



Tixly 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 weaponry 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

1	SQL Driver for Tixly API	1
2	SQL Driver Attributes for Tixly API	2
3	Schema: Customer	15
3.1	Tables	15
3.1.1	AnonymizedCustomers: Tixly Anonymized Customers	15
3.1.2	CustomerById: Tixly Customer by ID	16
3.1.3	CustomerDonationSalesByCustomerId: Tixly Customer Donation Sales by Customer ID	18
3.1.4	CustomerEventSalesAllocationsByCustomerId: Tixly Customer Event Sales Allocations by Customer ID	19
3.1.5	CustomerEventSalesByCustomerId: Tixly Customer Event Sales by Customer ID	21
3.1.6	CustomerEventSalesCategoriesByCustomerId: Tixly Customer Event Sales Categories by Customer ID	22
3.1.7	CustomerEventSalesTagsByCustomerId: Tixly Customer Event Sales Tags by Customer ID	24
3.1.8	CustomerGiftCardsById: Tixly Customer Gift Cards by ID	25
3.1.9	CustomerMembershipSalesByCustomerId: Tixly Customer Membership Sales by Customer ID	27
3.1.10	CustomerMetadataById: Tixly Customer Metadata by ID	28
3.1.11	CustomerPlusMetadata: Tixly Customer plus Metadata	30
3.1.12	CustomerPlusTags: Tixly Customer plus Tags	31
3.1.13	CustomerProductSalesByCustomerId: Tixly Customer Product Sales by Customer ID	33
3.1.14	Customers: Tixly Customers	34
3.1.15	CustomerSubscriptionSalesByCustomerId: Tixly Customer Subscription Sales by Customer ID	36
3.1.16	CustomerTags: Tixly Customer Tags	37
3.1.17	CustomerTags2ByCustomerId: Tixly Customer Tags by Customer ID	38
3.1.18	CustomerTagsByCustomerId: Tixly Customer Tags by Customer ID	39
3.1.19	CustomerTicketsByCustomerId: Tixly Customer Tickets by Customer ID	41
3.1.20	DeletedCustomerPlusMetadata: Tixly Deleted Customers plus Metadata	42
3.1.21	DeletedCustomers: Tixly Deleted Customers	44
3.1.22	DeletedCustomerTags: Tixly Deleted Customer Tags	45
3.1.23	MergedCustomerOperationContents	47
3.1.24	MergedCustomerOperations	48
3.1.25	SaleMetadataRange: Tixly Sale Metadata Range	48
3.1.26	SaleRange: Tixly Sale Range	50
3.1.27	SaleTagsRange: Tixly Sale Tags Range	52
4	Schema: Donation	54
4.1	Tables	54
4.1.1	DeletedMembershipDonations: Tixly Deleted Membership Donations	54
4.1.2	Donations: Tixly Donations	55
4.1.3	DonationTypes: Tixly Donation Types	56
5	Schema: Event	57
5.1	Tables	57
5.1.1	DeletedEvents: Tixly Deleted Events	57
5.1.2	EventAllocations: Tixly Event Allocations	58
5.1.3	EventAllocationsByEventId: Tixly Event Allocations by Event ID	60
5.1.4	EventCategories: Tixly Event Categories	61
5.1.5	EventCategories1: Tixly Event Categories	63
5.1.6	EventCategoriesData: Tixly Event Categories Data	63
5.1.7	EventCustomerPlusMetadataById: Tixly Event Customer plus Metadata by ID	64
5.1.8	EventCustomersByEventId: Tixly Event Customers by Event ID	66
5.1.9	EventCustomerTagsById: Tixly Customer Tags by ID	67

5.1.10	EventDeletedSeasons: Tixly Event Deleted Seasons	69
5.1.11	Events: Tixly Events	70
5.1.12	EventTags: Tixly Event Tags	72
5.1.13	EventTags1: Tixly Event Tags	74
5.1.14	EventTagsData: Tixly Event Tags Data	74
5.1.15	EventTickets: Tixly Event Tickets	75
5.1.16	EventTicketsByEventId: Tixly Event Tickets by Event ID	76
5.1.17	FutureEventAllocations: Tixly Future Event Allocations	78
5.1.18	FutureEventCategories: Tixly Future Event Categories	80
5.1.19	FutureEvents: Tixly Future Events	81
5.1.20	FutureEventTags: Tixly Future Event Tags	83
6	Schema: EventGroup	85
6.1	Tables	85
6.1.1	DeletedEventGroups: Tixly Deleted Event Groups	85
6.1.2	EventGroupById: Tixly Event Group by ID	86
6.1.3	EventGroupEventAllocationsByEventGroupId: Tixly Event Group Allocations by Event Group ID	87
6.1.4	EventGroupEventCategoriesByEventGroupId: Tixly Event Group Categories by Event Group ID	88
6.1.5	EventGroupEventsByEventGroupId: Tixly Event Group Events by Event Group ID	90
6.1.6	EventGroupEventTagsByEventGroupId: Tixly Event Group Tags by Event Group ID	92
6.1.7	EventGroups: Tixly Event Groups	93
7	Schema: Giftcard	94
7.1	Tables	94
7.1.1	GiftCardRange: Tixly Gift Card Range	94
8	Schema: Membership	95
8.1	Tables	95
8.1.1	DeletedMemberships: Tixly Deleted Memberships	95
8.1.2	MembershipDonationTypes: Tixly Membership Donation Types	97
8.1.3	Memberships: Tixly Memberships	97
9	Schema: Native	99
9.1	Tables	99
9.1.1	NATIVEPLATFORMSCALARREQUESTS: Tixly Native Platform Scalar Requests	99
10	Schema: Permission	100
10.1	Tables	100
10.1.1	DeletedPermissions: Tixly Deleted Permissions	100
10.1.2	PermissionRegistrationsByPermissionId: Tixly Permission Registrations by Permission ID	101
10.1.3	PermissionRemovalsByPermissionId: Tixly Permission Removals by Permission ID	102
10.1.4	Permissions: Tixly Permissions	103
11	Schema: Product	104
11.1	Tables	104
11.1.1	Products	104
11.1.2	SoldProducts: Tixly Sold Products	105
12	Schema: Subscription	106
12.1	Tables	106
12.1.1	Subscriptions: Tixly Subscriptions	106
12.1.2	SubscriptionTypeEvents: Tixly Subscription Type Events	107
12.1.3	SubscriptionTypes: Tixly Subscription Types	108
13	Schema: Tag	109
13.1	Tables	109

13.1.1	CustomerDataByTagId: Tixly Customer Data by Tag ID	109
13.1.2	CustomerMetadataByTagId: Tixly Customer Metadata by Tag ID	111
13.1.3	CustomersByTagId: Tixly Customers by Tag ID	113
13.1.4	DeletedTags: Tixly Deleted Tags	115
13.1.5	Tags: Tixly Tags	116
14	Schema: Ticket	116
14.1	Tables	116
14.1.1	DeletedTicketTypes: Tixly Deleted Ticket Types	116
14.1.2	TicketTypeGroups: Tixly Ticket Type Groups	117
14.1.3	TicketTypes: Tixly Ticket Types	118
	Index	119

1 SQL Driver for Tixly API

Invantive SQL is the fastest, easiest and most reliable way to exchange data with the Tixly 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](#). Invantive Support or other users will try to help you.

Tixly is online software for leisure and entertainment management. Tixly is available globally.

The Tixly driver covers 79 tables and 1847 columns.

Tixly 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 [Tixly Power BI connector](#) is based on the Invantive SQL driver for Tixly, 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 Tixly API into traditional databases such as SQL Server (on-premises and Azure), MySQL, PostgreSQL and Oracle is possible using [Invantive Data Replicator](#). Invantive Data Replicator automatically creates and maintains Tixly datawarehouses, possibly in combination with data from over 75 other (cloud) platforms. Invantive Data Replicator supports data volumes up to over 1 TB and over 5.000 companies. The on-premise edition of Invantive Bridge offers an Tixly ADO.net provider.

Finally, online web apps can be build for Tixly 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 Tixly 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 Tixly API request and response.

Specifications

The SQL driver for Tixly does not support partitioning. Define one data container in a database for each company in Tixly 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 from the database definition, on log on and during use. A full list of configuration options is listed in the [driver attributes](#) ².

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 Tixly the comparison of two texts is case sensitive by default.

Changes and bug fixes on the Tixly SQL driver can be found in the [release notes](#). Get access to the community through the [Tixly section](#) of the Invantive forums.

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

Alias: `tixly`

Recommended alias: `tly`

Driver code for use in settings.xml

Updated 22-12-2022 21:15 using Invantive SQL version 22.1.101-BETA+3681.

2 SQL Driver Attributes for Tixly API

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

The Tixly 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.
- Log on: value to be specified interactively by user during log on in a user interface.

The connection string for Tixly can be found in the settings*.xml file used for the database. The reference manuals contain instructions how to relocate the settings*.xml files. Settings*.xml files are typically located in the %USERPROFILE%\invantive folder in most deployment scenarios. 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 Tixly driver can be configured using the following attributes:

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Drivers 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	Enforce rows to be unique for software analysis.	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 authorization scope(s) to request an OAuth token for.	OData		✓		✓	
api-token-url	The token URI is the OAuth2 endpoint to exchange tokens with.	OData		✓		✓	
api-url	URL of web service.	OData		✓		✓	
bulk-delete-page-size-rows	Number of rows to delete per batch when bulk deleting.	Shared	10000	✓	✓	✓	
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	✓	✓	✓	

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-400-bad-request-max-tries	Maximum number of tries w hen HTTP server reports bad format during retrieval of data.		3	✓	✓	✓	
dow nload-error-400-bad-request-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen HTTP server reports that the API server is unavailable during retrieval of data.		500	✓	✓	✓	
dow nload-error-400-bad-request-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen HTTP server reports that the API server is unavailable during retrieval of data.		5000	✓	✓	✓	
dow nload-error-400-bad-request-sleep-multiplicator	Multiplication factor for sleep betw een retries HTTP server reports that the API server is unavailable during retrieval of data.		2	✓	✓	✓	
dow nload-error-408-request-timeout-max-tries	Maximum number of tries w hen the w ebsite reports a HTTP status 408.		10	✓	✓	✓	
dow nload-error-408-request-timeout-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen the w ebsite reports a HTTP status 408.		10000	✓	✓	✓	
dow nload-error-408-request-timeout-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen the w ebsite reports a HTTP status 408.		300000	✓	✓	✓	
dow nload-error-408-request-timeout-sleep-multiplicator	Multiplication factor for sleep betw een retries w hen the w ebsite reports a HTTP status 408.		2	✓	✓	✓	
dow nload-error-422-bad-request-max-tries	Maximum number of tries w hen HTTP server reports unprocessable entity during retrieval of data.		30	✓	✓	✓	
dow nload-error-422-bad-request-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen HTTP server reports unprocessable entity during retrieval of data.		10000	✓	✓	✓	
dow nload-error-422-bad-request-sleep-max-ms	Maximum sleep in milliseconds betw een retries w hen HTTP server reports unprocessable entity during retrieval of data.		300000	✓	✓	✓	
dow nload-error-422-bad-request-sleep-multiplicator	Multiplication factor for sleep betw een retries HTTP server reports unprocessable entity during retrieval of data.		2	✓	✓	✓	
dow nload-error-429-too-many-requests-max-tries	Maximum number of tries w hen the w ebsite reports that too many requests have been made during a timeslot of one minute or one day.		10	✓	✓	✓	
dow nload-error-429-too-many-requests-sleep-initial-ms	Initial sleep in milliseconds betw een retries w hen the w ebsite reports that too many requests have been made during a timeslot of one minute or one day.		10000	✓	✓	✓	

