CREATE [ OR REPLACE ] FILE FORMAT [ IF NOT EXISTS ] <name>

TYPE = { CSV | JSON | AVRO | ORC | PARQUET | XML } [ formatTypeOptions ]

[ COMMENT = '<string\_literal>' ]

formatTypeOptions ::=

-- If TYPE = CSV

COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW\_DEFLATE | NONE

RECORD\_DELIMITER = '<character>' | NONE

FIELD\_DELIMITER = '<character>' | NONE

FILE\_EXTENSION = '<string>'

SKIP\_HEADER = <integer>

SKIP\_BLANK\_LINES = TRUE | FALSE

DATE\_FORMAT = '<string>' | AUTO

TIME\_FORMAT = '<string>' | AUTO

TIMESTAMP\_FORMAT = '<string>' | AUTO

BINARY\_FORMAT = HEX | BASE64 | UTF8

ESCAPE = '<character>' | NONE

ESCAPE\_UNENCLOSED\_FIELD = '<character>' | NONE

TRIM\_SPACE = TRUE | FALSE

FIELD\_OPTIONALLY\_ENCLOSED\_BY = '<character>' | NONE

NULL\_IF = ( '<string>' [ , '<string>' ... ] )

ERROR\_ON\_COLUMN\_COUNT\_MISMATCH = TRUE | FALSE

REPLACE\_INVALID\_CHARACTERS = TRUE | FALSE

VALIDATE\_UTF8 = TRUE | FALSE

EMPTY\_FIELD\_AS\_NULL = TRUE | FALSE

SKIP\_BYTE\_ORDER\_MARK = TRUE | FALSE

ENCODING = '<string>' | UTF8

TYPE = CSV

**COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW\_DEFLATE | NONE**

**Use**

Data loading and unloading

**Definition**

* When loading data, specifies the current compression algorithm for the data file. Snowflake uses this option to detect how an ***already-compressed*** data file was compressed so that the compressed data in the file can be extracted for loading.
* When unloading data, compresses the data file using the specified compression algorithm.

**Values**

| **Supported Values** | **Notes** |
| --- | --- |
| AUTO | When loading data, compression algorithm detected automatically, except for Brotli-compressed files, which cannot currently be detected automatically. When unloading data, files are automatically compressed using the default, which is gzip. |
| GZIP |  |
| BZ2 |  |
| BROTLI | Must be specified when loading/unloading Brotli-compressed files. |
| ZSTD | Zstandard v0.8 (and higher) is supported. |
| DEFLATE | Deflate-compressed files (with zlib header, RFC1950). |
| RAW\_DEFLATE | Raw Deflate-compressed files (without header, RFC1951). |
| NONE | When loading data, indicates that the files have not been compressed. When unloading data, specifies that the unloaded files are not compressed. |

**Default**

AUTO

**RECORD\_DELIMITER = '*character*' | NONE**

**Use**

Data loading and unloading

**Default**

**Data loading**

New line character. Note that “new line” is logical such that \r\n will be understood as a new line for files on a Windows platform.

**Data unloading**

New line character (\n).

**FIELD\_DELIMITER = '*character*' | NONE**

**Use**

Data loading and unloading

The specified delimiter must be a valid UTF-8 character and not a random sequence of bytes. Also note that the delimiter is limited to a maximum of 20 characters.

Also accepts a value of NONE.

**Default**

comma (,)

**FILE\_EXTENSION = '*string*' | NONE**

**Use**

Data unloading only

**Definition**

Specifies the extension for files unloaded to a stage. Accepts any extension. The user is responsible for specifying a file extension that can be read by any desired software or services.

**Default**

null, meaning the file extension is determined by the format type: .csv[*compression*], where *compression* is the extension added by the compression method, if COMPRESSION is set.

**Note**

If the SINGLE copy option is TRUE, then the COPY command unloads a file without a file extension by default. To specify a file extension, provide a file name and extension in the *internal\_location* or *external\_location* path (e.g. copy into @stage/data.csv).

**SKIP\_HEADER = *integer***

**Use**

Data loading only

**Definition**

Number of lines at the start of the file to skip.

Note that SKIP\_HEADER does not use the RECORD\_DELIMITER or FIELD\_DELIMITER values to determine what a header line is; rather, it simply skips the specified number of CRLF (Carriage Return, Line Feed)-delimited lines in the file. RECORD\_DELIMITER and FIELD\_DELIMITER are then used to determine the rows of data to load.

