, 236, 238, 240, 244, 246  
 join-set-points-per-request 2

## - L -

language 207, 209, 213  
 last\_client\_version 206, 207, 209, 210, 212, 213  
 last\_login\_time 206, 207, 209, 210, 212, 213  
 last\_message\_sent\_time 104  
 last\_name 32, 99, 113, 201, 206, 207, 209, 210, 212, 213  
 last\_start\_time 86, 88, 90  
 leader\_pin 184, 186, 188, 189, 190, 191  
 leave\_time 67, 70, 72, 78, 81, 83, 158, 173  
 limit-partition-calls-left 2  
 listen\_only\_password 143, 144, 145  
 live\_meeting\_duration 88, 90  
 live\_meeting\_email 88, 90  
 live\_meeting\_end\_time 88, 90  
 live\_meeting\_has\_3rd\_party\_audio 88, 90  
 live\_meeting\_has\_pstn 88, 90  
 live\_meeting\_has\_recording 88, 90  
 live\_meeting\_has\_screen\_share 88, 90  
 live\_meeting\_has\_sip 88, 90  
 live\_meeting\_has\_video 88, 90

live\_meeting\_has\_voip 88, 90  
 live\_meeting\_host 88, 90  
 live\_meeting\_id 88, 90  
 live\_meeting\_participants 88, 90  
 live\_meeting\_start\_time 88, 90  
 live\_meeting\_topic 88, 90  
 live\_meeting\_user\_type 88, 90  
 live\_meeting\_uuid 88, 90  
 location 67, 70, 72, 78, 81, 83  
 login\_type 207, 209, 213, 218  
 log-native-calls-to-disk-max-events 2  
 log-native-calls-to-disk-max-seconds 2  
 log-native-calls-to-disk-on-error 2  
 log-native-calls-to-disk-on-success 2  
 log-native-calls-to-trace 2

## - M -

mac\_addr 67, 70, 72, 78, 81, 83  
 max\_usage 55  
 maximum-discovered-column-count 2  
 maximum-length-identifiers 2  
 max-odata-filters 2  
 max-url-length-accepted 2  
 max-url-length-desired 2  
 mc 51, 53  
 meeting\_id 148  
 meeting\_minutes 146, 149  
 meeting\_type 148  
 meetingId 44, 45, 46, 47, 48, 49, 50, 65, 67, 69, 70, 72, 74, 118, 119, 120, 122, 123, 124, 125, 126, 130, 131, 132, 133, 134, 137, 138, 139, 153, 154, 156, 157, 158, 159  
 meetings 146, 149  
 MeetingsByMeetingId 120  
 MeetingsByMeetingIdInvitation 122  
 MeetingsByMeetingIdPolls 123  
 MeetingsByMeetingIdPollsPollId 124  
 MeetingsByMeetingIdRecordings 46  
 MeetingsByMeetingIdRecordingsSettings 47  
 MeetingsByMeetingIdRegistrants 125  
 meetingUUID 127, 128  
 memberId 96, 111  
 message 107  
 metadata-cache-max-age-sec 2  
 Metrics\_Crc\_ports\_usageCrc\_ports\_hour\_usageCrc 55  
 Metrics\_MeetingsMeetings 56  
 Metrics\_UsersIm 58  
 Metrics\_WebinarsWebinars 60  
 MetricsCrc 62