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-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-multiplicator	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 HTTP 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 HTTP 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 HTTP server reports that a bad gateway during retrieval of data.		300000	✓	✓	✓	
dow nload-error-502-server-unavailable-sleep-multiplicator	Multiplication factor for sleep between retries HTTP server reports a bad gateway during retrieval of data.		2	✓	✓	✓	
dow nload-error-503-server-unavailable-max-tries	Maximum number of tries when HTTP 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 HTTP 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 HTTP server reports that the API server is unavailable during retrieval of data.		300000	✓	✓	✓	
dow nload-error-503-server-unavailable-sleep-multiplicator	Multiplication factor for sleep between retries HTTP 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-timeout-sleep-initial-ms	Initial sleep in milliseconds between retries when the website reports a gateway timeout.		10000	✓	✓	✓	
dow nload-error-504-gateway-	Maximum sleep in milliseconds between retries when the website reports a gateway timeout.		300000	✓	✓	✓	

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-max-ms							
dowload-error-504-gateway-timeout-sleep-multiplicator	Multiplication factor for sleep between retries when the website reports a gateway timeout.		2	✓	✓	✓	
dowload-error-590-network-connect-timeout-max-tries	Maximum number of tries when the website reports a HTTP status 590.		10	✓	✓	✓	
dowload-error-590-network-connect-timeout-sleep-initial-ms	Initial sleep in milliseconds between retries when the website reports a HTTP status 590.		10000	✓	✓	✓	
dowload-error-590-network-connect-timeout-sleep-max-ms	Maximum sleep in milliseconds between retries when the website reports a HTTP status 590.		300000	✓	✓	✓	
dowload-error-590-network-connect-timeout-sleep-multiplicator	Multiplication factor for sleep between retries when the website reports a HTTP status 590.		2	✓	✓	✓	
dowload-error-599-network-connect-timeout-max-tries	Maximum number of tries when the website reports a HTTP status 599.		10	✓	✓	✓	
dowload-error-599-network-connect-timeout-sleep-initial-ms	Initial sleep in milliseconds between retries when the website reports a HTTP status 599.		10000	✓	✓	✓	
dowload-error-599-network-connect-timeout-sleep-max-ms	Maximum sleep in milliseconds between retries when the website reports a HTTP status 599.		300000	✓	✓	✓	
dowload-error-599-network-connect-timeout-sleep-multiplicator	Multiplication factor for sleep between retries when the website reports a HTTP status 599.		2	✓	✓	✓	
dowload-error-argument-exception-max-tries	Maximum number of tries when an argument exception is returned when downloading a blob.		10	✓	✓	✓	
dowload-error-argument-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when an argument exception is returned when downloading a blob.		10000	✓	✓	✓	
dowload-error-argument-exception-sleep-max-ms	Maximum sleep in milliseconds between retries when an argument exception is returned when downloading a blob.		300000	✓	✓	✓	
dowload-error-argument-exception-	Multiplication factor for sleep between retries when an argument		2	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
sleep-multiplicator	exception is returned when downloading a blob.						
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	✓	✓	✓	
download-error-other-exception-sleep-initial-ms	Initial sleep in milliseconds between retries when an unqualified error occurs during retrieval of data.		10000	✓	✓	✓	

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-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-	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
implemented-sleep-multiplicator							
download-error-web-timeout-max-tries	Maximum number of tries when the connection reports a timeout.		10	✓	✓	✓	
download-error-web-timeout-sleep-initial-ms	Initial sleep in milliseconds between retries when the connection reports a timeout.		1000	✓	✓	✓	
download-error-web-timeout-sleep-max-ms	Maximum sleep in milliseconds between retries when the connection reports a timeout.		30000	✓	✓	✓	
download-error-web-timeout-sleep-multiplicator	Multiplication factor for sleep between retries when the connection reports a timeout.		2	✓	✓	✓	
download-error-web-unauthorized-max-tries	Maximum number of tries when the connection reports an unauthorized error.		1	✓	✓	✓	
download-error-web-unauthorized-sleep-initial-ms	Initial sleep in milliseconds between retries when the connection reports an unauthorized error.		10000	✓	✓	✓	
download-error-web-unauthorized-sleep-max-ms	Maximum sleep in milliseconds between retries when the connection reports an unauthorized error.		300000	✓	✓	✓	
download-error-web-unauthorized-sleep-multiplicator	Multiplication factor for sleep between retries when the connection reports an unauthorized error.		2	✓	✓	✓	
fail-on-duplicate-rows	When true, an error is raised when there are multiple rows with identical values in a single table..		False	✓	✓	✓	✓
force-case-sensitive-identifiers	Consider identifiers as case-sensitive independent of the platform capabilities.	Shared	False	✓	✓	✓	
forced-casing-identifiers	Forced casing of identifiers. Choose from: Unset, Lower, 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-directory	Directory where HTTP cache is stored.	Shared	C:\Users\gle3.WS212\Invasive\Cache\http\gle3\shared	✓	✓	✓	
http-disk-cache-ignore-write-errors	Whether to ignore write 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	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
http-get-timeout-max-ms	HTTP GET maximum timeout on retry (ms).		24000	✓	✓	✓	
http-get-timeout-ms	HTTP GET timeout (ms).		56000	✓	✓	✓	
http-memory-cache-compression-level	Compression level for the HTTP memory cache, ranging from 1 (little) to 9 (intense). Default is 5.	OData	5	✓	✓	✓	
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).		58000	✓	✓	✓	
http-post-timeout-ms	HTTP POST timeout (ms).		57000	✓	✓	✓	
ignore-http-400-errors	Ignore HTTP 400 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-401-errors	Ignore HTTP 401 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-402-errors	Ignore HTTP 402 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-403-errors	Ignore HTTP 403 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-404-errors	Ignore HTTP 404 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-422-errors	Ignore HTTP 422 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-429-errors	Ignore HTTP 429 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-500-errors	Ignore HTTP 500 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-502-errors	Ignore HTTP 502 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
ignore-http-503-errors	Ignore HTTP 503 errors when exchanging results with the HTTP 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		True	✓	✓	✓	✓

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Drivers File	Set from Log On
	generated when a value occurs outside a path otherwise.						
invalid-json-on-get-max-tries	Maximum number of tries when the JSON received on GET is invalid.		1	✓	✓	✓	
invalid-json-on-get-sleep-initial-ms	Initial sleep in milliseconds between retries when the JSON received on GET is invalid.		1000	✓	✓	✓	
invalid-json-on-get-sleep-max-ms	Maximum sleep in milliseconds between retries when the JSON received on GET is invalid.		10000	✓	✓	✓	
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.		1000	✓	✓	✓	
invalid-json-on-post-sleep-max-ms	Maximum sleep in milliseconds between retries when the JSON received on POST is invalid.		10000	✓	✓	✓	
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	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
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 call events to register from last activation.	Shared		✓	✓	✓	
log-native-calls-to-disk-max-seconds	Maximum number of seconds to register calls from last activation.	Shared		✓	✓	✓	
log-native-calls-to-disk-on-error	Registers native calls to data container backend as disk files when the call raised an error.	Shared	False	✓	✓	✓	
log-native-calls-to-disk-on-success	Registers native calls to data container backend as disk files when the call raised no error.	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	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	✓	✓	✓	
partition-slot-based-rate-limit-length-ms	Total length in milliseconds 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		✓		✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Driver's File	Set from Log On
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		✓	✓	✓	
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 HTTP endpoint.		False	✓	✓	✓	
simulate-http-400-errors-percentage	Percentage of simulated HTTP 400 errors when exchanging results with the HTTP endpoint.		0	✓	✓	✓	
simulate-http-401-errors	Simulate HTTP 401 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
simulate-http-401-errors-percentage	Percentage of simulated HTTP 401 errors when exchanging results with the HTTP endpoint.		0	✓	✓	✓	
simulate-http-403-errors	Simulate HTTP 403 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
simulate-http-403-errors-percentage	Percentage of simulated HTTP 403 errors when exchanging results with the HTTP endpoint.		0	✓	✓	✓	
simulate-http-408-errors	Simulate HTTP 408 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
simulate-http-408-errors-percentage	Percentage of simulated HTTP 408 errors when exchanging results with the HTTP endpoint.		0	✓	✓	✓	
simulate-http-429-errors	Simulate HTTP 429 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
simulate-http-429-errors-percentage	Percentage of simulated HTTP 429 errors when exchanging results with the HTTP endpoint.		0	✓	✓	✓	
simulate-http-500-errors	Simulate HTTP 500 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
simulate-http-500-errors-percentage	Percentage of simulated HTTP 500 errors when exchanging results with the HTTP endpoint.		0	✓	✓	✓	
simulate-http-502-errors	Simulate HTTP 502 errors when exchanging results with the HTTP endpoint.		False	✓	✓	✓	
simulate-http-502-errors-percentage	Percentage of simulated HTTP 502 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 HTTP endpoint.						
simulate-http-503-errors	Simulate HTTP 503 errors w hen exchanging results w ith the HTTP endpoint.		False	✓	✓	✓	
simulate-http-503-errors-percentage	Percentage of simulated HTTP 503 errors w hen exchanging results w ith the HTTP endpoint.		0	✓	✓	✓	
simulate-http-protocol-errors	Simulate HTTP protocol errors w hen exchanging results w ith the HTTP endpoint.		False	✓	✓	✓	
simulate-http-protocol-errors-percentage	Percentage of simulated HTTP protocol errors w hen exchanging results w ith the HTTP endpoint.		0	✓	✓	✓	
simulate-http-timeout-errors	Simulate HTTP timeout errors w hen exchanging results w ith the HTTP endpoint..		False	✓	✓	✓	
simulate-http-timeout-errors-percentage	Percentage of simulated HTTP timeout errors w hen exchanging results w ith the HTTP endpoint.		0	✓	✓	✓	
slot-based-rate-limit-length-ms	Total length in milliseconds 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	False	✓	✓	✓	
use-http-disk-cache-write	Whether to memorize HTTP responses on disk.	Shared	False	✓	✓	✓	

Code	Description	Origin	Default Value	Set from Connection String	Set from Set SQL-Statement	Set from Drivers File	Set from Log On
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: Customer

3.1 Tables

3.1.1 AnonymizedCustomers: Tixly Anonymized Customers

Gets merged customers details.