**Default**

0

**SKIP\_BLANK\_LINES = TRUE | FALSE**

**Use**

Data loading only

**Definition**

Boolean that specifies to skip any blank lines encountered in the data files; otherwise, blank lines produce an end-of-record error (default behavior).

Default: FALSE

**DATE\_FORMAT = '*string*' | AUTO**

**Use**

Data loading and unloading

**Definition**

Defines the format of date values in the data files (data loading) or table (data unloading). If a value is not specified or is AUTO, the value for the [DATE\_INPUT\_FORMAT](https://docs.snowflake.com/en/sql-reference/parameters.html#label-date-input-format) (data loading) or [DATE\_OUTPUT\_FORMAT](https://docs.snowflake.com/en/sql-reference/parameters.html#label-date-output-format) (data unloading) parameter is used.

**Default**

AUTO

**TIME\_FORMAT = '*string*' | AUTO**

**Use**

Data loading and unloading

**Definition**

Defines the format of time values in the data files (data loading) or table (data unloading). If a value is not specified or is AUTO, the value for the [TIME\_INPUT\_FORMAT](https://docs.snowflake.com/en/sql-reference/parameters.html#label-time-input-format) (data loading) or [TIME\_OUTPUT\_FORMAT](https://docs.snowflake.com/en/sql-reference/parameters.html#label-time-output-format) (data unloading) parameter is used.

**Default**

AUTO

**TIMESTAMP\_FORMAT = *string*' | AUTO**

**Use**

Data loading and unloading

**Definition**

Defines the format of timestamp values in the data files (data loading) or table (data unloading). If a value is not specified or is AUTO, the value for the [TIMESTAMP\_INPUT\_FORMAT](https://docs.snowflake.com/en/sql-reference/parameters.html#label-timestamp-input-format) (data loading) or [TIMESTAMP\_OUTPUT\_FORMAT](https://docs.snowflake.com/en/sql-reference/parameters.html#label-timestamp-output-format) (data unloading) parameter is used.

**Default**

AUTO

**BINARY\_FORMAT = HEX | BASE64 | UTF8**

**Use**

Data loading and unloading

**Definition**

Defines the encoding format for binary input or output. The option can be used when loading data into or unloading data from binary columns in a table.

**Default**

HEX

**ESCAPE = '*character*' | NONE**

**Use**

Data loading and unloading

**Definition**

Single character string used as the escape character for enclosed or unenclosed field values. An escape character invokes an alternative interpretation on subsequent characters in a character sequence. You can use the ESCAPE character to interpret instances of the FIELD\_OPTIONALLY\_ENCLOSED\_BY character in the data as literals.

Accepts common escape sequences, octal values, or hex values.

* When loading data, specifies the escape character for enclosed fields. Specify the character used to enclose fields by setting FIELD\_OPTIONALLY\_ENCLOSED\_BY.
* When unloading data, if this option is set, it overrides the escape character set for ESCAPE\_UNENCLOSED\_FIELD.

**Default**

NONE

Consider an example where ESCAPE\_UNENCLOSED\_FIELD is defined with backslash "\" and the file delimiter is comma ",". Imagine if the comma character (",") is also present inside the data fields. Now, any commas preceded with the backslash "\" will be considered as a data character instead of a delimiter. Note that the backslash '\' will be consumed as an escape sequence and will not be in the loaded data.

ESCAPE\_UNENCLOSED\_FIELD **only** **works with delimiters**. In the following example when we try to escape '\' it makes no difference but when it comes with any of the delimiter(record or column) then it works fine.

|  |  |  |
| --- | --- | --- |
| **Data in File** | **After Data Loading** | **Comments** |
| data\so | data\so | It will not escape '\' as it is not with a comma (,) |
| dt\,ss | dt\ss | escaping '\' because it is placed with a delimiter (comma) |

**ESCAPE\_UNENCLOSED\_FIELD = '*character*' | NONE**

**Use**

Data loading and unloading

**Definition**

Single character string used as the escape character for unenclosed field values only. An escape character invokes an alternative interpretation on subsequent characters in a character sequence. You can use the ESCAPE character to interpret instances of the FIELD\_DELIMITER or RECORD\_DELIMITER characters in the data as literals. The escape character can also be used to escape instances of itself in the data.

Accepts common escape sequences, octal values, or hex values.

Note that when unloading data, if ESCAPE is set, the escape character set for that file format option overrides this option.