- MetricsIm 63  
 MetricsMeetings 64  
 MetricsMeetingsByMeetingId 65  
 MetricsMeetingsByMeetingIdParticipants 67  
 MetricsMeetingsByMeetingIdParticipants\_ParticipantsDetailsSharing 69  
 MetricsMeetingsByMeetingIdParticipantsParticipantsQos 70  
 MetricsMeetingsByMeetingIdParticipantsQos 72  
 MetricsMeetingsByMeetingIdParticipantsSharing 74  
 MetricsWebinars 75  
 MetricsWebinarsByWebinarId 77  
 MetricsWebinarsByWebinarIdParticipants 78  
 MetricsWebinarsByWebinarIdParticipants\_ParticipantsDetailsSharing 80  
 MetricsWebinarsByWebinarIdParticipantsParticipantsQos 81  
 MetricsWebinarsByWebinarIdParticipantsQos 83  
 MetricsWebinarsByWebinarIdParticipantsSharing 85  
 MetricsZoomrooms 86  
 MetricsZoomrooms\_Past\_meetingsMeetingsByZoomroomId 88  
 MetricsZoomroomsByZoomroomId 90  
 microphone 67, 78, 86, 88, 90  
 month 146, 152
- N -**
- name 93, 94, 98, 102, 104, 112, 116, 128, 156, 157, 158, 169, 170, 171, 172, 173  
 Native Platform Scalar Requests 141  
 NATIVEPLATFORMSCALARREQUESTS 141  
 network\_type 67, 78  
 new\_users 146  
 next\_page\_token 51, 53, 56, 58, 60, 63, 64, 67, 69, 72, 74, 75, 78, 80, 83, 85, 104, 105, 107, 108, 128, 158, 163, 165, 173  
 npt 141  
 number 143, 144, 184, 187, 188, 190
- O -**
- oauth-unauthorized-max-tries 2  
 oauth-unauthorized-sleep-initial-ms 2  
 oauth-unauthorized-sleep-max-ms 2  
 oauth-unauthorized-sleep-multiplicator 2  
 occurrence\_id 118, 125, 138, 230, 242, 250  
 occurrence\_ids 134, 240  
 on\_demand 47  
 options\_meeting\_connectors 16  
 options\_pay\_mode 16  
 options\_room\_connectors 16  
 options\_share\_mc 16  
 options\_share\_rc 16  
 ORIG\_SYSTEM\_GROUP 141  
 ORIG\_SYSTEM\_REFERENCE 141  
 Original System Group 141  
 Original System Reference 141  
 owner\_email 15, 16, 27  
 owner\_id 16, 27
- P -**
- page\_count 51, 53, 56, 58, 60, 63, 64, 75, 125, 148, 149, 160, 161, 163, 165, 250  
 page\_number 15, 86, 88, 90, 99, 113, 125, 140, 148, 149, 160, 161, 206, 210, 212, 244, 250  
 page\_size 15, 51, 53, 56, 58, 60, 63, 64, 67, 69, 72, 74, 75, 78, 80, 83, 85, 86, 88, 90, 99, 104, 105, 107, 108, 113, 125, 128, 140, 148, 149, 158, 160, 161, 163, 165, 173, 206, 210, 212, 244, 250  
 page-size-rows 2  
 panelistId 232  
 participant\_password 143, 144, 145  
 participantId 70, 81  
 participants 56, 60, 65, 77, 88, 146, 149  
 participants\_count 127, 153, 154, 163, 166, 167  
 partition-slot-based-rate-limit-length-ms 2  
 partition-slot-based-rate-limit-slots 2  
 password 47, 120, 135  
 PastMeetingsByMeetingIdInstances 126  
 PastMeetingsByMeetingUUID 127  
 PastMeetingsByMeetingUUIDParticipants 128  
 PastWebinarsByWebinarIdInstances 234  
 patchAccountsByAccountIdBilling 34  
 patchAccountsByAccountIdOptions 25  
 patchAccountsByAccountIdSettings 26  
 patchGroupsByGroupId 101  
 patchH323DevicesByDeviceId 93  
 patchImGroupsByGroupId 115  
 patchMeetingsByMeetingId 130  
 patchMeetingsByMeetingIdLivestream 131  
 patchMeetingsByMeetingIdLivestreamStatus 132  
 patchMeetingsByMeetingIdRecordingsSettings 48  
 patchTsp 182  
 patchUsersByUserId 199  
 patchUsersByUserIdSettings 200  
 patchUsersByUserIdTspTspId 183  
 patchV2TrackingFieldsByFieldId 178  
 patchWebhooksByWebhookId 225  
 patchWebhooksOptions 226  
 patchWebinarsByWebinarId 235