Catalog: Tixly

Schema: Customer

Label: Anonymized Customers

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function AnonymizedCustomers. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		
DateTo	datetime	<input type="checkbox"/>		

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Anonymized	datetime	Anonymized	<input type="checkbox"/>	
AnonymizedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	

3.1.2 CustomerByID: Tixly Customer by ID

Fetches a specific customer by his customer ID

Catalog: Tixly

Schema: Customer

Label: Customer by ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerByID. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.3 CustomerDonationSalesByCustomerId: Tixly Customer Donation Sales by Customer ID

Fetches the donations from a specific customer

Catalog: Tixly

Schema: Customer

Label: Customer Donation Sales by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerDonationSalesByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Renew DayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
Renew MonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Vat	double	VAT	<input type="checkbox"/>	

3.1.4 CustomerEventSalesAllocationsByCustomerId: Tixly Customer Event Sales Allocations by Customer ID

Fetches the events that a specific customer has bought a ticket to

Catalog: Tixly

Schema: Customer

Label: Customer Event Sales Allocations by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerEventSalesAllocationsByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Block	boolean	Block	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
ExpiresHoursBefore	int32	Expires Hours Before	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
ReservedTickets	int32	Reserved Tickets	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Seats	int32	Seats	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
SoldTickets	int32	Sold Tickets	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

3.1.5 CustomerEventSalesByCustomerId: Tixly Customer Event Sales by Customer ID

Fetches the events that a specific customer has bought a ticket to

Catalog: Tixly

Schema: Customer

Label: Customer Event Sales by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerEventSalesByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
EventGroupld	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
Hallld	int32	Hall ID	<input type="checkbox"/>	
ld	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllow ed	boolean	Online Reservation Allow ed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
Promoterld	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusld	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
Seasonld	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
Venueld	int32	Venue ID	<input type="checkbox"/>	

3.1.6 CustomerEventSalesCategoriesByCustomerId: Tixly Customer Event Sales Categories by Customer ID

Fetches the events that a specific customer has bought a ticket to

Catalog: Tixly

Schema: Customer

Label: Customer Event Sales Categories by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerEventSalesCategoriesByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a `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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
Venueld	int32	Venue ID	<input type="checkbox"/>	

3.1.7 CustomerEventSalesTagsByCustomerId: Tixly Customer Event Sales Tags by Customer ID

Fetches the events that a specific customer has bought a ticket to

Catalog: Tixly

Schema: Customer

Label: Customer Event Sales Tags by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerEventSalesTagsByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

3.1.8 CustomerGiftCardsById: Tixly Customer Gift Cards by ID

Fetches the giftcars that a specific customer has bought

Catalog: Tixly

Schema: Customer

Label: Customer Gift Cards by ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerGiftCardsByld. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Amount	double	Amount	<input type="checkbox"/>	
AmountLeft	double	Amount Left	<input type="checkbox"/>	
AmountUsed	double	Amount Used	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Expires	datetime	Expires	<input type="checkbox"/>	
ExpiresUTCUnix	int64		<input type="checkbox"/>	
GiftCardCount	int32	Giftcard Count	<input type="checkbox"/>	
Number	string	Number	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Pin	string	PIN	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	

3.1.9 CustomerMembershipSalesByCustomerId: Tixly Customer Membership Sales by Customer ID

Fetches the membership a specific customer has bought

Catalog: Tixly

Schema: Customer

Label: Customer Membership Sales by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerMembershipSalesByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Renew DayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Renew MonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

3.1.10 CustomerMetadataById: Tixly Customer Metadata by ID

Fetches a specific customer by his customer ID

Catalog: Tixly

Schema: Customer

Label: Customer Metadata by ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerMetadataById. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
CustomerAttributeID	int32	Customer Attribute ID	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Value	string	Value	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.11 CustomerPlusMetadata: Tixly Customer plus Metadata

Fetches all customers you have access to. Pass in the DateFrom and DateTo query parameters to get customers that have been edited or created in a specific date range.

Catalog: Tixly

Schema: Customer

Label: Customer plus Metadata

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerPlusMetadata. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Created/Edited customer from
DateTo	datetime	<input type="checkbox"/>		Created/Edited customer to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
CRN	string	CRN	<input type="checkbox"/>	
CustomerAttributeID	int32	Customer Attribute ID	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Value	string	Value	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.12 CustomerPlusTags: Tixly Customer plus Tags

Fetches all customers you have access to. Pass in the DateFrom and DateTo query parameters to get customers that have been edited or created in a specific date range.

Catalog: Tixly

Schema: Customer

Label: Customer plus Tags

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `CustomerPlusTags`. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Created/Edited customer from
DateTo	datetime	<input type="checkbox"/>		Created/Edited customer to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.13 CustomerProductSalesByCustomerId: Tixly Customer Product Sales by Customer ID

Fetches the products a specific customer has bought

Catalog: Tixly

Schema: Customer

Label: Customer Product Sales by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerProductSalesByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
ProductCount	int32	Product Count	<input type="checkbox"/>	
ProductId	int32	Product ID	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	

3.1.14 Customers: Tixly Customers

Fetches all customers you have access to. Pass in the DateFrom and DateTo query parameters to get customers that have been edited or created in a specific date range.

Catalog: Tixly

Schema: Customer

Label: Customers

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function Customers. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Created/Edited customer from
DateTo	datetime	<input type="checkbox"/>		Created/Edited customer to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.15 CustomerSubscriptionSalesByCustomerId: Tixly Customer Subscription Sales by Customer ID

Fetches the subscriptions from a specific customer

Catalog: Tixly

Schema: Customer

Label: Customer Subscription Sales by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `CustomerSubscriptionSalesByCustomerId`. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
CustomerName	string	Customer Name	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
SubscriptionTypeId	int32	Subscription Type ID	<input type="checkbox"/>	

3.1.16 CustomerTags: Tixly Customer Tags

Fetches all customer tag to customer connections that you have access to. This can be used with the Get all customer tags endpoint and all fetched customer data to quickly connect the two together.

Catalog: Tixly

Schema: Customer

Label: Customer Tags

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
TagId	int32	Tag ID	<input type="checkbox"/>	

3.1.17 CustomerTags2ByCustomerId: Tixly Customer Tags by Customer ID

Fetches all customer tags on a specific customer. This endpoint is not meant to be called once for every customer in the feed, for that purpose it is much more efficient to use the Get all customer tags customer connection endpoint and link them together that way.

Catalog: Tixly

Schema: Customer

Label: Customer Tags by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerTags2ByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

The columns of the table function CustomerTags2ByCustomerId are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

3.1.18 CustomerTagsByCustomerId: Tixly Customer Tags by Customer ID

Fetches a specific customer by his customer ID

Catalog: Tixly

Schema: Customer

Label: Customer Tags by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerTagsByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.19 CustomerTicketsByCustomerId: Tixly Customer Tickets by Customer ID

Fetches the tickets a specific customer has bought

Catalog: Tixly

Schema: Customer

Label: Customer Tickets by Customer ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerTicketsByCustomerId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the customer

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Entrance	string	Entrance	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Fee	double	Fee	<input type="checkbox"/>	
IsAnonymousSale	boolean	Is Anonymous Sale	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
PriceZone	string	Price Zone	<input type="checkbox"/>	
PriceZoneld	int32	Price Zone ID	<input type="checkbox"/>	
Row	string	Row	<input type="checkbox"/>	
Scanned	boolean	Scanned	<input type="checkbox"/>	
Seat	string	Seat	<input type="checkbox"/>	
Section	string	Section	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	
SubscriptionTypeld	int32	Subscription Type ID	<input type="checkbox"/>	
TicketCount	int32	Ticket Count	<input type="checkbox"/>	
TicketId	int32	Ticket ID	<input type="checkbox"/>	
TicketType	string	Ticket Type	<input type="checkbox"/>	
TicketTypeld	int32	Ticket Type ID	<input type="checkbox"/>	

3.1.20 DeletedCustomerPlusMetadata: Tixly Deleted Customers plus Metadata

Fetches all deleted customers you have access to.

Catalog: Tixly

Schema: Customer

Label: Deleted Customers plus Metadata

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedCustomerPlusMetadata. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a `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
DateFrom	datetime	<input type="checkbox"/>		Deleted customer from
DateTo	datetime	<input type="checkbox"/>		Deleted customer to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
CustomerAttributeID	int32	Customer Attribute ID	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Value	string	Value	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.21 DeletedCustomers: Tixly Deleted Customers

Fetches all deleted customers you have access to.

Catalog: Tixly

Schema: Customer

Label: Deleted Customers

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedCustomers. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Deleted customer from
DateTo	datetime	<input type="checkbox"/>		Deleted customer to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.22 DeletedCustomerTags: Tixly Deleted Customer Tags

Fetches all deleted customers you have access to.

Catalog: Tixly

Schema: Customer

Label: Deleted Customer Tags

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedCustomerTags. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Deleted customer from
DateTo	datetime	<input type="checkbox"/>		Deleted customer to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Description	string	Description	<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.23 MergedCustomerOperationContents

Gets merged customers details.

Catalog: Tixly

Schema: Customer

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
MasterCustomerId	int32	Master Customer ID	<input type="checkbox"/>	
Merged	datetime	Merged	<input type="checkbox"/>	
MergedUTCUnix	int64		<input type="checkbox"/>	
TEXT	int32	Text	<input type="checkbox"/>	

3.1.24 MergedCustomerOperations

Gets merged customers details.

Catalog: Tixly

Schema: Customer

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
MasterCustomerId	int32	Master Customer ID	<input type="checkbox"/>	
Merged	datetime	Merged	<input type="checkbox"/>	
MergedUTCUnix	int64		<input type="checkbox"/>	

3.1.25 SaleMetadataRange: Tixly Sale Metadata Range

Fetches customers by sale range, which means if the customer buys or is refunded an order in the date range it will be included. This endpoint is good to use if you are fetching from other endpoints by a specific date range, that can be matched in this endpoint to link the data. The dates are to and from and include time as well so if you would like an entire day then you need to indicate the end time of that day or the start of the next as the DateTo parameter.

Catalog: Tixly

Schema: Customer

Label: Sale Metadata Range

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function SaleMetadataRange. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input checked="" type="checkbox"/>		Created sale from
DateTo	datetime	<input checked="" type="checkbox"/>		Created sale to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
CustomerAttributeID	int32	Customer Attribute ID	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Value	string	Value	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.26 SaleRange: Tixly Sale Range

Fetches customers by sale range, which means if the customer buys or is refunded an order in the date range it will be included. This endpoint is good to use if you are fetching from other endpoints by a specific date range, that can be matched in this endpoint to link the data. The dates are to and from and include time as well so if you would like an entire day then you need to indicate the end time of that day or the start of the next as the DateTo parameter.

