


# OS Data Model

*for use with Invantive SQL*

24.0

invantive
the SQL company

# Copyright

(C) Copyright 2004-2025 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 an 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 OS 1

2 SQL Driver Attributes 1

3 Catalog: FileSystem 4

3.1 Schemas 4
3.1.1 csv_split_row : OS Split Row as CSV 4
3.1.2 directories: OS File System Directories 6
3.1.3 file_actions: OS File Actions 7
3.1.4 file_copy_actions: OS File Copy Actions 8
3.1.5 file_delete_actions: OS File Delete Actions 8
3.1.6 file_info: OS File Metadata Properties 9
3.1.7 file_move_actions: OS File Move Actions 10
3.1.8 file_rename_actions: OS File Rename Actions 10
3.1.9 files: OS File System Files 11
3.1.10 read_file: OS File Binary Contents 12
3.1.11 read_file_text: OS File Text Contents 13
3.1.12 regexp_split_row : OS Split Row on Regular Expression 14
3.1.13 w rite_file: OS Write File 16

4 Package: dcr_metadata 16

4.1 Procedures 16
4.1.1 dcr_metadata.get_partitions: OS Data container metadata package 16

Index 17


*(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.*

# 1 SQL Driver for OS

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.

Access data through the operating system.

The OS driver covers 13 tables and 162 columns.

## OS Clients

Invantive UniversalSQL is available on many user interfaces ("clients" in traditional server-client paradigma). All Invantive UniversalSQL 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.

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.

## Specifications

The SQL driver for OS does not support partitioning.

An introduction into the concepts of Invantive UniversalSQL such as databases, data containers and partitioning can be found in the Invantive UniversalSQL 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.

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

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

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

Alias: `os`

Recommended alias: `os`

Technical support (URL): https://forums.invantive.com

Function documentation (URL): https://cloud.invantive.com/data-models

Function support (URL): https://forums.invantive.com

# 2 SQL Driver Attributes

The SQL driver for OS has many attributes that can be finetuned to improve handling in scenarios with unreliable network connections to the OS server or high volumes of data.

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

Also, many drivers have driver-specific attributes to finetune actual behaviour or handle data not matching specifications.

The OS 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 OS 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 OS 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 |
|---|---|---|---|---|---|---|---|
| analysis-enforce-row-uniqueness | Enforce rows to be unique for software analysis. A fingerprint is calculated from the whole row of data when the primary key column is unknown. | Shared | False | ✓ | ✓ | ✓ | |
| 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 | 10000 | ✓ | ✓ | ✓ | |
| force-case-sensitive-identifiers | Consider identifiers as case-sensitive independent of the platform capabilities. | Shared | False | ✓ | ✓ | ✓ | |

*(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.*

| Code | Description | Origin | Default Value | Set from Connection String | Set from Set SQL-Statement | Set from Drivers File | Set from Log On |
|---|---|---|---|---|---|---|---|
| forced-casing-identifiers | Forced casing of identifiers. Choose from: Unset, Lower, Upper and Mixed. | Shared | | ✓ | ✓ | ✓ | |
| 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-execution-profile-disk-path | {res:itgen_pae_invantive_sql_execution_profile_disk_path} | SQL Engine V1 | | ✓ | ✓ | ✓ | |
| invantive-sql-execution-profile-to-disk | {res:itgen_pae_invantive_sql_execution_profile_to_disk} | 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 | ✓ | ✓ | ✓ | |
| 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-length-identifiers | Non-default maximum length in characters of identifier names. | Shared | | ✓ | ✓ | ✓ | |
| partition-slot-based-rate-limit-length-ms | Total length in milliseconds across all slots of a partition-based rate limit. | Shared | 60000 | ✓ | | ✓ | |

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

| Code | Description | Origin | Default Value | Set from Connection String | Set from Set SQL-Statement | Set from Drivers File | Set from Log On |
|---|---|---|---|---|---|---|---|
| partition-slot-based-rate-limit-slots | Number of slots per partition-based rate limit. Null means no slot-based rate limit. | Shared | | ✓ | | ✓ | |
| pre-request-delay-ms | Pre-request delay in milliseconds per request. | Shared | 0 | ✓ | ✓ | ✓ | |
| requested-maximum-usable-partitions | Requested Maximum Usable Partitions | Shared | | ✓ | ✓ | ✓ | |
| 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 | ✓ | ✓ | ✓ | |
| 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 | Rewrite all identifiers to the preferred standards as configured by standardize-identifiers-casing and maximum-length-identifiers. | Shared | True | ✓ | ✓ | ✓ | |
| standardize-identifiers-casing | Rewrite all identifiers to the recommended standard platform-specific casing when changing a data model on a case-dependent platform. | Shared | True | ✓ | ✓ | ✓ | |