Payload	141	putAccountsByAccountIdPlansBase	42
PAYLOAD_TEXT	141	putMeetingsByMeetingIdPollsPollId	137
pc_name	67, 70, 72, 78, 81, 83	putMeetingsByMeetingIdRecordingsRecordingIdStatus	49
personal_meeting_url	207, 209, 213	putMeetingsByMeetingIdRecordingsStatus	50
phone_number	32, 148	putMeetingsByMeetingIdRegistrantsStatus	138
pic_file	202	putMeetingsByMeetingIdStatus	139
pic_url	207, 209, 213	putUsersByUserIdEmail	203
plan_audio_callout_countries	28, 30, 33, 35, 37, 39	putUsersByUserIdPassword	204
plan_audio_ddi_numbers	28, 30, 33, 35, 37, 39	putUsersByUserIdStatus	205
plan_audio_premium_countries	28, 30, 33, 35, 37, 39	putWebinarsByWebinarIdPollsPollId	241
plan_audio_tollfree_countries	28, 30, 33, 35, 37, 39	putWebinarsByWebinarIdRegistrantsStatus	242
plan_audio_type	28, 30, 33, 35, 37, 39	putWebinarsByWebinarIdStatus	243
plan_base_hosts	28, 30, 33, 35, 37, 39		
plan_base_type	28, 30, 33, 35, 37, 39		
plan_recording	28, 30, 33, 35, 37, 39		
plan_room_connector_hosts	28, 30, 33, 35, 37, 39		
plan_room_connector_type	28, 30, 33, 35, 37, 39		
plan_usage	151		
plan_zoom_rooms_hosts	28, 30, 33, 35, 37, 39		
plan_zoom_rooms_type	28, 30, 33, 35, 37, 39		
pmi	206, 207, 209, 210, 212, 213		
pollId	119, 124, 137, 233, 241, 249		
postAccounts	27		
postAccountsByAccountId_Plan_large_meetingPlans	35		
postAccountsByAccountId_Plan_webinarPlans	37		
postAccountsByAccountIdPlans	39		
postAccountsByAccountIdPlansAddons	40		
postGroups	102		
postGroupsByGroupIdMembers	103		
postH323Devices	94		
postImGroups	116		
postImGroupsByGroupIdMembers	117		
postMeetingsByMeetingIdPolls	133		
postMeetingsByMeetingIdRegistrants	134		
postUsers	201		
postUsersByUserId_Dial_in_numbersTsp	184		
postUsersByUserIdAssistants	201		
postUsersByUserIdMeetings	135		
postUsersByUserIdPicture	202		
postUsersByUserIdTsp	186		
postUsersByUserIdWebinars	236		
postV2TrackingFields	179		
postWebhooks	227		
postWebinarsByWebinarIdPanelists	238		
postWebinarsByWebinarIdPolls	239		
postWebinarsByWebinarIdRegistrants	240		
pre-request-delay-ms	2		
protocol	93, 94		
putAccountsByAccountIdPlansAddons	41		
		question	157, 171, 172

## - Q -

## - R -

recording	67, 78
recording_account_user_access_recording	19
recording_auto_delete_cmr	19, 218
recording_auto_delete_cmr_days	19, 218
recording_auto_recording	19, 218
recording_cloud_recording	19, 218
recording_cloud_recording_download	19
recording_cloud_recording_download_host	19
recording_count	46, 51
recording_local_recording	19, 218
recording_record_audio_file	19, 218
recording_record_gallery_view	19, 218
recording_record_speaker_view	19, 218
recording_recording_audio_transcript	19, 218
recording_save_chat_text	19, 218
recording_show_timestamp	19, 218
recordingId	45, 49
registrant_id	134, 240
Report_DatesDaily	146
Report_Telephony_usageTelephone	148
Report_UsersUsers	149
ReportCloudRecording	151
ReportDaily	152
ReportMeetings_Tracking_fieldsByMeetingId	153
ReportMeetingsByMeetingId	154
ReportMeetingsByMeetingId_QuestionsPolls	156
ReportMeetingsByMeetingId_QuestionsQuestion_detailsPolls	157
ReportMeetingsByMeetingIdParticipants	158
ReportMeetingsByMeetingIdPolls	159
ReportTelephone	160