Catalog: Tixly

Schema: Customer

Label: Sale Range

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function SaleRange. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input checked="" type="checkbox"/>		Created sale from
DateTo	datetime	<input checked="" type="checkbox"/>		Created sale to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

3.1.27 SaleTagsRange: Tixly Sale Tags Range

Fetches customers by sale range, which means if the customer buys or is refunded an order in the date range it will be included. This endpoint is good to use if you are fetching from other endpoints by a specific date range, that can be matched in this endpoint to link the data. The dates are to and from and include time as well so if you would like an entire day then you need to indicate the end time of that day or the start of the next as the DateTo parameter.

Catalog: Tixly

Schema: Customer

Label: Sale Tags Range

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function SaleTagsRange. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input checked="" type="checkbox"/>		Created sale from
DateTo	datetime	<input checked="" type="checkbox"/>		Created sale to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

4 Schema: Donation

4.1 Tables

4.1.1 DeletedMembershipDonations: Tixly Deleted Membership Donations

Fetches all deleted memberships you have access to.

Catalog: Tixly

Schema: Donation

Label: Deleted Membership Donations

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedMembershipDonations. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		
DateTo	datetime	<input type="checkbox"/>		

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Renew DayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
Renew MonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

4.1.2 Donations: Tixly Donations

Fetches all donatons that you have access to, pass in the SoldFrom and SoldTo to get only donations posted on a specific date range.

Catalog: Tixly

Schema: Donation

Label: Donations

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function Donations. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
DonationID	int32	Donation ID	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
isCancelled	boolean	Is Cancelled	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
RenewalType	string	Renewal Type	<input type="checkbox"/>	
RenewDayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
RenewMonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

4.1.3 DonationTypes: Tixly Donation Types

Fetches all the donation types you have access to.

Catalog: Tixly

Schema: Donation

Label: Donation Types

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Renew DayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
Renew MonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

5 Schema: Event

5.1 Tables

5.1.1 DeletedEvents: Tixly Deleted Events

Fetches all deleted events you have access to.

Catalog: Tixly

Schema: Event

Label: Deleted Events

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedEvents. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Events deleted from
DateTo	datetime	<input type="checkbox"/>		Events deleted to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	

5.1.2 EventAllocations: Tixly Event Allocations

Fetches all events you have access to.

Catalog: Tixly

Schema: Event

Label: Event Allocations

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventAllocations. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date
StartFrom	datetime	<input type="checkbox"/>		Get events that start after this date
StartTo	datetime	<input type="checkbox"/>		get events that start before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Block	boolean	Block	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
ExpiresHoursBefore	int32	Expires Hours Before	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllow ed	boolean	Online Reservation Allow ed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
ReservedTickets	int32	Reserved Tickets	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Seats	int32	Seats	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
SoldTickets	int32	Sold Tickets	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

5.1.3 EventAllocationsByEventId: Tixly Event Allocations by Event ID

Fetches all allocations for a specific events

Catalog: Tixly

Schema: Event

Label: Event Allocations by Event ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventAllocationsByEventId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		date of the specific event

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Block	boolean	Block	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
ExpiresHoursBefore	int32	Expires Hours Before	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
ReservedTickets	int32	Reserved Tickets	<input type="checkbox"/>	
Seats	int32	Seats	<input type="checkbox"/>	
SoldTickets	int32	Sold Tickets	<input type="checkbox"/>	

5.1.4 EventCategories: Tixly Event Categories

Fetches all events you have access to.

Catalog: Tixly

Schema: Event

Label: Event Categories

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `EventCategories`. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date
StartFrom	datetime	<input type="checkbox"/>		Get events that start after this date
StartTo	datetime	<input type="checkbox"/>		get events that start before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

5.1.5 EventCategories1: Tixly Event Categories

Fetches all events categories you have access to. The categories are returned with a list of IDs of the events they are connected to.

Catalog: Tixly

Schema: Event

Label: Event Categories

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

Table Columns

The columns of the table EventCategories1 are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

5.1.6 EventCategoriesData: Tixly Event Categories Data

Fetches all events categories you have access to. The categories are returned with a list of IDs of the events they are connected to.

Catalog: Tixly

Schema: Event

Label: Event Categories Data

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
TEXT	int32	Text	<input type="checkbox"/>	

5.1.7 EventCustomerPlusMetadataById: Tixly Event Customer plus Metadata by ID

Fetches the customers that have bought tickets in a specific event

Catalog: Tixly

Schema: Event

Label: Event Customer plus Metadata by ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventCustomerPlusMetadataById. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		date of the specific event

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
CustomerAttributeID	int32	Customer Attribute ID	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Value	string	Value	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

5.1.8 EventCustomersByEventId: Tixly Event Customers by Event ID

Fetches the customers that have bought tickets in a specific event

Catalog: Tixly

Schema: Event

Label: Event Customers by Event ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventCustomersByEventId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		date of the specific event

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

5.1.9 EventCustomerTagsById: Tixly Customer Tags by ID

Fetches the customers that have bought tickets in a specific event

Catalog: Tixly

Schema: Event

Label: Customer Tags by ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `EventCustomerTagsByld`. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		date of the specific event

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

5.1.10 EventDeletedSeasons: Tixly Event Deleted Seasons

Fetches all events categories you have access to. The categories are returned with a list of IDs of the events they are connected to.

Catalog: Tixly

Schema: Event

Label: Event Deleted Seasons

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventDeletedSeasons. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
dateFrom	datetime	<input type="checkbox"/>		
dateTo	datetime	<input type="checkbox"/>		

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
DateFrom	datetime	Date from	<input type="checkbox"/>	
DateFromUTCUnix	int64		<input type="checkbox"/>	
DateTo	datetime	Date to	<input type="checkbox"/>	
DateToUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

5.1.11 Events: Tixly Events

Fetches all events you have access to.

Catalog: Tixly

Schema: Event

Label: Events

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function Events. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date
StartFrom	datetime	<input type="checkbox"/>		Get events that start after this date
StartTo	datetime	<input type="checkbox"/>		get events that start before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

5.1.12 EventTags: Tixly Event Tags

Fetches all events you have access to.

Catalog: Tixly

Schema: Event

Label: Event Tags

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventTags. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date
StartFrom	datetime	<input type="checkbox"/>		Get events that start after this date
StartTo	datetime	<input type="checkbox"/>		get events that start before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllow ed	boolean	Online Reservation Allow ed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
Venueld	int32	Venue ID	<input type="checkbox"/>	

5.1.13 EventTags1: Tixly Event Tags

Fetches all events tags you have access to. The tags are returned with a list of IDs of the events they are connected to.

Catalog: Tixly

Schema: Event

Label: Event Tags

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

Table Columns

The columns of the table EventTags1 are shown below. Each column has an SQL data type.

Name	Data Type	Label	Required	Documentation
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

5.1.14 EventTagsData: Tixly Event Tags Data

Fetches all events tags you have access to. The tags are returned with a list of IDs of the events they are connected to.

Catalog: Tixly

Schema: Event

Label: Event Tags Data

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
TEXT	int32	Text	<input type="checkbox"/>	

5.1.15 EventTickets: Tixly Event Tickets

Fetches all the tickets that you have access to, pass in the SoldFrom and SoldTo to get tickets sold within a specific date range

Catalog: Tixly

Schema: Event

Label: Event Tickets

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventTickets. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Entrance	string	Entrance	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Fee	double	Fee	<input type="checkbox"/>	
IsAnonymousSale	boolean	Is Anonymous Sale	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
PriceZone	string	Price Zone	<input type="checkbox"/>	
PriceZoneId	int32	Price Zone ID	<input type="checkbox"/>	
Row	string	Row	<input type="checkbox"/>	
Scanned	boolean	Scanned	<input type="checkbox"/>	
Seat	string	Seat	<input type="checkbox"/>	
Section	string	Section	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	
SubscriptionTypeId	int32	Subscription Type ID	<input type="checkbox"/>	
TicketCount	int32	Ticket Count	<input type="checkbox"/>	
TicketId	int32	Ticket ID	<input type="checkbox"/>	
TicketType	string	Ticket Type	<input type="checkbox"/>	
TicketTypeId	int32	Ticket Type ID	<input type="checkbox"/>	

5.1.16 EventTicketsByEventId: Tixly Event Tickets by Event ID

Fetches tickets sold to a specific event

Catalog: Tixly

Schema: Event

Label: Event Tickets by Event ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `EventTicketsByEventId`. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		date of the specific event

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Entrance	string	Entrance	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Fee	double	Fee	<input type="checkbox"/>	
IsAnonymousSale	boolean	Is Anonymous Sale	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
PriceZone	string	Price Zone	<input type="checkbox"/>	
PriceZoneId	int32	Price Zone ID	<input type="checkbox"/>	
Row	string	Row	<input type="checkbox"/>	
Scanned	boolean	Scanned	<input type="checkbox"/>	
Seat	string	Seat	<input type="checkbox"/>	
Section	string	Section	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	
SubscriptionTypeId	int32	Subscription Type ID	<input type="checkbox"/>	
TicketCount	int32	Ticket Count	<input type="checkbox"/>	
TicketId	int32	Ticket ID	<input type="checkbox"/>	
TicketType	string	Ticket Type	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
TicketTypeId	int32	Ticket Type ID	<input type="checkbox"/>	

5.1.17 FutureEventAllocations: Tixly Future Event Allocations

Fetches events you have access to that have a start date in the future.

Catalog: Tixly

Schema: Event

Label: Future Event Allocations

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function FutureEventAllocations. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Block	boolean	Block	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
ExpiresHoursBefore	int32	Expires Hours Before	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
ReservedTickets	int32	Reserved Tickets	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Seats	int32	Seats	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
SoldTickets	int32	Sold Tickets	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

5.1.18 FutureEventCategories: Tixly Future Event Categories

Fetches events you have access to that have a start date in the future.

Catalog: Tixly

Schema: Event

Label: Future Event Categories

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function FutureEventCategories. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

5.1.19 FutureEvents: Tixly Future Events

Fetches events you have access to that have a start date in the future.

Catalog: Tixly

Schema: Event

Label: Future Events

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function FutureEvents. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
Venueld	int32	Venue ID	<input type="checkbox"/>	

5.1.20 FutureEventTags: Tixly Future Event Tags

Fetches events you have access to that have a start date in the future.

Catalog: Tixly

Schema: Event

Label: Future Event Tags

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function FutureEventTags. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
EditedFrom	datetime	<input type="checkbox"/>		Get events edited after this date
EditedTo	datetime	<input type="checkbox"/>		Get events edited before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

6 Schema: EventGroup

6.1 Tables

6.1.1 DeletedEventGroups: Tixly Deleted Event Groups

Fetches all deleted event groups you have access to.