**Default**

backslash (\\)

|  |  |
| --- | --- |
| **Simple Escape Sequences** |  |
| \' | A single quote (') character |
| \" | A double quote (") character |
| \\ | A backslash (\) character |
| \b | A backspace character |
| \f | A formfeed character |
| \n | A newline (linefeed) character |
| \r | A carriage return character |
| \t | A tab character |

In **File-format** options, ESCAPE can only work with the FIELD\_OPTIONALLY\_ENCLOSED\_BY's character. It is used to distinguish between FIELD\_OPTIONALLY\_ENCLOSED\_BY's and the same delimiter characters as part of actual data. Please find below a reusable example to understand this concept thoroughly.

At the time of loading, it is only removing “ \ “ when it is enclosed in the apostrophe.

|  |  |  |
| --- | --- | --- |
| **Data in File** | **After Data Loading** | **Comments** |
| ‘sasa\’sas’ | sasa’sas | As Expected |
| abc\’sas | abc\’sas | It did not remove "\" Because it was not enclosed in ' |

**TRIM\_SPACE = TRUE | FALSE**

**Use**

Data loading only

**Definition**

Boolean that specifies whether to remove white space from fields.

For example, if your external database software encloses fields in quotes, but inserts a leading space, Snowflake reads the leading space rather than the opening quotation character as the beginning of the field (i.e. the quotation marks are interpreted as part of the string of field data). Set this option to TRUE to remove undesirable spaces during the data load.

As another example, if leading or trailing spaces surround quotes that enclose strings, you can remove the surrounding spaces using this option and the quote character using the FIELD\_OPTIONALLY\_ENCLOSED\_BY option. Note that any spaces ***within*** the quotes are preserved. For example, assuming FIELD\_DELIMITER = '|' and FIELD\_OPTIONALLY\_ENCLOSED\_BY = '"':

|" Hello world "| /\* loads as \*/ >hello world<

|" Hello world "| /\* loads as \*/ > hello world <

| "Hello world" | /\* loads as \*/ >hello world<

(the brackets in this example are not loaded; they are used to demarcate the beginning and end of the loaded strings)

**Default**

FALSE

**FIELD\_OPTIONALLY\_ENCLOSED\_BY = '*character*' | NONE**

**Use**

Data loading and unloading

**Definition**

Character used to enclose strings. Value can be NONE, single quote character ('), or double quote character ("). To use the single quote character, use the octal or hex representation (0x27) or the double single-quoted escape ('').

When a field contains this character, escape it using the same character. For example, if the value is the double quote character and a field contains the string A "B" C, escape the double quotes as follows:

A ""B"" C

**Default**

NONE

**NULL\_IF = ( '*string1*' [ , '*string2*' , ... ] )**

**Use**

Data loading and unloading

**Definition**

String used to convert to and from SQL NULL:

* When loading data, Snowflake replaces these values in the data load source with SQL NULL. To specify more than one string, enclose the list of strings in parentheses and use commas to separate each value.

Note that Snowflake converts all instances of the value to NULL, regardless of the data type. For example, if 2 is specified as a value, all instances of 2 as either a string or number are converted.

For example:

NULL\_IF = ('\\N', 'NULL', 'NUL', '')

Note that this option can include empty strings.

* When unloading data, Snowflake converts SQL NULL values to the first value in the list.

**Default**

\\N (i.e. NULL, which assumes the ESCAPE\_UNENCLOSED\_FIELD value is \\)

**ERROR\_ON\_COLUMN\_COUNT\_MISMATCH = TRUE | FALSE**

**Use**

Data loading only

**Definition**

Boolean that specifies whether to generate a parsing error if the number of delimited columns (i.e. fields) in an input file does not match the number of columns in the corresponding table.

If set to FALSE, an error is not generated and the load continues. If the file is successfully loaded:

* If the input file contains records with more fields than columns in the table, the matching fields are loaded in order of occurrence in the file and the remaining fields are not loaded.
* If the input file contains records with fewer fields than columns in the table, the non-matching columns in the table are loaded with NULL values.

This option assumes all the records within the input file are the same length (i.e. a file containing records of varying length return an error regardless of the value specified for this parameter).

**Default**

TRUE

**Note**

When [transforming data during loading](https://docs.snowflake.com/en/user-guide/data-load-transform.html) (i.e. using a query as the source for the COPY command), this option is ignored. There is no requirement for your data files to have the same number and ordering of columns as your target table.

**REPLACE\_INVALID\_CHARACTERS = TRUE | FALSE**

**Use**

Data loading only

**Definition**

Boolean that specifies whether to replace invalid UTF-8 characters with the Unicode replacement character.

If set to TRUE, Snowflake replaces invalid UTF-8 characters with the Unicode replacement character.

If set to FALSE, the load operation produces an error when invalid UTF-8 character encoding is detected.