# 3 Catalog: FileSystem

## 3.1 Schemas

### 3.1.1 csv_split_row: OS Split Row as CSV

Catalog: FileSystem

Label: Split Row as CSV

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function csv_split_row. 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

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

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 |
|---|---|---|---|---|
| csv | varchar2 | ☑ | | Text in comma-separated format to split into individual columns. |
| max_entries_per_row | int32 | ☐ | | Maximum number of individual columns to return from the text. Remaining content is included in the last columns. Defaults to null (no limit). |

## Columns of Table Function

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| text_content_1 | varchar2 | CSV Text Content #{0} | | |
| text_content_10 | varchar2 | CSV Text Content #{0} | | |
| text_content_11 | varchar2 | CSV Text Content #{0} | | |
| text_content_12 | varchar2 | CSV Text Content #{0} | | |
| text_content_13 | varchar2 | CSV Text Content #{0} | | |
| text_content_14 | varchar2 | CSV Text Content #{0} | | |
| text_content_15 | varchar2 | CSV Text Content #{0} | | |
| text_content_16 | varchar2 | CSV Text Content #{0} | | |
| text_content_17 | varchar2 | CSV Text Content #{0} | | |
| text_content_18 | varchar2 | CSV Text Content #{0} | | |
| text_content_19 | varchar2 | CSV Text Content #{0} | | |
| text_content_2 | varchar2 | CSV Text Content #{0} | | |
| text_content_20 | varchar2 | CSV Text Content #{0} | | |
| text_content_21 | varchar2 | CSV Text Content #{0} | | |
| text_content_22 | varchar2 | CSV Text Content #{0} | | |
| text_content_23 | varchar2 | CSV Text Content #{0} | | |
| text_content_24 | varchar2 | CSV Text Content #{0} | | |
| text_content_25 | varchar2 | CSV Text Content #{0} | | |
| text_content_26 | varchar2 | CSV Text Content #{0} | | |
| text_content_27 | varchar2 | CSV Text Content #{0} | | |
| text_content_28 | varchar2 | CSV Text Content #{0} | | |
| text_content_29 | varchar2 | CSV Text Content #{0} | | |
| text_content_3 | varchar2 | CSV Text Content #{0} | | |
| text_content_30 | varchar2 | CSV Text Content #{0} | | |

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| text_content_31 | varchar2 | CSV Text Content #{0} | | |
| text_content_32 | varchar2 | CSV Text Content #{0} | | |
| text_content_33 | varchar2 | CSV Text Content #{0} | | |
| text_content_34 | varchar2 | CSV Text Content #{0} | | |
| text_content_35 | varchar2 | CSV Text Content #{0} | | |
| text_content_36 | varchar2 | CSV Text Content #{0} | | |
| text_content_37 | varchar2 | CSV Text Content #{0} | | |
| text_content_38 | varchar2 | CSV Text Content #{0} | | |
| text_content_39 | varchar2 | CSV Text Content #{0} | | |
| text_content_4 | varchar2 | CSV Text Content #{0} | | |
| text_content_40 | varchar2 | CSV Text Content #{0} | | |
| text_content_41 | varchar2 | CSV Text Content #{0} | | |
| text_content_42 | varchar2 | CSV Text Content #{0} | | |
| text_content_43 | varchar2 | CSV Text Content #{0} | | |
| text_content_44 | varchar2 | CSV Text Content #{0} | | |
| text_content_45 | varchar2 | CSV Text Content #{0} | | |
| text_content_46 | varchar2 | CSV Text Content #{0} | | |
| text_content_47 | varchar2 | CSV Text Content #{0} | | |
| text_content_48 | varchar2 | CSV Text Content #{0} | | |
| text_content_49 | varchar2 | CSV Text Content #{0} | | |
| text_content_5 | varchar2 | CSV Text Content #{0} | | |
| text_content_50 | varchar2 | CSV Text Content #{0} | | |
| text_content_6 | varchar2 | CSV Text Content #{0} | | |
| text_content_7 | varchar2 | CSV Text Content #{0} | | |
| text_content_8 | varchar2 | CSV Text Content #{0} | | |
| text_content_9 | varchar2 | CSV Text Content #{0} | | |