Catalog: Tixly

Schema: EventGroup

Label: Deleted Event Groups

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedEventGroups. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Event groups deleted from
DateTo	datetime	<input type="checkbox"/>		Event groups deleted to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	

6.1.2 EventGroupById: Tixly Event Group by ID

Gets a specific event group by it's ID

Catalog: Tixly

Schema: EventGroup

Label: Event Group by ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventGroupById. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		ID of the specific event group

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	

6.1.3 EventGroupEventAllocationsByEventGroupId: Tixly Event Group Allocations by Event Group ID

Gets all the events within the event group

Catalog: Tixly

Schema: EventGroup

Label: Event Group Allocations by Event Group ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventGroupEventAllocationsByEventGroupId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		ID of the specific event group

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Block	boolean	Block	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
ExpiresHoursBefore	int32	Expires Hours Before	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
ReservedTickets	int32	Reserved Tickets	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Seats	int32	Seats	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
SoldTickets	int32	Sold Tickets	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

6.1.4 EventGroupEventCategoriesByEventGroupId: Tixly Event Group Categories by Event Group ID

Gets all the events within the event group

Catalog: Tixly

Schema: EventGroup

Label: Event Group Categories by Event Group ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventGroupEventCategoriesByEventGroupId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		ID of the specific event group

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllow ed	boolean	Online Reservation Allow ed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

6.1.5 EventGroupEventsByEventGroupId: Tixly Event Group Events by Event Group ID

Gets all the events within the event group

Catalog: Tixly

Schema: EventGroup

Label: Event Group Events by Event Group ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventGroupEventsByEventGroupId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		ID of the specific event group

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
Venueld	int32	Venue ID	<input type="checkbox"/>	

6.1.6 EventGroupEventTagsByEventGroupId: Tixly Event Group Tags by Event Group ID

Gets all the events within the event group

Catalog: Tixly

Schema: EventGroup

Label: Event Group Tags by Event Group ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function EventGroupEventTagsByEventGroupId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		ID of the specific event group

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Allocated	int32	Allocated	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
BlockedAllocated	int32	Blocked Allocated	<input type="checkbox"/>	
Capacity	int32	Capacity	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
DateConfirmed	boolean	Date Confirmed	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EndDate	datetime	End Date	<input type="checkbox"/>	
EndDateUTCUnix	int64		<input type="checkbox"/>	
EventGroup	string	Event Group	<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
HallId	int32	Hall ID	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsFreeEvent	boolean	Is Free Event	<input type="checkbox"/>	
IsNumbered	boolean	Is Numbered	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineReservationAllowed	boolean	Online Reservation Allowed	<input type="checkbox"/>	
Promoter	string	Promoter	<input type="checkbox"/>	
PromoterId	int32	Promoter ID	<input type="checkbox"/>	
Reserved	int32	Reserved	<input type="checkbox"/>	
SaleStatusId	int32	Sale Status ID	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
Sold	int32	Sold	<input type="checkbox"/>	
StartDate	datetime	Start Date	<input type="checkbox"/>	
StartDateUTCUnix	int64		<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	
Venue	string	Venue	<input type="checkbox"/>	
VenueId	int32	Venue ID	<input type="checkbox"/>	

6.1.7 EventGroups: Tixly Event Groups

Gets all the event groups this workgroup has access to.

Catalog: Tixly

Schema: EventGroup

Label: Event Groups

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
EventGroupId	int32	Event Group ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	

7 Schema: Giftcard

7.1 Tables

7.1.1 GiftCardRange: Tixly Gift Card Range

Fetches all giftcard sales, pass in the SoldFrom and SoldTo query parameters to get giftcards from a specific daterange

Catalog: Tixly

Schema: Giftcard

Label: Gift Card Range

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function GiftCardRange. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Amount	double	Amount	<input type="checkbox"/>	
AmountLeft	double	Amount Left	<input type="checkbox"/>	
AmountUsed	double	Amount Used	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Expires	datetime	Expires	<input type="checkbox"/>	
ExpiresUTCUnix	int64		<input type="checkbox"/>	
GiftCardCount	int32	Giftcard Count	<input type="checkbox"/>	
Number	string	Number	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Pin	string	PIN	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	

8 Schema: Membership

8.1 Tables

8.1.1 DeletedMemberships: Tixly Deleted Memberships

Fetches all deleted memberships you have access to.

Catalog: Tixly

Schema: Membership

Label: Deleted Memberships

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedMemberships. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		
DateTo	datetime	<input type="checkbox"/>		

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Renew DayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
Renew MonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

8.1.2 MembershipDonationTypes: Tixly Membership Donation Types

Fetches all membership types you have access to.

Catalog: Tixly

Schema: Membership

Label: Membership Donation Types

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Renew DayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
Renew MonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

8.1.3 Memberships: Tixly Memberships

Fetches all memberships that you have access to, pass in the SoldFrom and SoldTo to get only memberships sold on a specific date range.

Catalog: Tixly

Schema: Membership

Label: Memberships

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function Memberships. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Expires	datetime	Expires	<input type="checkbox"/>	
ExpiresUTCUnix	int64		<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
isCancelled	boolean	Is Cancelled	<input type="checkbox"/>	
MembershipID	int32	Membership ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
RenewalType	string	Renewal Type	<input type="checkbox"/>	
RenewDayOfMonth	int32	Renew Day of Month	<input type="checkbox"/>	
RenewMonthOfYear	int32	Renew Month of Year	<input type="checkbox"/>	
ShortDescription	string	Short Description	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	
Tagline	string	Tagline	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

9 Schema: Native

9.1 Tables

9.1.1 NATIVEPLATFORMSCALARREQUESTS: Tixly Native Platform Scalar Requests

{res:itgen_native_platform_scalar_requests_desc}

Catalog: Tixly

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 Tixly 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 Tixly 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.

Name	Data Type	Label	Required	Documentation
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.
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	Result Date Time	<input type="checkbox"/>	
RESULT_NUMBER	decimal	Result Number	<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"/>	

10 Schema: Permission

10.1 Tables

10.1.1 DeletedPermissions: Tixly Deleted Permissions

Fetches all deleted permission types.

Catalog: Tixly

Schema: Permission

Label: Deleted Permissions

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedPermissions. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Deleted permission types from
DateTo	datetime	<input type="checkbox"/>		Deleted permission types to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Deleted	datetime	Deleted	<input type="checkbox"/>	
DeletedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
EmailPermission	boolean	Email Permission	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
MobilePermission	boolean	Mobile Permission	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Type	int32	Type	<input type="checkbox"/>	

10.1.2 PermissionRegistrationsByPermissionId: Tixly Permission Registrations by Permission ID

Fetches all registrations to the specified permission.

Catalog: Tixly

Schema: Permission

Label: Permission Registrations by Permission ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function PermissionRegistrationsByPermissionId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		If this is set, then only get registrations after this date
DateTo	datetime	<input type="checkbox"/>		If this is set, then only get registrations before this date
id	int32	<input checked="" type="checkbox"/>		Id of the permission

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
PermissionId	int32	Permission ID	<input type="checkbox"/>	

10.1.3 PermissionRemovalsByPermissionId: Tixly Permission Removals by Permission ID

Fetches all removals from the specified permission.

Catalog: Tixly

Schema: Permission

Label: Permission Removals by Permission ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function `PermissionRemovalsByPermissionId`. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		If this is set, then only get registration removals after this date
DateTo	datetime	<input type="checkbox"/>		If this is set, then only get registration removals before this date
id	int32	<input checked="" type="checkbox"/>		Id of the permission

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
CustomerId	int32	Customer ID	<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
PermissionId	int32	Permission ID	<input type="checkbox"/>	
Removed	datetime	Removed	<input type="checkbox"/>	
RemovedUTCUnix	int64		<input type="checkbox"/>	

10.1.4 Permissions: Tixly Permissions

Fetches all permissions you have access to.

Catalog: Tixly

Schema: Permission

Label: Permissions

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
EmailPermission	boolean	Email Permission	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
MobilePermission	boolean	Mobile Permission	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Type	int32	Type	<input type="checkbox"/>	

11 Schema: Product

11.1 Tables

11.1.1 Products

Fetches all products you have access to.

Catalog: Tixly

Schema: Product

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function Products. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
GroupingText	string	Grouping Text	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
Vat	double	VAT	<input type="checkbox"/>	

11.1.2 SoldProducts: Tixly Sold Products

Get all the sold products. Pass in the SoldFrom and SoldTo parameters to get sales within a specific date range

Catalog: Tixly

Schema: Product

Label: Sold Products

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function SoldProducts. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
SoldFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
SoldTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
EventId	int32	Event ID	<input type="checkbox"/>	
Online	boolean	Online	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
Price	double	Price	<input type="checkbox"/>	
ProductCount	int32	Product Count	<input type="checkbox"/>	
ProductId	int32	Product ID	<input type="checkbox"/>	
SkinID	int32	Skin ID	<input type="checkbox"/>	

12 Schema: Subscription

12.1 Tables

12.1.1 Subscriptions: Tixly Subscriptions

Gets all subscription sales that you have access to

Catalog: Tixly

Schema: Subscription

Label: Subscriptions

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function Subscriptions. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		If this is set, then only get sales after this date
DateTo	datetime	<input type="checkbox"/>		If this is set, then only get sales before this date

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CustomerId	int32	Customer ID	<input type="checkbox"/>	
CustomerName	string	Customer Name	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OrderId	int32	Order ID	<input type="checkbox"/>	
OrganisationID	int32	Organization ID	<input type="checkbox"/>	
SubscriptionTypeId	int32	Subscription Type ID	<input type="checkbox"/>	

12.1.2 SubscriptionTypeEvents: Tixly Subscription Type Events

Gets all subscription types that you have access to

Catalog: Tixly

Schema: Subscription

Label: Subscription Type Events

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Allow Change	boolean	Allow Change	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsConcert	boolean	Is Concert	<input type="checkbox"/>	
MaxEvents	int32	Maximum Events	<input type="checkbox"/>	
MaxTickets	int32	Maximum Tickets	<input type="checkbox"/>	
MinEvents	int32	Minimum Events	<input type="checkbox"/>	
MinTickets	int32	Minimum Tickets	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineSaleEnd	datetime	Online Sale End	<input type="checkbox"/>	
OnlineSaleStart	datetime	Online Sale Start	<input type="checkbox"/>	
PickPerEvent	boolean	Pick per Event	<input type="checkbox"/>	
RenewalEndDate	datetime	Renewal End Date	<input type="checkbox"/>	
RenewalStartDate	datetime	Renewal Start Date	<input type="checkbox"/>	
ResourceName	string	Resource Name	<input type="checkbox"/>	
SameSeat	boolean	Same Seat	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	
SubTitle	string	Subtitle	<input type="checkbox"/>	

12.1.3 SubscriptionTypes: Tixly Subscription Types

Gets all subscription types that you have access to

Catalog: Tixly

Schema: Subscription

Label: Subscription Types

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Allow Change	boolean	Allow Change	<input type="checkbox"/>	
AvailableOnline	boolean	Available Online	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Hall	string	Hall	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
IsConcert	boolean	Is Concert	<input type="checkbox"/>	
MaxEvents	int32	Maximum Events	<input type="checkbox"/>	
MaxTickets	int32	Maximum Tickets	<input type="checkbox"/>	
MinEvents	int32	Minimum Events	<input type="checkbox"/>	
MinTickets	int32	Minimum Tickets	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
OnlineSaleEnd	datetime	Online Sale End	<input type="checkbox"/>	
OnlineSaleStart	datetime	Online Sale Start	<input type="checkbox"/>	
PickPerEvent	boolean	Pick per Event	<input type="checkbox"/>	
RenewalEndDate	datetime	Renewal End Date	<input type="checkbox"/>	
RenewalStartDate	datetime	Renewal Start Date	<input type="checkbox"/>	
ResourceName	string	Resource Name	<input type="checkbox"/>	
SameSeat	boolean	Same Seat	<input type="checkbox"/>	
Season	string	Season	<input type="checkbox"/>	
SeasonId	int32	Season ID	<input type="checkbox"/>	

13 Schema: Tag

13.1 Tables

13.1.1 CustomerDataByTagId: Tixly Customer Data by Tag ID

Gets all customers that have a specific customer tag

Catalog: Tixly

Schema: Tag

Label: Customer Data by Tag ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerDataByTagId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a `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
id	int32	<input checked="" type="checkbox"/>		Id of the specific customer tag to get

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created_1	datetime	Created 1	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix_1	int64		<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id_1	int32	ID 1	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

13.1.2 CustomerMetadataByTagId: Tixly Customer Metadata by Tag ID

Gets all customers that have a specific customer tag

Catalog: Tixly

Schema: Tag

Label: Customer Metadata by Tag ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomerMetadataByTagId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a `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
id	int32	<input checked="" type="checkbox"/>		Id of the specific customer tag to get

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
CustomerAttributeID	int32	Customer Attribute ID	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name_1	string	Name 1	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
Value	string	Value	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

13.1.3 CustomersByTagId: Tixly Customers by Tag ID

Gets all customers that have a specific customer tag

Catalog: Tixly

Schema: Tag

Label: Customers by Tag ID

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function CustomersByTagId. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
id	int32	<input checked="" type="checkbox"/>		Id of the specific customer tag to get

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
AddressOne	string	Address One	<input type="checkbox"/>	
AddressTwo	string	Address Two	<input type="checkbox"/>	
AlternativeEmails	string	Alternative Email Addresses	<input type="checkbox"/>	
City	string	City	<input type="checkbox"/>	
CompanyName	string	Company Name	<input type="checkbox"/>	
Country	string	Country	<input type="checkbox"/>	
CountryCode	string	Country Code	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
CRN	string	CRN	<input type="checkbox"/>	
DateOfBirth	datetime	Date of Birth	<input type="checkbox"/>	
DateOfBirthUTCUnix	int64		<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Description	string	Description	<input type="checkbox"/>	
Disallow Merge	boolean	Disallow Merge	<input type="checkbox"/>	
Edited	datetime	Edited	<input type="checkbox"/>	
EditedUTCUnix	int64		<input type="checkbox"/>	
Email	string	Email	<input type="checkbox"/>	
FirstName	string	First Name	<input type="checkbox"/>	
Gender	int32	Gender	<input type="checkbox"/>	
HomePhone	string	Home Phone	<input type="checkbox"/>	
HouseExtension	string	House Extension	<input type="checkbox"/>	
HouseNumber	string	House Number	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Initials	string	Initials	<input type="checkbox"/>	
IsAnonymized	boolean	Is Anonymized	<input type="checkbox"/>	
IsDeleted	boolean	Is Deleted	<input type="checkbox"/>	
LanguageId	int32	Language ID	<input type="checkbox"/>	
LastName	string	Last Name	<input type="checkbox"/>	
LastNamePrefix	string	Last Name Prefix	<input type="checkbox"/>	
Latitude	double	Latitude	<input type="checkbox"/>	

Name	Data Type	Label	Required	Documentation
Longitude	double	Longitude	<input type="checkbox"/>	
Mobile	string	Mobile	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
SSN	string	SSN	<input type="checkbox"/>	
StreetName	string	Street Name	<input type="checkbox"/>	
Title	string	Title	<input type="checkbox"/>	
WorkPhone	string	Work Phone	<input type="checkbox"/>	
ZipCode	string	ZIP-code	<input type="checkbox"/>	

13.1.4 DeletedTags: Tixly Deleted Tags

Fetches all deleted customer tags you have access to.

Catalog: Tixly

Schema: Tag

Label: Deleted Tags

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedTags. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		Deleted customer tag from
DateTo	datetime	<input type="checkbox"/>		Deleted customer tag to

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
Deleted	datetime	Deleted	<input type="checkbox"/>	
DeletedUTCUnix	int64		<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

13.1.5 Tags: Tixly Tags

Fetches all customer tags that you have access to.

Catalog: Tixly

Schema: Tag

Label: Tags

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Abbreviation	string	Abbreviation	<input type="checkbox"/>	
Color	string	Color	<input type="checkbox"/>	
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Icon	string	Icon	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	

14 Schema: Ticket

14.1 Tables

14.1.1 DeletedTicketTypes: Tixly Deleted Ticket Types

Fetches all deleted ticket types you have access to.

Catalog: Tixly

Schema: Ticket

Label: Deleted Ticket Types

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

Parameters of Table Function

The following parameters can be used to control the behaviour of the table function DeletedTicketTypes. 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 evaluated using their default values.

Value specification by position is done by listing all values from the first to the last needed value. For example: a ``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
DateFrom	datetime	<input type="checkbox"/>		
DateTo	datetime	<input type="checkbox"/>		

Columns of Table Function

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

Name	Data Type	Label	Required	Documentation
Default	boolean	Default	<input type="checkbox"/>	
DeletedDate	datetime	Deleted Date	<input type="checkbox"/>	
DeletedDateUTCUnix	int64		<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Position	int32	Position	<input type="checkbox"/>	
TicketTypeGroupId	int32	Ticket Type Group ID	<input type="checkbox"/>	
Type	int32	Type	<input type="checkbox"/>	
TypeName	string	Type Name	<input type="checkbox"/>	

14.1.2 TicketTypeGroups: Tixly Ticket Type Groups

Fetches all ticket type groups you have access to.

Catalog: Tixly

Schema: Ticket

Label: Ticket Type Groups

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Position	int32	Position	<input type="checkbox"/>	

14.1.3 TicketTypes: Tixly Ticket Types

Fetches all ticket types you have access to.

Catalog: Tixly

Schema: Ticket

Label: Ticket Types

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

Table Columns

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

Name	Data Type	Label	Required	Documentation
Created	datetime	Created	<input type="checkbox"/>	
CreatedUTCUnix	int64		<input type="checkbox"/>	
Default	boolean	Default	<input type="checkbox"/>	
Id	int32	ID	<input type="checkbox"/>	
Name	string	Name	<input type="checkbox"/>	
Position	int32	Position	<input type="checkbox"/>	
TicketTypeGroupId	int32	Ticket Type Group ID	<input type="checkbox"/>	
Type	int32	Type	<input type="checkbox"/>	
TypeName	string	Type Name	<input type="checkbox"/>	

Index

- A -

Abbreviation 18, 19, 27, 31, 38, 39, 45, 52, 54, 55, 56, 58, 60, 67, 78, 87, 95, 97, 107, 108, 109, 115, 116, 117, 118, 80, 81, 83, 87, 88, 90, 92

add-odata-mandatory-filters 2

Address One 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Address Two 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

AddressOne 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

AddressTwo 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Allocated 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

Allow Change 107, 108

AllowChange 107, 108

Alternative Email Addresses 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

AlternativeEmails 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Amount 25, 94

Amount Left 25, 94

Amount Used 25, 94

AmountLeft 25, 94

AmountUsed 25, 94

analysis-enforce-row-uniqueness 2

Anonymized 15

Anonymized Customers 15

AnonymizedCustomers 15

AnonymizedUTCUnix 15

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

Available Online 18, 19, 21, 22, 24, 27, 54, 55, 56, 58, 61, 70, 72, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93, 95, 97, 107, 108

AvailableOnline 18, 19, 21, 22, 24, 27, 54, 55, 56, 58, 61, 70, 72, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93, 95, 97, 107, 108

- B -

BLOB Preferred 99

BLOB_PREFERRED 99

Block 19, 58, 60, 78, 87

Blocked Allocated 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

BlockedAllocated 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

BOL_RESPONSE_CACHE_MAX_AGE_SEC 99

bulk-delete-page-size-rows 2

bulk-insert-page-size-bytes 2

bulk-insert-page-size-rows 2

- C -

Capacity 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

City 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Color 31, 38, 39, 45, 52, 67, 109, 115, 116

Company Name 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

CompanyName 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Content Type 99

CONTENT_TYPE 99

Country 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Country Code 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

CountryCode 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Created 1 19, 31, 39, 52, 58, 67, 78, 87, 109

Created_1 19, 31, 39, 52, 58, 67, 78, 87, 109

CreatedUTCUnix 16, 18, 19, 21, 22, 24, 25, 27, 28, 30, 31, 33, 34, 36, 37, 38, 39, 41, 45, 48, 50, 52, 55, 56, 58, 60, 61, 64, 66, 67, 70, 72, 75, 76, 78, 80, 81, 83, 86, 87, 88, 90, 92, 93, 94, 97, 101, 103, 104, 105, 106, 109, 111, 113, 116, 117, 118

CreatedUTCUnix_1 19, 31, 39, 52, 58, 67, 78, 87, 109

CRN 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Customer Attribute ID 28, 30, 42, 48, 64, 111