**Default**

FALSE

**VALIDATE\_UTF8 = TRUE | FALSE**

**Use**

Data loading only

**Definition**

Boolean that specifies whether to validate UTF-8 character encoding in string column data.

If set to TRUE, Snowflake validates UTF-8 character encoding in string column data. When invalid UTF-8 character encoding is detected, the COPY command produces an error.

**Default**

TRUE

**Important**

This option is provided only to ensure backward compatibility with earlier versions of Snowflake. You should ***not*** disable this option unless instructed by Snowflake Support.

**EMPTY\_FIELD\_AS\_NULL = TRUE | FALSE**

**Use**

Data loading and unloading

**Definition**

* When loading data, specifies whether to insert SQL NULL for empty fields in an input file, which are represented by two successive delimiters (e.g. ,,).

If set to FALSE, Snowflake attempts to cast an empty field to the corresponding column type. An empty string is inserted into columns of type STRING. For other column types, the COPY command produces an error.

* When unloading data, this option is used in combination with FIELD\_OPTIONALLY\_ENCLOSED\_BY. When FIELD\_OPTIONALLY\_ENCLOSED\_BY = NONE, setting EMPTY\_FIELD\_AS\_NULL = FALSE specifies to unload empty strings in tables to empty string values without quotes enclosing the field values.

If set to TRUE, FIELD\_OPTIONALLY\_ENCLOSED\_BY must specify a character to enclose strings.

**Default**

TRUE

**ENCODING = '*string*'**

**Use**

Data loading only

**Definition**

String (constant) that specifies the character set of the source data when loading data into a table.

| **Character Set** | | **ENCODING Value** | **Supported Languages** | | | **Notes** | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Big5 | | BIG5 | Traditional Chinese | | |  | |
| EUC-JP | | EUCJP | Japanese | | |  | |
| EUC-KR | | EUCKR | Korean | | |  | |
| GB18030 | | GB18030 | Chinese | | |  | |
| IBM420 | | IBM420 | Arabic | | |  | |
| IBM424 | | IBM424 | Hebrew | | |  | |
| ISO-2022-CN | | ISO2022CN | Simplified Chinese | | |  | |
| ISO-2022-JP | | ISO2022JP | Japanese | | |  | |
| ISO-2022-KR | | ISO2022KR | Korean | | |  | |
| ISO-8859-1 | | ISO88591 | Danish, Dutch, English, French, German, Italian, Norwegian, Portuguese, Swedish | | |  | |
| ISO-8859-2 | | ISO88592 | Czech, Hungarian, Polish, Romanian | | |  | |
| ISO-8859-5 | | ISO88595 | Russian | | |  | |
| ISO-8859-6 | | ISO88596 | Arabic | | |  | |
| ISO-8859-7 | | ISO88597 | Greek | | |  | |
| ISO-8859-8 | | ISO88598 | Hebrew | | |  | |
| ISO-8859-9 | | ISO88599 | Turkish | | |  | |
| KOI8-R | | KOI8R | Russian | | |  | |
| Shift\_JIS | | SHIFTJIS | Japanese | | |  | |
| UTF-8 | | UTF8 | All languages | | |  | |
| For loading data from all other supported file formats (JSON, Avro, etc.), as well as unloading data, UTF-8 is the only supported character set. | | For loading data from delimited files (CSV, TSV, etc.), UTF-8 is the default. | |
| UTF-16 | | UTF16 | All languages | | |  |
| UTF-16BE | | UTF16BE | All languages | | |  |
| UTF-16LE | | UTF16LE | All languages | | |  |
| UTF-32 | | UTF32 | All languages | | |  |
| UTF-32BE | | UTF32BE | All languages | | |  |
| UTF-32LE | | UTF32LE | All languages | | |  |
| windows-1250 | | WINDOWS1250 | Czech, Hungarian, Polish, Romanian | | |  |
| windows-1251 | | WINDOWS1251 | Russian | | |  |
| windows-1252 | | WINDOWS1252 | Danish, Dutch, English, French, German, Italian, Norwegian, Portuguese, Swedish | | |  |
| windows-1253 | | WINDOWS1253 | Greek | | |  |
| windows-1254 | | WINDOWS1254 | Turkish | | |  |
| windows-1255 | | WINDOWS1255 | Hebrew | | |  |
| windows-1256 | | WINDOWS1256 | Arabic | | |  |

**Default**

UTF8

**Note**

Snowflake stores all data internally in the UTF-8 character set. The data is converted into UTF-8 before it is loaded into Snowflake.