#### 3.1.2 directories: OS File System Directories

Catalog: FileSystem

Label: File System Directories

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function directories. 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

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

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 |
|---|---|---|---|---|
| all_directories | char | ☑ | | Whether to recurse into folders. Defaults to false. |
| path | varchar2 | ☑ | | Root of folder structure to search. |
| search_pattern | varchar2 | ☑ | | Directory name match pattern. Use '*' to signal text of arbitrary length and '?' as any single character. |

## Columns of Table Function

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| directory_path | varchar2 | Directory Path | ☑ | |

### 3.1.3 file_actions: OS File Actions

Catalog: FileSystem

Label: File Actions

Documentation:

File actions executed.

Retrieve: true

## Table Columns

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| action | varchar2 | Action | ☑ | Type of action. |
| duration_ms | int32 | Duration (ms) | ☐ | Duration of action in milliseconds. |
| exception_code | varchar2 | Exception Code | ☐ | The message code of the error. |
| exception_message | varchar2 | Exception Message | ☐ | The text of the error. |
| file_path_source | varchar2 | File Name and Path for Source | ☐ | File path of source for action. |
| file_path_target | varchar2 | File Name and Path for Target | ☐ | File path of targt for action. |
| has_exception | varchar2 | Has Exception | ☐ | Whether an exception occurred during execution. |

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| start_time | datetime | Start Time | ☐ | Start time of action. |

#### 3.1.4 file_copy_actions: OS File Copy Actions

Catalog: FileSystem

Label: File Copy Actions

Documentation:

Copies a file on file system.

Retrieve: false

## Table Columns

The columns of the table file_copy_actions 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 |
|---|---|---|---|---|
| create_directory | varchar2 | Create directory | ☐ | Whether to create the directory w hen non-existing. |
| file_path_source | varchar2 | File Name and Path for Source | ☑ | Path of file to copy. |
| file_path_target | varchar2 | File Name and Path for Target | ☑ | Path of new file. |
| ignore_error | varchar2 | Ignore Error | ☐ | Whether to ignore an error. |
| overw rite_existing | varchar2 | Overw rite Existing | ☐ | Whether to overw rite an existing file. |

#### 3.1.5 file_delete_actions: OS File Delete Actions

Catalog: FileSystem

Label: File Delete Actions

Documentation:

Deletes specific files on file system.

Retrieve: false

## Table Columns

The columns of the table file_delete_actions 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 |
|---|---|---|---|---|
| file_path | varchar2 | Filename and Path | ☑ | Path of file to delete. |

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

### 3.1.6 file_info: OS File Metadata Properties

Catalog: FileSystem

Label: File Metadata Properties

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function file_info. 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 |
|---|---|---|---|---|
| ignore_errors | varchar2 | ☐ | | Whether to ignore errors w hen accessing the file. Defaults to false. |
| path | varchar2 | ☑ | | Relative or absolute path to the file. |

## Columns of Table Function

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| created_utc | datetime | Created (UTC) | ☐ | |
| created | datetime | Created | ☐ | |
| directory_name | varchar2 | Directory Name | ☐ | |
| exception_code | varchar2 | Exception Code | ☐ | The message code of the error. |
| exception_message | varchar2 | Exception Message | ☐ | The text of the error. |
| extension | varchar2 | Extension | ☐ | |
| is_archive | varchar2 | Is Archive File | ☐ | |
| is_compressed | varchar2 | Is Compressed File | ☐ | |
| is_content_indexed | varchar2 | Is Content Indexed File | ☐ | |
| is_device | varchar2 | Is Device File | ☐ | |
| is_directory | varchar2 | Is Directory | ☐ | |
| is_encrypted | varchar2 | Is Encrypted File | ☐ | |
| is_existing | varchar2 | Is Existing | ☐ | |