ReportUsers 161  
 ReportUsersByUserId\_MeetingsMeetings 163  
 ReportUsersByUserIdMeetings 165  
 ReportWebinars\_Tracking\_fieldsByWebinarId 166  
 ReportWebinarsByWebinarId 167  
 ReportWebinarsByWebinarId\_QuestionsPolls 169  
 ReportWebinarsByWebinarId\_QuestionsQa 170  
 ReportWebinarsByWebinarId\_QuestionsQuestion\_detailsPolls 171  
 ReportWebinarsByWebinarId\_QuestionsQuestion\_detailsQa 172  
 ReportWebinarsByWebinarIdParticipants 173  
 ReportWebinarsByWebinarIdPolls 175  
 ReportWebinarsByWebinarIdQa 176  
 requested-page-size 2  
 requests-parallel-max 2  
 required 179, 180  
 Response Cache Maximum Age (sec) 141  
 RESULT 24, 25, 26, 27, 34, 35, 37, 39, 40, 41, 42, 44, 45, 48, 49, 50, 92, 93, 94, 95, 96, 101, 102, 103, 110, 111, 115, 116, 117, 118, 119, 130, 131, 132, 133, 134, 135, 137, 138, 139, 177, 178, 179, 181, 182, 183, 184, 186, 192, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 224, 225, 226, 227, 230, 231, 232, 233, 235, 236, 238, 239, 240, 241, 242, 243  
 Result BLOB 141  
 Result Text 141  
 RESULT\_BLOB 141  
 RESULT\_DATE\_TIME\_UTC 141  
 RESULT\_NUMBER 141  
 RESULT\_TEXT 141  
 room\_name 86, 88, 90  
 Run without Actions 141  
 schedule\_meting\_participant\_video 19  
 schedulerId 197  
 search\_by\_account 112, 116  
 search\_by\_domain 112, 116  
 search\_by\_ma\_account 112, 116  
 seats 15  
 security\_admin\_change\_name\_pic 19  
 security\_hide\_billing\_info 19  
 security\_import\_photos\_from\_devices 19  
 send\_email\_to\_host 47  
 sender 107  
 session\_id 104, 107, 108  
 sessionId 107, 108  
 settings\_allow\_multiple\_devices 236, 246  
 settings\_alternative\_hosts 120, 135, 236, 246  
 settings\_approval\_type 120, 135, 236, 246  
 settings\_audio 120, 135, 236, 246  
 settings\_auto\_recording 120, 135, 236, 246  
 settings\_close\_registration 120, 135, 236, 246  
 settings\_cn\_meeting 120, 135  
 settings\_enforce\_login 120, 135, 236, 246  
 settings\_enforce\_login\_domains 120, 135, 236, 246  
 settings\_hd\_video 236, 246  
 settings\_host\_video 120, 135, 236, 246  
 settings\_in\_meeting 120, 135  
 settings\_join\_before\_host 120, 135  
 settings\_mute\_upon\_entry 120, 135  
 settings\_on\_demand 236, 246  
 settings\_panelists\_video 236, 246  
 settings\_participant\_video 120, 135  
 settings\_practice\_session 236, 246  
 settings\_registration\_type 120, 135, 236, 246  
 settings\_show\_share\_button 236, 246  
 settings\_use\_pmi 120, 135  
 settings\_waiting\_room 120, 135  
 settings\_watermark 120, 135  
 share\_application 67, 78  
 share\_desktop 67, 78  
 share\_recording 47  
 share\_whiteboard 67, 78  
 show\_social\_share\_buttons 47  
 simulate-http-400-errors 2  
 simulate-http-400-errors-percentage 2  
 simulate-http-401-errors 2  
 simulate-http-401-errors-percentage 2  
 simulate-http-403-errors 2  
 simulate-http-403-errors-percentage 2  
 simulate-http-408-errors 2  
 simulate-http-408-errors-percentage 2  
 simulate-http-429-errors 2  
 simulate-http-429-errors-percentage 2