Customer by ID 16

Customer Data by Tag ID 109

Customer Donation Sales by Customer ID 18

Customer Event Sales Allocations by Customer ID 19

Customer Event Sales by Customer ID 21

Customer Event Sales Categories by Customer ID 22
 Customer Event Sales Tags by Customer ID 24
 Customer Gift Cards by ID 25
 Customer ID 15, 25, 33, 36, 37, 41, 55, 75, 76, 94, 97, 101, 102, 105, 106
 Customer Membership Sales by Customer ID 27
 Customer Metadata by ID 28
 Customer Metadata by Tag ID 111
 Customer Name 36, 106
 Customer plus Metadata 30
 Customer plus Tags 31
 Customer Product Sales by Customer ID 33
 Customer Subscription Sales by Customer ID 36
 Customer Tags 37
 Customer Tags by Customer ID 38, 39
 Customer Tags by ID 67
 Customer Tickets by Customer ID 41
 CustomerAttributeID 28, 30, 42, 48, 64, 111
 CustomerByID 16
 CustomerDataByTagId 109
 CustomerDonationSalesByCustomerId 18
 CustomerEventSalesAllocationsByCustomerId 19
 CustomerEventSalesByCustomerId 21
 CustomerEventSalesCategoriesByCustomerId 22
 CustomerEventSalesTagsByCustomerId 24
 CustomerGiftCardsById 25
 CustomerId 15, 25, 33, 36, 37, 41, 55, 75, 76, 94, 97, 101, 102, 105, 106
 CustomerMembershipSalesByCustomerId 27
 CustomerMetadataById 28
 CustomerMetadataByTagId 111
 CustomerName 36, 106
 CustomerPlusMetadata 30
 CustomerPlusTags 31
 CustomerProductSalesByCustomerId 33
 Customers 34
 Customers by Tag ID 113
 CustomersByTagId 113
 CustomerSubscriptionSalesByCustomerId 36
 CustomerTags 37
 CustomerTags2ByCustomerId 38
 CustomerTagsByCustomerId 39
 CustomerTicketsByCustomerId 41
 Date of Birth 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Date to 69
 DATE_ENDED 99
 DATE_STARTED 99
 DateConfirmed 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 DateFrom 15, 30, 31, 34, 42, 44, 45, 48, 50, 52, 54, 57, 69, 85, 95, 100, 101, 102, 106, 115, 116
 DateFromUTCUnix 69
 DateOfBirth 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 DateOfBirthUTCUnix 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 DateTo 15, 30, 31, 34, 42, 44, 45, 48, 50, 52, 54, 57, 69, 85, 95, 100, 101, 102, 106, 115, 116
 DateToUTCUnix 69
 Default 116, 118
 Deleted 100, 115
 Deleted Customer Tags 45
 Deleted Customers 44
 Deleted Customers plus Metadata 42
 Deleted Date 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 54, 57, 64, 66, 67, 69, 85, 95, 109, 111, 113, 116
 Deleted Event Groups 85
 Deleted Events 57
 Deleted Membership Donations 54
 Deleted Memberships 95
 Deleted Permissions 100
 Deleted Tags 115
 Deleted Ticket Types 116
 DeletedCustomerPlusMetadata 42
 DeletedCustomers 44
 DeletedCustomerTags 45
 DeletedDate 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 54, 57, 64, 66, 67, 69, 85, 95, 109, 111, 113, 116
 DeletedDateUTCUnix 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 54, 57, 64, 66, 67, 69, 85, 95, 109, 111, 113, 116
 DeletedEventGroups 85
 DeletedEvents 57
 DeletedMembershipDonations 54
 DeletedMemberships 95
 DeletedPermissions 100
 DeletedTags 115
 DeletedTicketTypes 116
 DeletedUTCUnix 100, 115
 Disallow Merge 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113
 DisallowMerge 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113

- D -

Database Driver 1
 Date Confirmed 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Date from 69

Donation ID	55	download-error-590-network-connect-timeout-sleep-max-ms	2
Donation Types	56	download-error-590-network-connect-timeout-sleep-multiplier	2
DonationID	55	download-error-599-network-connect-timeout-max-tries	2
Donations	55	download-error-599-network-connect-timeout-sleep-initial-ms	2
DonationTypes	56	download-error-599-network-connect-timeout-sleep-max-ms	2
download-error-400-bad-request-max-tries	2	download-error-599-network-connect-timeout-sleep-multiplier	2
download-error-400-bad-request-sleep-initial-ms	2	download-error-argument-exception-max-tries	2
download-error-400-bad-request-sleep-max-ms	2	download-error-argument-exception-sleep-initial-ms	2
download-error-400-bad-request-sleep-multiplier	2	download-error-argument-exception-sleep-max-ms	2
download-error-408-request-timeout-max-tries	2	download-error-argument-exception-sleep-multiplier	2
download-error-408-request-timeout-sleep-initial-ms	2	download-error-internet-down-max-tries	2
download-error-408-request-timeout-sleep-max-ms	2	download-error-internet-down-sleep-initial-ms	2
download-error-408-request-timeout-sleep-multiplier	2	download-error-internet-down-sleep-max-ms	2
download-error-422-bad-request-max-tries	2	download-error-internet-down-sleep-multiplier	2
download-error-422-bad-request-sleep-initial-ms	2	download-error-io-exception-max-tries	2
download-error-422-bad-request-sleep-max-ms	2	download-error-io-exception-sleep-initial-ms	2
download-error-422-bad-request-sleep-multiplier	2	download-error-io-exception-sleep-max-ms	2
download-error-429-too-many-requests-max-tries	2	download-error-io-exception-sleep-multiplier	2
download-error-429-too-many-requests-sleep-initial-ms	2	download-error-json-exception-max-tries	2
download-error-429-too-many-requests-sleep-max-ms	2	download-error-json-exception-sleep-initial-ms	2
download-error-429-too-many-requests-sleep-multiplier	2	download-error-json-exception-sleep-max-ms	2
download-error-502-server-unavailable-max-tries	2	download-error-json-exception-sleep-multiplier	2
download-error-502-server-unavailable-sleep-initial-ms	2	download-error-other-exception-max-tries	2
download-error-502-server-unavailable-sleep-max-ms	2	download-error-other-exception-sleep-initial-ms	2
download-error-502-server-unavailable-sleep-multiplier	2	download-error-other-exception-sleep-max-ms	2
download-error-503-server-unavailable-max-tries	2	download-error-other-exception-sleep-multiplier	2
download-error-503-server-unavailable-sleep-initial-ms	2	download-error-socket-exception-max-tries	2
download-error-503-server-unavailable-sleep-max-ms	2	download-error-socket-exception-sleep-initial-ms	2
download-error-503-server-unavailable-sleep-multiplier	2	download-error-socket-exception-sleep-max-ms	2
download-error-504-gateway-timeout-max-tries	2	download-error-socket-exception-sleep-multiplier	2
download-error-504-gateway-timeout-sleep-initial-ms	2	download-error-web-exception-max-tries	2
download-error-504-gateway-timeout-sleep-max-ms	2	download-error-web-exception-sleep-initial-ms	2
download-error-504-gateway-timeout-sleep-multiplier	2	download-error-web-exception-sleep-max-ms	2
download-error-590-network-connect-timeout-max-tries	2	download-error-web-exception-sleep-multiplier	2
download-error-590-network-connect-timeout-sleep-initial-ms	2	download-error-web-not-implemented-max-tries	2
		download-error-web-not-implemented-sleep-initial-ms	2
		download-error-web-not-implemented-sleep-max-ms	2
		download-error-web-not-implemented-sleep-multiplier	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-multiplier 2
 download-error-web-unauthorized-max-tries 2
 download-error-web-unauthorized-sleep-initial-ms 2
 download-error-web-unauthorized-sleep-max-ms 2
 download-error-web-unauthorized-sleep-multiplier 2
 DRY_RUN 99
 Duration (ms) 99
 DURATION_MS 99

- E -

Edited 16, 19, 21, 22, 24, 28, 30, 31, 34, 39, 48, 50, 52, 58, 61, 64, 66, 67, 70, 72, 78, 80, 81, 83, 86, 87, 88, 90, 92, 93, 104, 109, 111, 113
 EditedFrom 58, 61, 70, 72, 78, 80, 81, 83
 EditedTo 58, 61, 70, 72, 78, 80, 81, 83
 EditedUTCUnix 16, 19, 21, 22, 24, 28, 30, 31, 34, 39, 48, 50, 52, 58, 61, 64, 66, 67, 70, 72, 78, 80, 81, 83, 86, 87, 88, 90, 92, 93, 104, 109, 111, 113
 Email 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 101, 102, 109, 111, 113
 Email Permission 100, 103
 EmailPermission 100, 103
 End Date 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92, 99
 EndDate 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 EndDateUTCUnix 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Entrance 41, 75, 76
 Error Message Code 99
 Error Message Text 99
 ERROR_MESSAGE_CODE 99
 ERROR_MESSAGE_TEXT 99
 Event Allocations 58
 Event Allocations by Event ID 60
 Event Categories 61, 63
 Event Categories Data 63
 Event Customer plus Metadata by ID 64
 Event Customers by Event ID 66
 Event Deleted Seasons 69
 Event Group 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Event Group Allocations by Event Group ID 87
 Event Group by ID 86
 Event Group Categories by Event Group ID 88
 Event Group Events by Event Group ID 90
 Event Group ID 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93
 Event Group Tags by Event Group ID 92
 Event Groups 93
 Event ID 33, 41, 75, 76, 101, 102, 105
 Event Tags 72, 74
 Event Tags Data 74
 Event Tickets 75
 Event Tickets by Event ID 76
 EventAllocations 58
 EventAllocationsByEventId 60
 EventCategories 61
 EventCategories1 63
 EventCategoriesData 63
 EventCustomerPlusMetadataById 64
 EventCustomersByEventId 66
 EventCustomerTagsById 67
 EventDeletedSeasons 69
 EventGroup 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 EventGroupById 86
 EventGroupEventAllocationsByEventGroupId 87
 EventGroupEventCategoriesByEventGroupId 88
 EventGroupEventsByEventGroupId 90
 EventGroupEventTagsByEventGroupId 92
 EventGroupId 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93
 EventGroups 93
 EventId 33, 41, 75, 76, 101, 102, 105
 Events 70
 EventTags 72
 EventTags1 74
 EventTagsData 74
 EventTickets 75
 EventTicketsByEventId 76
 Expires 25, 94, 97
 Expires Hours Before 19, 58, 60, 78, 87
 ExpiresHoursBefore 19, 58, 60, 78, 87
 ExpiresUTCUnix 25, 94, 97

- F -

Fail on Error 99
 FAIL_ON_ERROR 99
 fail-on-duplicate-rows 2
 Fee 41, 75, 76
 First Name 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 FirstName 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 force-case-sensitive-identifiers 2
 forced-casing-identifiers 2
 Future Event Allocations 78
 Future Event Categories 80
 Future Event Tags 83

Future Events 81

FutureEventAllocations 78

FutureEventCategories 80

FutureEvents 81

FutureEventTags 83

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

- G -

Gender 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

Gift Card Range 94

Giftcard Count 25, 94

GiftCardCount 25, 94

GiftCardRange 94

Grouping Text 104

GroupingText 104

Icon 31, 38, 39, 45, 52, 67, 109, 115, 116

id 16, 18, 19, 21, 22, 24, 25, 27, 28, 33, 36, 38, 39, 41, 60, 64, 66, 67, 76, 86, 87, 88, 90, 92, 101, 102, 109, 111, 113

ID 1 19, 22, 24, 31, 39, 45, 52, 58, 61, 67, 72, 78, 80, 83, 87, 88, 92, 107, 109

Id_1 19, 22, 24, 31, 39, 45, 52, 58, 61, 67, 72, 78, 80, 83, 87, 88, 92, 107, 109

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

Initials 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

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-multiplier 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-multiplier 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