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| is_hidden | varchar2 | Is Hidden File | ☐ | |
| is_integrity_stream | varchar2 | Is Integrity Stream File | ☐ | |
| is_normal | varchar2 | Is Normal File | ☐ | |
| is_offline | varchar2 | Is Offline File | ☐ | |
| is_reparse_point | varchar2 | Is Reparse Point File | ☐ | |
| is_scrub_data | varchar2 | Is Scrub Data File | ☐ | |
| is_sparse | varchar2 | Is Sparse File | ☐ | |
| is_system | varchar2 | Is System File | ☐ | |
| is_temporary | varchar2 | Is Temporary File | ☐ | |
| is_writable | varchar2 | Is Writable File | ☐ | |
| last_access_utc | datetime | Last Access (UTC) | ☐ | |
| last_access | datetime | Last Access | ☐ | |
| last_write_utc | datetime | Last Write (UTC) | ☐ | |
| last_write | datetime | | ☐ | |
| length | number | File Size (bytes) | ☐ | |
| name | varchar2 | Name | ☐ | |

### 3.1.7 file_move_actions: OS File Move Actions

Catalog: FileSystem

Label: File Move Actions

Documentation:

Moves specific files on file system.

Retrieve: false

## Table Columns

The columns of the table file_move_actions 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 |
|---|---|---|---|---|
| file_path_source | varchar2 | File Name and Path for Source | ☑ | Path of source file to move. |
| file_path_target | varchar2 | File Name and Path for Target | ☑ | Path of target file. |

### 3.1.8 file_rename_actions: OS File Rename Actions

Catalog: FileSystem

Label: File Rename Actions

Documentation:

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

Renames specific files on file system.

Retrieve: false

## Table Columns

The columns of the table file_rename_actions 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 |
|---|---|---|---|---|
| file_path_source | varchar2 | File Name and Path for Source | ☑ | Path of file to rename. |
| file_path_target | varchar2 | File Name and Path for Target | ☑ | Path of file after rename. |

#### 3.1.9 files: OS File System Files

Catalog: FileSystem

Label: File System Files

Documentation:

Scan file system for files matching the specifications.

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function files. 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 |
|---|---|---|---|---|
| all_directories | char | ☑ | | Whether to recurse into folders. Defaults to false. |
| ignore_case | char | ☑ | | Ignore case. |
| ignore_unauthorized_exception | char | ☑ | | Whether to ignore unauthorized exceptions. |
| path | varchar2 | ☑ | | Root of folder structure to search. |
| search_pattern | varchar2 | ☑ | | File name match pattern. Use '*' to signal text of arbitrary length |

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

| Name | Data Type | Required | Default Value | Documentation |
|---|---|---|---|---|
| | | | | and '?' as any single character. |

## Columns of Table Function

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| file_path | varchar2 | Filename and Path | ☑ | |

#### 3.1.10 read_file: OS File Binary Contents

Catalog: FileSystem

Label: File Binary Contents

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function read_file. 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 |
|---|---|---|---|---|
| ignore_errors | varchar2 | ☐ | | Whether to ignore errors w hen accessing the file. Defaults to false. |
| path | varchar2 | ☑ | | Relative or absolute path to the file. |

## Columns of Table Function

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| file_contents | blob | File Contents | ☐ | |

*(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.*

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| file_path | varchar2 | Filename and Path | ☑ | |
| is_existing | varchar2 | Is Existing | ☑ | |

#### 3.1.11 read_file_text: OS File Text Contents

Catalog: FileSystem

Label: File Text Contents

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function read_file_text. 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 |
|---|---|---|---|---|
| encoding | varchar2 | ☐ | | Character encoding, defaults to UTF8, alternative names listed in code page table on https://docs.microsoft.com/en-us/dotnet/api/system.text.encoding?view=netframework-4.8. |
| ignore_errors | varchar2 | ☐ | | Whether to ignore errors when accessing the file. Defaults to false. |
| path | varchar2 | ☑ | | Relative or absolute path to the file. |
| record_separator | varchar2 | ☐ | | Record separator is a text signaling the start of a new record. Defaults to platform-specific new line. |
| separate_on_record | varchar2 | ☐ | | Separates file contents into records based upon the record separator when true. Returns full file contents as one large string when false. Defaults to false. |

## Columns of Table Function

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| file_contents | clob | File Contents | ☐ | |
| file_path | varchar2 | Filename and Path | ☑ | |
| is_existing | varchar2 | Is Existing | ☑ | |