## - S -

Save HTTP Disk Cache 141  
 Save HTTP Memory Cache 141  
 schedule\_meeting\_audio\_type 218  
 schedule\_meeting\_force\_pmi\_jbh\_password 218  
 schedule\_meeting\_host\_video 218  
 schedule\_meeting\_join\_before\_host 218  
 schedule\_meeting\_participants\_video 218  
 schedule\_meeting\_pstn\_password\_protected 218  
 schedule\_meting\_audio\_type 19  
 schedule\_meting\_enforce\_login 19  
 schedule\_meting\_enforce\_login\_domains 19  
 schedule\_meting\_enforce\_login\_with\_domains 19  
 schedule\_meting\_force\_pmi\_jbh\_password 19  
 schedule\_meting\_host\_video 19  
 schedule\_meting\_join\_before\_host 19  
 schedule\_meting\_not\_store\_meeting\_topic 19

simulate-http-500-errors	2	total_minutes	127, 153, 154, 163, 166, 167
simulate-http-500-errors-percentage	2	total_participants	149, 161
simulate-http-502-errors	2	total_receive	58
simulate-http-502-errors-percentage	2	total_records	17, 18, 51, 53, 56, 58, 60, 63, 64, 75, 97, 98, 123, 125, 148, 149, 160, 161, 163, 165, 180, 229, 247, 248, 250
simulate-http-503-errors	2	total_send	58
simulate-http-503-errors-percentage	2	total_size	46, 51
simulate-http-protocol-errors	2	total_usage	55
simulate-http-protocol-errors-percentage	2	Transaction ID	141
simulate-http-timeout-errors	2	TRANSACTION_ID	141
simulate-http-timeout-errors-percentage	2	transfer_email	192
slot-based-rate-limit-length-ms	2	transfer_meeting	192
slot-based-rate-limit-slots	2	transfer_recording	192
speaker	67, 78, 86, 88, 90	transfer_webinar	192
standardize-identifiers	2	trash	51, 53
standardize-identifiers-casing	2	Tsp	187
Start Date	141	Tsp_Dial_in_numbers	187
start_time	46, 51, 56, 60, 65, 69, 77, 80, 88, 120, 127, 134, 135, 140, 148, 153, 154, 156, 157, 159, 166, 167, 169, 170, 171, 172, 175, 176, 236, 240, 246	Tsp_provider	187
start_url	120, 135, 236, 246	tspld	181, 183, 190, 191
state	32	type	28, 30, 35, 37, 56, 60, 64, 65, 67, 69, 70, 72, 74, 75, 77, 78, 80, 81, 83, 85, 99, 104, 112, 113, 120, 127, 135, 140, 148, 149, 153, 154, 160, 161, 163, 166, 167, 184, 187, 188, 190, 201, 206, 207, 209, 210, 212, 213, 220, 236, 244, 246
status	17, 86, 88, 90, 124, 125, 133, 206, 210, 212, 239, 249, 250	url	141, 227, 229
subscription_end_time	15	usage	151
subscription_start_time	15	Use HTTP Disk Cache	141
Successful	141	Use HTTP Memory Cache	141
SUCCESSFUL	141	use_pmi	207, 209, 213
swagger-specification-file	2	use-batch-insert	2
swagger-specification-http-disk-cache-max-age-sec	2	use-http-disk-cache-read	2
swagger-specification-url	2	use-http-disk-cache-write	2
		use-http-memory-cache-read	2
		use-http-memory-cache-write	2
		user_email	127, 153, 154, 158, 163, 166, 167, 173
		user_id	58, 67, 69, 70, 72, 74, 78, 80, 81, 83, 85, 158, 173
		user_name	58, 67, 69, 70, 72, 74, 78, 80, 81, 83, 85, 127, 149, 153, 154, 163, 166, 167
		user_qos_audio_input_avg_loss	70, 72, 81, 83
		user_qos_audio_input_bitrate	70, 72, 81, 83
		user_qos_audio_input_jitter	70, 72, 81, 83
		user_qos_audio_input_latency	70, 72, 81, 83
		user_qos_audio_input_max_loss	70, 72, 81, 83
		user_qos_audio_output_avg_loss	70, 72, 81, 83
		user_qos_audio_output_bitrate	70, 72, 81, 83
		user_qos_audio_output_jitter	70, 72, 81, 83
		user_qos_audio_output_latency	70, 72, 81, 83

## - T -

telephony_audio_conference_info	19, 218
telephony_show_international_numbers_link	218
telephony_third_party_audio	19, 218
TEXT	216
Timeout (sec)	141
TIMEOUT_SEC	141
timezone	120, 135, 140, 206, 207, 209, 210, 212, 213, 236, 244, 246
title	124, 133, 239, 249
to	51, 53, 55, 56, 58, 60, 62, 63, 64, 75, 88, 90, 104, 105, 107, 108, 148, 149, 151, 160, 161, 163, 165
token	220
topic	46, 51, 56, 60, 65, 77, 88, 120, 127, 134, 135, 140, 153, 154, 163, 166, 167, 236, 240, 244, 246
total	148
total_meeting_minutes	149, 161
total_meetings	149, 161
total_members	98, 102, 112, 116

## - U -

url	141, 227, 229
usage	151
Use HTTP Disk Cache	141
Use HTTP Memory Cache	141
use_pmi	207, 209, 213
use-batch-insert	2
use-http-disk-cache-read	2
use-http-disk-cache-write	2
use-http-memory-cache-read	2
use-http-memory-cache-write	2
user_email	127, 153, 154, 158, 163, 166, 167, 173
user_id	58, 67, 69, 70, 72, 74, 78, 80, 81, 83, 85, 158, 173
user_name	58, 67, 69, 70, 72, 74, 78, 80, 81, 83, 85, 127, 149, 153, 154, 163, 166, 167
user_qos_audio_input_avg_loss	70, 72, 81, 83
user_qos_audio_input_bitrate	70, 72, 81, 83
user_qos_audio_input_jitter	70, 72, 81, 83
user_qos_audio_input_latency	70, 72, 81, 83
user_qos_audio_input_max_loss	70, 72, 81, 83
user_qos_audio_output_avg_loss	70, 72, 81, 83
user_qos_audio_output_bitrate	70, 72, 81, 83
user_qos_audio_output_jitter	70, 72, 81, 83
user_qos_audio_output_latency	70, 72, 81, 83

- user\_qos\_audio\_output\_max\_loss 70, 72, 81, 83  
 user\_qos\_cpu\_usage\_system\_max\_cpu\_usage 70, 72, 81, 83  
 user\_qos\_cpu\_usage\_zoom\_avg\_cpu\_usage 70, 72, 81, 83  
 user\_qos\_cpu\_usage\_zoom\_max\_cpu\_usage 70, 72, 81, 83  
 user\_qos\_cpu\_usage\_zoom\_min\_cpu\_usage 70, 72, 81, 83  
 user\_qos\_date\_time 70, 72, 81, 83  
 user\_type 56, 60, 65, 77, 88  
 userId 51, 53, 135, 140, 143, 144, 145, 163, 165, 181, 183, 184, 186, 188, 189, 190, 191, 192, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 207, 209, 213, 215, 216, 217, 218, 220, 236, 244  
 Users 206  
 Users\_Group\_idsByUserId 207  
 Users\_Im\_group\_idsByUserId 209  
 Users\_UsersGroup\_ids 210  
 Users\_UsersIm\_group\_ids 212  
 UsersByUserId 213  
 UsersByUserId\_MeetingsRecordings 51  
 UsersByUserId\_Tsp\_accountsDedicated\_dial\_in\_numberPac 143  
 UsersByUserId\_Tsp\_accountsDial\_in\_numbersTsp 188  
 UsersByUserId\_Tsp\_accountsGlobal\_dial\_in\_numberPac 144  
 UsersByUserIdAssistants 215  
 UsersByUserIdMeetings 140  
 UsersByUserIdPac 145  
 UsersByUserIdPermissions 216  
 UsersByUserIdRecordings 53  
 UsersByUserIdSchedulers 217  
 UsersByUserIdSettings 218  
 UsersByUserIdToken 220  
 UsersByUserIdTsp 189  
 UsersByUserIdTsp\_Dial\_in\_numbersTspId 190  
 UsersByUserIdTspTspId 191  
 UsersByUserIdWebinars 244  
 UsersEmail 221  
 UsersVanityName 222  
 UsersZpk 223  
 uuid 46, 51, 56, 60, 65, 77, 88, 120, 127, 135, 140, 153, 154, 156, 157, 159, 163, 166, 167, 169, 170, 171, 172, 175, 176, 236, 244, 246
- V -**
- V2TrackingFields 180  
 V2TrackingFieldsByFieldId 180  
 value 153, 166, 207, 209, 210, 212
- vanity\_name 222  
 vanity\_url 16, 207, 209, 213  
 verified 207, 209, 213  
 version 67, 70, 72, 78, 81, 83  
 videos\_receive 58  
 videos\_send 58  
 viewer\_download 47  
 visible 179, 180  
 voice\_receive 58  
 voice\_send 58
- W -**
- webhook\_id 227, 229  
 webhookId 224, 225, 229  
 Webhooks 229  
 WebhooksByWebhookId 229  
 webinarId 77, 78, 80, 81, 83, 85, 166, 167, 169, 170, 171, 172, 173, 175, 176, 230, 231, 232, 233, 234, 235, 238, 239, 240, 241, 242, 243, 246, 247, 248, 249, 250  
 WebinarsByWebinarId 246  
 WebinarsByWebinarIdPanelists 247  
 WebinarsByWebinarIdPolls 248  
 WebinarsByWebinarIdPollsPollId 249  
 WebinarsByWebinarIdRegistrants 250
- Y -**
- year 146, 152
- Z -**
- zip 32  
 Zoom 1, 15, 16, 17, 18, 19, 24, 25, 26, 27, 28, 30, 32, 33, 34, 35, 37, 39, 40, 41, 42, 44, 45, 46, 47, 48, 49, 50, 51, 53, 55, 56, 58, 60, 62, 63, 64, 65, 67, 69, 70, 72, 74, 75, 77, 78, 80, 81, 83, 85, 86, 88, 90, 92, 93, 94, 95, 96, 97, 98, 99, 101, 102, 103, 104, 105, 107, 108, 110, 111, 112, 113, 115, 116, 117, 118, 119, 120, 122, 123, 124, 125, 126, 127, 128, 130, 131, 132, 133, 134, 135, 137, 138, 139, 140, 141, 143, 144, 145, 146, 148, 149, 151, 152, 153, 154, 156, 157, 158, 159, 160, 161, 163, 165, 166, 167, 169, 170, 171, 172, 173, 175, 176, 177, 178, 179, 180, 181, 182, 183, 184, 186, 187, 188, 189, 190, 191, 192, 194, 195, 196, 197, 198, 199, 200, 201, 202, 203, 204, 205, 206, 207, 209, 210, 212, 213, 215, 216, 217, 218, 220, 221, 222, 223, 224, 225, 226, 227, 229, 230, 231, 232, 233, 234, 235, 236, 238, 239, 240, 241, 242, 243, 244, 246, 247, 248, 249, 250

zoom\_rooms\_auto\_start\_stop\_scheduled\_meetings 19  
zoom\_rooms\_cmr\_for\_instant\_meeting 19  
zoom\_rooms\_force\_private\_meeting 19  
zoom\_rooms\_hide\_host\_information 19  
zoom\_rooms\_list\_meetings\_with\_calendar 19  
zoom\_rooms\_start\_airplay\_manually 19  
zoom\_rooms\_ultrasonic 19  
zoom\_rooms\_upcoming\_meeting\_alert 19  
zoom\_rooms\_weekly\_system\_restart 19  
zoom\_rooms\_zr\_post\_meeting\_feedback 19  
zoomroomld 88, 90  
zpk 223



# *invantive* the **SQL** company

Invantive B.V.  
Biesteweg 11  
3849 RD Hierden  
the Netherlands

Tel: +31 88 00 26 500  
Fax: +31 84 22 58 178  
info@invantive.com  
invantive.com

IBAN NL25 BUNQ 2098 2586 07  
Chamber of Industry and Commerce  
13031406  
VAT NL812602377B01  
RSIN 8122602377  
Managing Director: Guido Leenders  
Registered office: Roermond