Is Anonymized 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113

Is Anonymous Sale 41, 75, 76

Is Cancelled 55, 97

Is Concert 107, 108

- H -

Hall 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92, 107, 108

Hall ID 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

HallId 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

Home Phone 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

HomePhone 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

House Extension 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

House Number 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

HouseExtension 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

HouseNumber 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

HTTP Disk Cache Maximum Age (sec) 99

HTTP Memory Cache Maximum Age (sec) 99

HTTP Method 99

HTTP Status Code 99

HTTP_DISK_CACHE_MAX_AGE_SEC 99

HTTP_DISK_CACHE_SAVE 99

HTTP_DISK_CACHE_USE 99

HTTP_MEMORY_CACHE_MAX_AGE_SEC 99

HTTP_MEMORY_CACHE_SAVE 99

HTTP_MEMORY_CACHE_USE 99

HTTP_METHOD 99

HTTP_STATUS_CODE 99

http-disk-cache-compression-level 2

http-disk-cache-directory 2

http-disk-cache-ignore-write-errors 2

Is Deleted 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Is Free Event 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Is Numbered 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 IsAnonymized 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113
 IsAnonymousSale 41, 75, 76
 isCancelled 55, 97
 IsConcert 107, 108
 IsDeleted 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113
 IsFreeEvent 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 IsNumbered 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

- J -

join-set-points-per-request 2

- L -

Language ID 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Languageld 16, 28, 30, 31, 34, 39, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Last Name 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Last Name Prefix 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 LastName 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 LastNamePrefix 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Latitude 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 limit-partition-calls-left 2
 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
 Longitude 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

- M -

Master Customer ID 47, 48
 MasterCustomerId 47, 48
 MaxEvents 107, 108

Maximum Events 107, 108
 Maximum Tickets 107, 108
 maximum-discovered-column-count 2
 maximum-length-identifiers 2
 max-odata-filters 2
 MaxTickets 107, 108
 max-url-length-accepted 2
 max-url-length-desired 2
 Membership Donation Types 97
 Membership ID 97
 MembershipDonationTypes 97
 MembershipID 97
 Memberships 97
 Merged 47, 48
 MergedCustomerOperationContents 47
 MergedCustomerOperations 48
 MergedUTCUnix 47, 48
 metadata-cache-max-age-sec 2
 MinEvents 107, 108
 Minimum Events 107, 108
 Minimum Tickets 107, 108
 MinTickets 107, 108
 Mobile 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 101, 102, 109, 111, 113
 Mobile Permission 100, 103
 MobilePermission 100, 103

- N -

Name 16, 18, 19, 21, 22, 24, 27, 28, 30, 31, 34, 36, 38, 39, 42, 44, 45, 48, 50, 52, 54, 55, 56, 57, 58, 60, 61, 63, 64, 66, 67, 69, 70, 72, 74, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93, 95, 97, 100, 101, 103, 104, 106, 107, 108, 109, 111, 113, 115, 116, 117, 118
 Name 1 19, 22, 24, 28, 30, 31, 39, 42, 45, 48, 52, 58, 61, 64, 67, 72, 78, 80, 83, 87, 88, 92, 107, 109, 111
 Name_1 19, 22, 24, 28, 30, 31, 39, 42, 45, 48, 52, 58, 61, 64, 67, 72, 78, 80, 83, 87, 88, 92, 107, 109, 111
 Native Platform Scalar Requests 99
 NATIVEPLATFORMSCALARREQUESTS 99
 npt 99
 Number 25, 94

- O -

oauth-unauthorized-max-tries 2
 oauth-unauthorized-sleep-initial-ms 2
 oauth-unauthorized-sleep-max-ms 2
 oauth-unauthorized-sleep-multiplicator 2

Online 25, 33, 41, 75, 76, 94, 105
 Online Reservation Allowed 19, 21, 22, 24, 58, 61, 83, 87, 88, 90, 92
 Online Sale End 107, 108
 Online Sale Start 107, 108
 OnlineReservationAllowed 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 OnlineSaleEnd 107, 108
 OnlineSaleStart 107, 108
 Order ID 25, 33, 36, 41, 55, 75, 76, 94, 97, 101, 102, 105, 106
 OrderId 25, 33, 36, 41, 55, 75, 76, 94, 97, 101, 102, 105, 106
 OrganisationID 25, 33, 36, 41, 55, 75, 76, 94, 97, 105, 106
 Organization ID 25, 33, 36, 41, 55, 75, 76, 94, 97, 105, 106
 ORIG_SYSTEM_GROUP 99
 ORIG_SYSTEM_REFERENCE 99
 Original System Group 99
 Original System Reference 99

- P -

partition-slot-based-rate-limit-length-ms 2
 partition-slot-based-rate-limit-slots 2
 Payload 99
 PAYLOAD_TEXT 99
 Permission ID 101, 102
 Permission Registrations by Permission ID 101
 Permission Removals by Permission ID 102
 PermissionId 101, 102
 PermissionRegistrationsByPermissionId 101
 PermissionRemovalsByPermissionId 102
 Permissions 103
 Pick per Event 107, 108
 PickPerEvent 107, 108
 PIN 25, 94
 Position 116, 117, 118
 pre-request-delay-ms 2
 Price 18, 25, 27, 33, 41, 54, 55, 56, 75, 76, 94, 95, 97, 104, 105
 Price Zone 41, 75, 76
 Price Zone ID 41, 75, 76
 PriceZone 41, 75, 76
 PriceZoneId 41, 75, 76
 Product Count 33, 105
 Product ID 33, 105
 ProductCount 33, 105
 ProductId 33, 105
 Products 104

Promoter 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Promoter ID 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 PromoterId 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

- R -

Removed 102
 RemovedUTCUnix 102
 Renew Day of Month 18, 27, 54, 55, 56, 95, 97
 Renew Month of Year 18, 27, 54, 55, 56, 95, 97
 Renewal End Date 107, 108
 Renewal Start Date 107, 108
 Renewal Type 55, 97
 RenewalEndDate 107, 108
 RenewalStartDate 107, 108
 RenewalType 55, 97
 RenewDayOfMonth 18, 27, 54, 55, 56, 95, 97
 RenewMonthOfYear 18, 27, 54, 55, 56, 95, 97
 requested-page-size 2
 requests-parallel-max 2
 Reserved 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Reserved Tickets 19, 58, 60, 78, 87
 ReservedTickets 19, 58, 60, 78, 87
 Resource Name 107, 108
 ResourceName 107, 108
 Response Cache Maximum Age (sec) 99
 Result BLOB 99
 Result Date Time 99
 Result Number 99
 Result Text 99
 RESULT_BLOB 99
 RESULT_DATE_TIME_UTC 99
 RESULT_NUMBER 99
 RESULT_TEXT 99
 Row 41, 75, 76
 Run without Actions 99

- S -

Sale Metadata Range 48
 Sale Range 50
 Sale Status ID 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Sale Tags Range 52
 SaleMetadataRange 48
 SaleRange 50

SaleStatusId 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 SaleTagsRange 52
 Same Seat 107, 108
 SameSeat 107, 108
 Save HTTP Disk Cache 99
 Save HTTP Memory Cache 99
 Scanned 41, 75, 76
 Season 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93, 107, 108
 Season ID 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92, 107, 108
 SeasonId 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 83, 87, 88, 90, 92, 107, 108
 Seat 41, 75, 76
 Seats 19, 58, 60, 78, 87
 Section 41, 75, 76
 Short Description 18, 27, 54, 55, 56, 95, 97
 ShortDescription 18, 27, 54, 55, 56, 95, 97
 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
 simulate-http-500-errors 2
 simulate-http-500-errors-percentage 2
 simulate-http-502-errors 2
 simulate-http-502-errors-percentage 2
 simulate-http-503-errors 2
 simulate-http-503-errors-percentage 2
 simulate-http-protocol-errors 2
 simulate-http-protocol-errors-percentage 2
 simulate-http-timeout-errors 2
 simulate-http-timeout-errors-percentage 2
 Skin ID 25, 33, 41, 55, 75, 76, 94, 97, 105
 SkinID 25, 33, 41, 55, 75, 76, 94, 97, 105
 slot-based-rate-limit-length-ms 2
 slot-based-rate-limit-slots 2
 Sold 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 Sold Products 105
 Sold Tickets 19, 58, 60, 78, 87
 SoldFrom 55, 58, 61, 70, 72, 75, 94, 97, 104, 105
 SoldProducts 105
 SoldTickets 19, 58, 60, 78, 87
 SoldTo 55, 58, 61, 70, 72, 75, 94, 97, 104, 105
 SSN 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 standardize-identifiers 2
 standardize-identifiers-casing 2
 Start Date 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92, 99
 StartDate 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 StartDateUTCUnix 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
 StartFrom 58, 61, 70, 72
 StartTo 58, 61, 70, 72
 Street Name 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 StreetName 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
 Subscription Type Events 107
 Subscription Type ID 36, 41, 75, 76, 106
 Subscription Types 108
 Subscriptions 106
 SubscriptionTypeEvents 107
 SubscriptionTypeId 36, 41, 75, 76, 106
 SubscriptionTypes 108
 SubTitle 19, 21, 22, 24, 57, 58, 61, 70, 72, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93, 107
 Successful 99
 SUCCESSFUL 99
 swagger-specification-file 2
 swagger-specification-http-disk-cache-max-age-sec 2
 swagger-specification-url 2

- T -

Tag ID 37
 TagId 37
 Tagline 18, 27, 54, 55, 56, 95, 97
 Tags 116
 TEXT 47, 63, 74
 Ticket Count 41, 75, 76
 Ticket ID 41, 75, 76
 Ticket Type 41, 75, 76
 Ticket Type Group ID 116, 118
 Ticket Type Groups 117
 Ticket Type ID 41, 75, 76
 Ticket Types 118
 TicketCount 41, 75, 76
 TicketId 41, 75, 76
 TicketType 41, 75, 76
 TicketTypeGroupId 116, 118
 TicketTypeGroups 117
 TicketTypeId 41, 75, 76

TicketTypes 118
Timeout (sec) 99
TIMEOUT_SEC 99
Title 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 100, 103, 109, 111, 113
Tixly 1, 15, 16, 18, 19, 21, 22, 24, 25, 27, 28, 30, 31, 33, 34, 36, 37, 38, 39, 41, 42, 44, 45, 47, 48, 50, 52, 54, 55, 56, 57, 58, 60, 61, 63, 64, 66, 67, 69, 70, 72, 74, 75, 76, 78, 80, 81, 83, 85, 86, 87, 88, 90, 92, 93, 94, 95, 97, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 111, 113, 115, 116, 117, 118
Transaction ID 99
TRANSACTION_ID 99
Type 100, 103, 116, 118
Type Name 116, 118
TypeName 116, 118

- U -

URL 99
Use HTTP Disk Cache 99
Use HTTP Memory Cache 99
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

- V -

Value 28, 30, 42, 48, 64, 111
VAT 18, 19, 21, 22, 24, 27, 54, 55, 56, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92, 95, 97, 104
Venue 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
Venue ID 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92
Venueld 19, 21, 22, 24, 58, 61, 70, 72, 78, 80, 81, 83, 87, 88, 90, 92

- W -

Work Phone 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113
WorkPhone 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113

- Z -

ZipCode 16, 28, 30, 31, 34, 39, 42, 44, 45, 48, 50, 52, 64, 66, 67, 109, 111, 113



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