### 3.1.12 regexp_split_row: OS Split Row on Regular Expression

Catalog: FileSystem

Label: Split Row on Regular Expression

Retrieve: true

## Parameters of Table Function

The following parameters can be used to control the behaviour of the table function regexp_split_row. 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 |
|---|---|---|---|---|
| regex | varchar | ☑ | | Regular expression to use. |
| row | varchar | ☑ | | Text to split into individual columns using a regular expression. |

## Columns of Table Function

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

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| input_text | varchar2 | Original Input | | Original input. |
| regular_expression | varchar2 | Regular Expression | | Regular expression used to split. |
| success | boolean | Success | | Could content be split. |
| text_content_1 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_10 | varchar2 | RE Text Content #{0} | | Split content of the text file. |

*(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.*

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| text_content_11 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_12 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_13 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_14 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_15 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_16 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_17 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_18 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_19 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_2 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_20 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_21 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_22 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_23 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_24 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_25 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_26 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_27 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_28 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_29 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_3 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_30 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_31 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_32 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_33 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_34 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_35 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_36 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_37 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_38 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_39 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_4 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_40 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_41 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_42 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_43 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_44 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_45 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_46 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_47 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_48 | varchar2 | RE Text Content #{0} | | Split content of the text file. |

*(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.*

| Name | Data Type | Label | Required | Documentation |
|---|---|---|---|---|
| text_content_49 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_5 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_50 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_6 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_7 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_8 | varchar2 | RE Text Content #{0} | | Split content of the text file. |
| text_content_9 | varchar2 | RE Text Content #{0} | | Split content of the text file. |

#### 3.1.13 write_file: OS Write File

Catalog: FileSystem

Label: Write File

Retrieve: false

## Table Columns

The columns of the table write_file 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 |
|---|---|---|---|---|
| creation_date_time_utc | datetime | Requested Creation Date/Time (UTC) | ☐ | |
| file_contents | blob | | ☐ | |
| file_path | varchar2 | Filename and Path | ☑ | |
| modified_date_time_utc | datetime | Requested Modified Date/Time (UTC) | ☐ | |

# 4 Package: dcr_metadata

## 4.1 Procedures

#### 4.1.1 dcr_metadata.get_partitions: OS Data container metadata package

Get all partitions.

Documentation:

List all partitions.

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

# Index

## - A -

Action 7
all_directories 6, 11
analysis-enforce-row-uniqueness 1

## - B -

bulk-delete-page-size-rows 1
bulk-insert-page-size-bytes 1
bulk-insert-page-size-rows 1

## - C -

Create directory 8
create_directory 8
created 9
Created (UTC) 9
created_utc 9
creation_date_time_utc 16
csv 4
CSV Text Content #{0} 4
csv_split_row 4

## - D -

Database Driver 1
directories 6
Directory Name 9
Directory Path 6
directory_name 9
directory_path 6
Duration (ms) 7
duration_ms 7

## - E -

encoding 13
Exception Code 7, 9
Exception Message 7, 9
exception_code 7, 9
exception_message 7, 9
Extension 9

## - F -

File Actions 7
File Binary Contents 12
File Contents 12, 13
File Copy Actions 8
File Delete Actions 8
File Metadata Properties 9
File Move Actions 10
File Name and Path for Source 7, 8, 10
File Name and Path for Target 7, 8, 10
File Rename Actions 10
File Size (bytes) 9
File System Directories 6
File System Files 11
File Text Contents 13
file_actions 7
file_contents 12, 13, 16
file_copy_actions 8
file_delete_actions 8
file_info 9
file_move_actions 10
file_path 8, 11, 12, 13, 16
file_path_source 7, 8, 10
file_path_target 7, 8, 10
file_rename_actions 10
Filename and Path 8, 11, 12, 13, 16
files 11
force-case-sensitive-identifiers 1
forced-casing-identifiers 1

## - H -

Has Exception 7
has_exception 7

## - I -

Ignore Error 8
ignore_case 11
ignore_error 8
ignore_errors 9, 12, 13
ignore_unauthorized_exception 11
input_text 14
invantive-sql-compress-sparse-arrays 1
invantive-sql-correct-invalid-date 1
invantive-sql-execution-profile-disk-path 1
invantive-sql-execution-profile-to-disk 1
invantive-sql-forward-filters-to-data-containers 1

*(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.*

invantive-sql-share-byte-arrays 1
invantive-sql-share-strings 1
invantive-sql-shuffle-fetch-results-data-containers 1
invantive-use-cache 1
Is Archive File 9
Is Compressed File 9
Is Content Indexed File 9
Is Device File 9
Is Directory 9
Is Encrypted File 9
Is Existing 9, 12, 13
Is Hidden File 9
Is Integrity Stream File 9
Is Normal File 9
Is Offline File 9
Is Reparse Point File 9
Is Scrub Data File 9
Is Sparse File 9
Is System File 9
Is Temporary File 9
Is Writable File 9
is_archive 9
is_compressed 9
is_content_indexed 9
is_device 9
is_directory 9
is_encrypted 9
is_existing 9, 12, 13
is_hidden 9
is_integrity_stream 9
is_normal 9
is_offline 9
is_reparse_point 9
is_scrub_data 9
is_sparse 9
is_system 9
is_temporary 9
is_writable 9

# - L -

Last Access 9
Last Access (UTC) 9
Last Write (UTC) 9
last_access 9
last_access_utc 9
last_write 9
last_write_utc 9
length 9
log-native-calls-to-disk-max-events 1
log-native-calls-to-disk-max-seconds 1
log-native-calls-to-disk-on-error 1
log-native-calls-to-disk-on-success 1
log-native-calls-to-trace 1

# - M -

max_entries_per_row 4
maximum-length-identifiers 1
modified_date_time_utc 16

# - N -

Name 9

# - O -

Original Input 14
OS 1, 4, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16
Overwrite Existing 8
overwrite_existing 8

# - P -

partition-slot-based-rate-limit-length-ms 1
partition-slot-based-rate-limit-slots 1
path 6, 9, 11, 12, 13
pre-request-delay-ms 1

# - R -

RE Text Content #{0} 14
read_file 12
read_file_text 13
record_separator 13
regex 14
regexp_split_row 14
Regular Expression 14
regular_expression 14
Requested Creation Date/Time (UTC) 16
Requested Modified Date/Time (UTC) 16
requested-maximum-usable-partitions 1
requested-page-size 1
requests-parallel-max 1
row 14

# - S -

search_pattern 6, 11
separate_on_record 13

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.

slot-based-rate-limit-length-ms 1
slot-based-rate-limit-slots 1
Split Row as CSV 4
Split Row on Regular Expression 14
standardize-identifiers 1
standardize-identifiers-casing 1
Start Time 7
start_time 7
Success 14

# - T -

text_content_1 4, 14
text_content_10 4, 14
text_content_11 4, 14
text_content_12 4, 14
text_content_13 4, 14
text_content_14 4, 14
text_content_15 4, 14
text_content_16 4, 14
text_content_17 4, 14
text_content_18 4, 14
text_content_19 4, 14
text_content_2 4, 14
text_content_20 4, 14
text_content_21 4, 14
text_content_22 4, 14
text_content_23 4, 14
text_content_24 4, 14
text_content_25 4, 14
text_content_26 4, 14
text_content_27 4, 14
text_content_28 4, 14
text_content_29 4, 14
text_content_3 4, 14
text_content_30 4, 14
text_content_31 4, 14
text_content_32 4, 14
text_content_33 4, 14
text_content_34 4, 14
text_content_35 4, 14
text_content_36 4, 14
text_content_37 4, 14
text_content_38 4, 14
text_content_39 4, 14
text_content_4 4, 14
text_content_40 4, 14
text_content_41 4, 14
text_content_42 4, 14
text_content_43 4, 14
text_content_44 4, 14
text_content_45 4, 14
text_content_46 4, 14
text_content_47 4, 14
text_content_48 4, 14
text_content_49 4, 14
text_content_5 4, 14
text_content_50 4, 14
text_content_6 4, 14
text_content_7 4, 14
text_content_8 4, 14
text_content_9 4, 14

# - W -

Write File 16
write_file 16

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.




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

(C) Copyright 2004-2025 Invantive Software B.V., the Netherlands. All rights reserved.
24.0.92
29/06/2026, 19:44
ISBN 978-94-6302-000-8: