CREATE FILE FORMAT

Creates a named file format that describes a set of staged data to access or load into Snowflake tables.

See also:

ALTER FILE FORMAT , DROP FILE FORMAT , SHOW FILE FORMATS , DESCRIBE FILE FORMAT

COPY INTO <location> , COPY INTO <table>

Syntax

```
CREATE [ OR REPLACE ] [ { TEMP | TEMPORARY | VOLATILE } ] FILE FORMAT [ IF
NOT EXISTS ] <name>
  [ TYPE = { CSV | JSON | AVRO | ORC | PARQUET | XML | CUSTOM} [
formatTypeOptions ] ]
  [ COMMENT = '<string_literal>' ]
```

Where:

```
formatTypeOptions ::=
-- If TYPE = CSV
     COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE |
RAW_DEFLATE | NONE
     RECORD_DELIMITER = '<string>' | NONE
     FIELD_DELIMITER = '<string>' | NONE
     FILE_EXTENSION = '<string>'
     PARSE_HEADER = TRUE | FALSE
     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
     EMPTY_FIELD_AS_NULL = TRUE | FALSE
     SKIP_BYTE_ORDER_MARK = TRUE | FALSE
     ENCODING = '<string>' | UTF8
-- If TYPE = JSON
     COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE |
RAW_DEFLATE | NONE
     DATE_FORMAT = '<string>' | AUTO
     TIME_FORMAT = '<string>' | AUTO
     TIMESTAMP_FORMAT = '<string>' | AUTO
     BINARY_FORMAT = HEX | BASE64 | UTF8
     TRIM_SPACE = TRUE | FALSE
     NULL_IF = ( '<string>' [ , '<string>' ... ] )
     FILE_EXTENSION = '<string>'
     ENABLE_OCTAL = TRUE | FALSE
     ALLOW_DUPLICATE = TRUE | FALSE
     STRIP_OUTER_ARRAY = TRUE | FALSE
     STRIP_NULL_VALUES = TRUE | FALSE
     REPLACE_INVALID_CHARACTERS = TRUE | FALSE
     IGNORE_UTF8_ERRORS = TRUE | FALSE
     SKIP_BYTE_ORDER_MARK = TRUE | FALSE
-- If TYPE = AVRO
     COMPRESSION = AUTO | GZIP | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE |
NONE
     TRIM_SPACE = TRUE | FALSE
     REPLACE_INVALID_CHARACTERS = TRUE | FALSE
     NULL_IF = ( '<string>' [ , '<string>' ... ] )
-- If TYPE = ORC
     TRIM_SPACE = TRUE | FALSE
     REPLACE_INVALID_CHARACTERS = TRUE | FALSE
     NULL_IF = ( '<string>' [ , '<string>' ... ] )
-- If TYPE = PARQUET
     COMPRESSION = AUTO | LZO | SNAPPY | NONE
     SNAPPY_COMPRESSION = TRUE | FALSE
     BINARY_AS_TEXT = TRUE | FALSE
     USE_LOGICAL_TYPE = TRUE | FALSE
     TRIM_SPACE = TRUE | FALSE
     REPLACE_INVALID_CHARACTERS = TRUE | FALSE
     NULL_IF = ( '<string>' [ , '<string>' ... ] )
-- If TYPE = XML
     COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE |
RAW_DEFLATE | NONE
     IGNORE_UTF8_ERRORS = TRUE | FALSE
     PRESERVE_SPACE = TRUE | FALSE
     STRIP_OUTER_ELEMENT = TRUE | FALSE
     DISABLE_SNOWFLAKE_DATA = TRUE | FALSE
     DISABLE_AUTO_CONVERT = TRUE | FALSE
     REPLACE_INVALID_CHARACTERS = TRUE | FALSE
     SKIP_BYTE_ORDER_MARK = TRUE | FALSE
```

Required parameters

`<name>`

Specifies the identifier for the file format; must be unique for the schema in which the file format is created.

The identifier value must start with an alphabetic character and cannot contain spaces or special characters unless the entire identifier string is enclosed in double quotes (e.g. `"My object"`), Identifiers enclosed in double quotes are also case-sensitive.

For more details, see Identifier requirements.

Optional parameters

`{ TEMP | TEMPORARY | VOLATILE }`

Specifies that the file format persists only for the duration of the session that you created it in. A temporary file format is dropped at the end of the session.

Default: No value. If a file format is not declared as `TEMPORARY`, the file format is permanent.

If you want to avoid unexpected conflicts, avoid naming temporary file formats after file formats that already exist in the schema.

If you created a temporary file format with the same name as another file format in the schema, all queries and operations used on the file format only affect the temporary file format in the session, until you drop the temporary file format. If you drop the file format using a DROP FILE FORMAT command, you drop the temporary file format, and not the file format that already exists in the schema.

`TYPE = CSV | JSON | AVRO | ORC | PARQUET | XML [ ... ]`

Specifies the format of the input files (for data loading) or output files (for data unloading). Depending on the format type, you can specify additional format-specific options. For more information, see Format Type Options (in this topic).

Valid values depend on whether the file format is for loading or unloading data:

`CSV` (for loading or unloading)

Any flat, delimited plain text file that uses specific characters such as the following:

- Separators for fields within records (for example, commas).
- Separators for records (for example, new line characters).

Although the name (CSV) suggests comma-separated values, you can use any valid character as a field separator.

`JSON` **(for loading or unloading)**

Any plain text file containing one or more JSON documents (such as objects or arrays). JSON is a semi-structured file format. The documents can be comma-separated and optionally enclosed in a big array. A single JSON document can span multiple lines.

> **Note**
>
> - When you load data from files into tables, Snowflake supports either NDJSON (newline delimited JSON) standard format or comma-separated JSON format.
> - When you unload table data to files, Snowflake outputs *only* to NDJSON format.

`AVRO` **(for loading only; you can't unload data to AVRO format)**

Binary file in AVRO format.

`ORC` **(for loading only; you can't unload data to ORC format)**

Binary file in ORC format.

`PARQUET` **(for loading or unloading)**

Binary file in PARQUET format.

`XML` **(for loading only; you can't unload data to XML format)**

Plain text file containing XML elements.

`CUSTOM` **(for loading unstructured data only)**

> PREVIEW FEATURE — OPEN
>
> Available to all accounts.

This format type specifies that the underlying stage holds unstructured data and can only be used with the `FILE_PROCESSOR` copy option.

For more information about CSV, see Usage Notes in this topic. For more information about JSON and the other semi-structured file formats, see Introduction to Loading Semi-structured Data. For more information about `CUSTOM` type, see Loading unstructured data with Document AI.

Default: `CSV`

`COMMENT = '<string_literal>'`

Specifies a comment for the file format.

Default: No value

Format type options (`formatTypeOptions`)

Depending on the file format type specified (`TYPE = ...`), you can include one or more of the following format-specific options (separated by blank spaces, commas, or new lines):

TYPE = CSV

`COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE | NONE`

Use Data loading, data unloading, and external tables

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.

Supported Values	Notes
`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 = '<string>' | NONE`

Use Data loading, data unloading, and external tables

Definition One or more singlebyte or multibyte characters that separate records in an input file (data loading) or unloaded file (data unloading). Accepts common escape sequences or the following singlebyte or multibyte characters:

Singlebyte characters	Octal values (prefixed by `\\`) or hex values (prefixed by `0x` or `\x`). For example, for records delimited by the circumflex accent (`^`) character, specify the octal (`\\136`) or hex (`0x5e`) value.
Multibyte characters	Hex values (prefixed by `\x`). For example, for records delimited by the cent (`¢`) character, specify the hex (`\xC2\xA2`) value. The delimiter for RECORD_DELIMITER or FIELD_DELIMITER cannot be a substring of the delimiter for the other file format option (e.g. `FIELD_DELIMITER = 'aa' RECORD_DELIMITER = 'aabb'`).

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

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 = '<string>' | NONE`

Use
Data loading, data unloading, and external tables

Definition
One or more singlebyte or multibyte characters that separate fields in an input file (data loading) or unloaded file (data unloading). Accepts common escape sequences or the following singlebyte or multibyte characters:

Singlebyte characters	Octal values (prefixed by `\\`) or hex values (prefixed by `0x` or `\x`). For example, for records delimited by the circumflex accent (`^`) character, specify the octal (`\\136`) or hex (`0x5e`) value.
Multibyte characters	Hex values (prefixed by `\x`). For example, for records delimited by the cent (`¢`) character, specify the hex (`\xC2\xA2`) value.

The delimiter for RECORD_DELIMITER or FIELD_DELIMITER cannot be a substring of the delimiter for the other file format option (e.g. `FIELD_DELIMITER = 'aa' RECORD_DELIMITER = 'aabb'`).

> **Note**
> For non-ASCII characters, you must use the hex byte sequence value to get a deterministic behavior.

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`).

`PARSE_HEADER = TRUE | FALSE`

Use Data loading only

Definition Boolean that specifies whether to use the first row headers in the data files to determine column names.

This file format option is applied to the following actions only:

- Automatically detecting column definitions by using the INFER_SCHEMA function.
- Loading CSV data into separate columns by using the INFER_SCHEMA function and MATCH_BY_COLUMN_NAME copy option.

If the option is set to TRUE, the first row headers will be used to determine column names. The default value FALSE will return column names as c*, where * is the position of the column.

> **Note**
> - This option isn't supported for external tables.

- The SKIP_HEADER option isn’t supported if you set `PARSE_HEADER = TRUE`.

Default: `FALSE`

`SKIP_HEADER = <integer>`

Use Data loading and external tables

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 and external tables

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 (data loading) or 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 (data loading) or 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 (data loading) or 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	A singlebyte 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.
Loading data	Specifies the escape character for enclosed fields only. Specify the character used to enclose fields by setting `FIELD_OPTIONALLY_ENCLOSED_BY`.

Note

This file format option supports singlebyte characters only. Note that UTF-8 character encoding represents high-order ASCII characters as multibyte characters. If your data file is encoded with the UTF-8 character set, you cannot specify a high-order ASCII character as the option value.

In addition, if you specify a high-order ASCII character, we recommend that you set the `ENCODING = '<string>'` file format option as the character encoding for your data files to ensure the character is interpreted correctly.

Unloading data If this option is set, it overrides the escape character set for `ESCAPE_UNENCLOSED_FIELD`.

Default `NONE`

`ESCAPE_UNENCLOSED_FIELD = '<character>' | NONE`

Use Data loading, data unloading, and external tables

Definition A singlebyte 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.

Loading data Specifies the escape character for unenclosed fields only.

Note

- The default value is `\\`. If a row in a data file ends in the backslash (`\`) character, this character escapes the newline or carriage return character specified for the `RECORD_DELIMITER` file format option. As a result, the load operation treats this row and the next row as a single row of data. To avoid this issue, set the value to `NONE`.
- This file format option supports singlebyte characters only. Note that UTF-8 character encoding represents high-order ASCII characters as multibyte characters. If your data file is encoded with the UTF-8 character set, you cannot specify a high-order ASCII character as the option value.

In addition, if you specify a high-order ASCII character, we recommend that you set the `ENCODING = '<string>'` file format option as the character encoding for your data files to ensure the character is interpreted correctly.

Unloading data If `ESCAPE` is set, the escape character set for that file format option overrides this option.

Default backslash (`\\`)

`TRIM_SPACE = TRUE | FALSE`

Use Data loading and external tables

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, data unloading, and external tables

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 (`''`).

Data unloading only	When a field in the source table contains this character, Snowflake escapes it using the same character for unloading. For example, if the value is the double quote character and a field contains the string `A "B" C`, Snowflake escapes the double quotes for unloading as follows: `A ""B"" C`

Default `NONE`

`NULL_IF = ( '<string1>' [ , '<string2>' , ... ] )`

Use Data loading, data unloading, and external tables

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 (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`

`EMPTY_FIELD_AS_NULL = TRUE | FALSE`

Use Data loading, data unloading, and external tables

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`

`SKIP_BYTE_ORDER_MARK = TRUE | FALSE`

Use Data loading only

Definition Boolean that specifies whether to skip the BOM (byte order mark), if present in a data file. A BOM is a character code at the beginning of a data file that defines the byte order and encoding form.

If set to `FALSE`, Snowflake recognizes any BOM in data files, which could result in the BOM either causing an error or being merged into the first column in the table.

Default `TRUE`

`ENCODING = '<string>'`

Use Data loading and external tables

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	

Character Set	`ENCODING` Value	Supported Languages	Notes
GB18030	`GB18030`	Chinese	
IBM420	`IBM420`	Arabic	
IBM424	`IBM424`	Hebrew	
IBM949	`IBM949`	Korean	
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	

Character Set	`ENCODING` Value	Supported Languages	Notes
ISO-8859-15	`ISO885915`	Danish, Dutch, English, French, German, Italian, Norwegian, Portuguese, Swedish	Identical to ISO-8859-1 except for 8 characters, including the Euro currency symbol.
KOI8-R	`KOI8R`	Russian	
Shift_JIS	`SHIFTJIS`	Japanese	
UTF-8	`UTF8`	All languages	For loading data from delimited files (CSV, TSV, etc.), UTF-8 is the default. 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.
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-874	`WINDOWS874`	Thai	
windows-949	`WINDOWS949`	Korean	

Character Set	`ENCODING` Value	Supported Languages	Notes
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.

TYPE = JSON

`COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE | NONE`

Use Data loading and external tables

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 if 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`

`DATE_FORMAT = '<string>' | AUTO`

Use Data loading only

Definition Defines the format of date string values in the data files. If a value is not specified or is `AUTO`, the value for the DATE_INPUT_FORMAT parameter is used.

This file format option is applied to the following actions only:

- Loading JSON data into separate columns using the MATCH_BY_COLUMN_NAME copy option.

- Loading JSON data into separate columns by specifying a query in the COPY statement (i.e. COPY transformation).

Default `AUTO`

`TIME_FORMAT = '<string>' | AUTO`

Use Data loading only

Definition Defines the format of time string values in the data files. If a value is not specified or is `AUTO`, the value for the TIME_INPUT_FORMAT parameter is used.

This file format option is applied to the following actions only:

- Loading JSON data into separate columns using the MATCH_BY_COLUMN_NAME copy option.
- Loading JSON data into separate columns by specifying a query in the COPY statement (i.e. COPY transformation).

Default `AUTO`

`TIMESTAMP_FORMAT = <string>' | AUTO`

Use Data loading only

Definition Defines the format of timestamp string values in the data files. If a value is not specified or is `AUTO`, the value for the TIMESTAMP_INPUT_FORMAT parameter is used.

This file format option is applied to the following actions only:

- Loading JSON data into separate columns using the MATCH_BY_COLUMN_NAME copy option.
- Loading JSON data into separate columns by specifying a query in the COPY statement (i.e. COPY transformation).

Default `AUTO`

`BINARY_FORMAT = HEX | BASE64 | UTF8`

Use Data loading only

Definition Defines the encoding format for binary string values in the data files. The option can be used when loading data into binary columns in a table.

This file format option is applied to the following actions only:

- Loading JSON data into separate columns using the MATCH_BY_COLUMN_NAME copy option.
- Loading JSON data into separate columns by specifying a query in the COPY statement (i.e. COPY transformation).

Default `HEX`

`TRIM_SPACE = TRUE | FALSE`

Use Data loading only

Definition Boolean that specifies whether to remove leading and trailing white space from strings.

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.

This file format option is applied to the following actions only when loading JSON data into separate columns using the MATCH_BY_COLUMN_NAME copy option.

Default `FALSE`

`NULL_IF = ( '<string1>' [ , '<string2>' , ... ] )`

Use Data loading only

Definition String used to convert to and from SQL NULL. Snowflake replaces these strings 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.

This file format option is applied to the following actions only when loading JSON data into separate columns using the MATCH_BY_COLUMN_NAME copy option.

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.

Default `\\N` (i.e. NULL, which assumes the `ESCAPE_UNENCLOSED_FIELD` value is `\\`)

`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: `.json[<compression>]`, where `<compression>` is the extension added by the compression method, if `COMPRESSION` is set.

`ENABLE_OCTAL = TRUE | FALSE`

Use Data loading only

Definition Boolean that enables parsing of octal numbers.

Default `FALSE`

`ALLOW_DUPLICATE = TRUE | FALSE`

Use Data loading and external tables

Definition Boolean that specifies to allow duplicate object field names (only the last one will be preserved).

Default `FALSE`

`STRIP_OUTER_ARRAY = TRUE | FALSE`

Use Data loading and external tables

Definition Boolean that instructs the JSON parser to remove outer brackets (i.e. `[ ]`).

Default `FALSE`

`STRIP_NULL_VALUES = TRUE | FALSE`

Use Data loading and external tables

Definition Boolean that instructs the JSON parser to remove object fields or array elements containing `null` values. For example, when set to `TRUE`:

Before	After
`[null]`	`[]`
`[null,null,3]`	`[,,3]`
`{"a":null,"b":null,"c":123}`	`{"c":123}`
`{"a":[1,null,2],"b":{"x":null,"y":88}}`	`{"a":[1,,2],"b":{"y":88}}`

Default `FALSE`

`REPLACE_INVALID_CHARACTERS = TRUE | FALSE`

Use Data loading and external table

Definition Boolean that specifies whether to replace invalid UTF-8 characters with the Unicode replacement character (`�`). This option performs a one-to-one character replacement.

Values 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`

`IGNORE_UTF8_ERRORS = TRUE | FALSE`

Use	Data loading and external table
Definition	Boolean that specifies whether UTF-8 encoding errors produce error conditions. It is an alternative syntax for `REPLACE_INVALID_CHARACTERS`.
Values	If set to `TRUE`, any invalid UTF-8 sequences are silently replaced with the Unicode character `U+FFFD` (i.e. "replacement character"). If set to `FALSE`, the load operation produces an error when invalid UTF-8 character encoding is detected.
Default	`FALSE`

`SKIP_BYTE_ORDER_MARK = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether to skip the BOM (byte order mark), if present in a data file. A BOM is a character code at the beginning of a data file that defines the byte order and encoding form. If set to `FALSE`, Snowflake recognizes any BOM in data files, which could result in the BOM either causing an error or being merged into the first column in the table.
Default	`TRUE`

TYPE = AVRO

`COMPRESSION = AUTO | GZIP | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE | NONE`

Use Data loading only

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`	
`BROTLI`	Must be specified if 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`.

Note
We recommend that you use the default `AUTO` option because it will determine both the file and codec compression. Specifying a compression option refers to the compression of files, not the compression of blocks (codecs).

`TRIM_SPACE = TRUE | FALSE`

Use Data loading only

Definition Boolean that specifies whether to remove leading and trailing white space from strings.

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.

This file format option is applied to the following actions only when loading Avro data into separate columns using the MATCH_BY_COLUMN_NAME copy option.

Default	`FALSE`

`REPLACE_INVALID_CHARACTERS = TRUE | FALSE`

Use	Data loading and external table
Definition	Boolean that specifies whether to replace invalid UTF-8 characters with the Unicode replacement character (`�`). This option performs a one-to-one character replacement.
Values	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`

`NULL_IF = ( '<string1>' [ , '<string2>' , ... ] )`

Use	Data loading only
Definition	String used to convert to and from SQL NULL. Snowflake replaces these strings 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. This file format option is applied to the following actions only when loading Avro data into separate columns using the MATCH_BY_COLUMN_NAME copy option. 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.
Default	`\\N` (i.e. NULL, which assumes the `ESCAPE_UNENCLOSED_FIELD` value is `\\`)

TYPE = ORC

`TRIM_SPACE = TRUE | FALSE`

Use	Data loading and external tables
Definition	Boolean that specifies whether to remove leading and trailing white space from strings. 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. This file format option is applied to the following actions only when loading Orc data into separate columns using the MATCH_BY_COLUMN_NAME copy option.
Default	`FALSE`

`REPLACE_INVALID_CHARACTERS = TRUE | FALSE`

Use	Data loading and external table
Definition	Boolean that specifies whether to replace invalid UTF-8 characters with the Unicode replacement character (`�`). This option performs a one-to-one character replacement.
Values	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`

`NULL_IF = ( '<string1>' [ , '<string2>' , ... ] )`

Use	Data loading and external tables
Definition	String used to convert to and from SQL NULL. Snowflake replaces these strings 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.

This file format option is applied to the following actions only when loading Orc data into separate columns using the MATCH_BY_COLUMN_NAME copy option.

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.

Default `\\N` (i.e. NULL, which assumes the `ESCAPE_UNENCLOSED_FIELD` value is `\\`)

TYPE = PARQUET

`COMPRESSION = AUTO | LZO | SNAPPY | NONE`

Use Data loading, data unloading, and external tables

Definition

- When loading data, specifies the current compression algorithm for columns in the Parquet files.
- When unloading data, compresses the data file using the specified compression algorithm.

Values

Supported Values	Notes
`AUTO`	When loading data, compression algorithm detected automatically. Supports the following compression algorithms: Brotli, gzip, Lempel-Ziv-Oberhumer (LZO), LZ4, Snappy, or Zstandard v0.8 (and higher). When unloading data, unloaded files are compressed using the Snappy compression algorithm by default.
`LZO`	When unloading data, files are compressed using the Snappy algorithm by default. If unloading data to LZO-compressed files, specify this value.
`SNAPPY`	When unloading data, files are compressed using the Snappy algorithm by default. You can optionally specify this value.
`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`

`SNAPPY_COMPRESSION = TRUE | FALSE`

Use Data unloading only

Supported Values	Notes
`AUTO`	Unloaded files are compressed using the Snappy compression algorithm by default.
`SNAPPY`	May be specified if unloading Snappy-compressed files.
`NONE`	When loading data, indicates that the files have not been compressed. When unloading data, specifies that the unloaded files are not compressed.

Definition Boolean that specifies whether unloaded file(s) are compressed using the SNAPPY algorithm.

Note
Deprecated. Use `COMPRESSION = SNAPPY` instead.

Limitations Only supported for data unloading operations.

Default `TRUE`

`BINARY_AS_TEXT = TRUE | FALSE`

Use Data loading and external tables

Definition Boolean that specifies whether to interpret columns with no defined logical data type as UTF-8 text. When set to `FALSE`, Snowflake interprets these columns as binary data.

Default `TRUE`

Note

Snowflake recommends that you set BINARY_AS_TEXT to FALSE to avoid any potential conversion issues.

`TRIM_SPACE = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether to remove leading and trailing white space from strings. 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. This file format option is applied to the following actions only when loading Parquet data into separate columns using the MATCH_BY_COLUMN_NAME copy option.
Default	`FALSE`

`USE_LOGICAL_TYPE = TRUE | FALSE`

Use	Data loading, data querying in staged files, and schema detection.
Definition	Boolean that specifies whether to use Parquet logical types. With this file format option, Snowflake can interpret Parquet logical types during data loading. For more information, see Parquet Logical Type Definitions. To enable Parquet logical types, set USE_LOGICAL_TYPE as TRUE when you create a new file format option.
Limitations	Not supported for data unloading.

`USE_VECTORIZED_SCANNER = TRUE | FALSE`

Use	Data loading and data querying in staged files
Definition	Boolean that specifies whether to use a vectorized scanner for loading Parquet files.
Default	`FALSE`. In a future BCR, the default value will be `TRUE`.

Using the vectorized scanner can significantly reduce the latency for loading Parquet files, because this scanner is well suited for the columnar format of a Parquet file. The scanner only downloads relevant sections of the Parquet file into memory, such as the subset of selected columns.

You can only enable the vectorized scanner if the following conditions are met:

- The `ON_ERROR` option must be set to `ABORT_STATEMENT` or `SKIP_FILE`.

  The other values, `CONTINUE`, `SKIP_FILE_<num>`, `'SKIP_FILE_<num>%'` are not supported.

If `USE_VECTORIZED_SCANNER` is set to `TRUE`, the vectorized scanner has the following behaviors:

- The `BINARY_AS_TEXT` option is always treated as `FALSE` and the `USE_LOGICAL_TYPE` option is always treated as `TRUE`, no matter what the actual value is being set to.
- The vectorized scanner supports Parquet map types. The output of scanning a map type is as follows:

  ```
  "my_map":
    {
     "k1": "v1",
     "k2": "v2"
    }
  ```

- The vectorized scanner shows `NULL` values in the output, as the following example demonstrates:

  ```
  "person":
   {
    "name": "Adam",
    "nickname": null,
    "age": 34,
    "phone_numbers":
    [
      "1234567890",
      "0987654321",
      null,
      "6781234590"
    ]
    }
  ```

- The vectorized scanner handles Time and Timestamp as follows:

Parquet	Snowflake vectorized scanner
TimeType(isAdjustedToUtc=True/False, unit=MILLIS/MICROS/NANOS)	TIME
TimestampType(isAdjustedToUtc=True, unit=MILLIS/MICROS/NANOS)	TIMESTAMP_LTZ
TimestampType(isAdjustedToUtc=False, unit=MILLIS/MICROS/NANOS)	TIMESTAMP_NTZ
INT96	TIMESTAMP_LTZ

If `USE_VECTORIZED_SCANNER` is set to `FALSE`, the scanner has the following behaviors:

- This option does not support Parquet maps. The output of scanning a map type is as follows:

```
"my_map":
 {
  "key_value":
  [
   {
          "key": "k1",
          "value": "v1"
      },
      {
          "key": "k2",
          "value": "v2"
      }
    ]
  }
```

- This option does not explicitly show `NULL` values in the scan output, as the following example demonstrates:

```
"person":
 {
  "name": "Adam",
  "age": 34
  "phone_numbers":
  [
   "1234567890",
   "0987654321",
   "6781234590"
  ]
 }
```

- This option handles Time and Timestamp as follows:

Parquet	When USE_LOGICAL_TYPE = TRUE	When USE_LOGICAL_TYPE = FALSE
TimeType(isAdjustedToUtc=True/False, unit=MILLIS/MICROS)	TIME	• TIME (If ConvertedType present) • INTEGER (If ConvertedType not present)
TimeType(isAdjustedToUtc=True/False, unit=NANOS)	TIME	INTEGER
TimestampType(isAdjustedToUtc=True, unit=MILLIS/MICROS)	TIMESTAMP_LTZ	TIMESTAMP_NTZ
TimestampType(isAdjustedToUtc=True, unit=NANOS)	TIMESTAMP_LTZ	INTEGER
TimestampType(isAdjustedToUtc=False, unit=MILLIS/MICROS)	TIMESTAMP_NTZ	• TIMESTAMP_LTZ (If ConvertedType present) • INTEGER (If ConvertedType not present)
TimestampType(isAdjustedToUtc=False, unit=NANOS)	TIMESTAMP_NTZ	INTEGER
INT96	TIMESTAMP_NTZ	TIMESTAMP_NTZ

`REPLACE_INVALID_CHARACTERS = TRUE | FALSE`

Use	Data loading and external table
Definition	Boolean that specifies whether to replace invalid UTF-8 characters with the Unicode replacement character (`�`). This option performs a one-to-one character replacement.
Values	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`

`NULL_IF = ( '<string1>' [ , '<string2>' , ... ] )`

Use Data loading only

Definition String used to convert to and from SQL NULL. Snowflake replaces these strings 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.

This file format option is applied to the following actions only when loading Parquet data into separate columns using the MATCH_BY_COLUMN_NAME copy option.

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.

Default `\\N` (i.e. NULL, which assumes the `ESCAPE_UNENCLOSED_FIELD` value is `\\`)

TYPE = XML

PREVIEW FEATURE — OPEN

Available to all accounts.

`COMPRESSION = AUTO | GZIP | BZ2 | BROTLI | ZSTD | DEFLATE | RAW_DEFLATE | NONE`

Use Data loading only

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 if 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`

`IGNORE_UTF8_ERRORS = TRUE | FALSE`

Use Data loading and external table

Definition Boolean that specifies whether UTF-8 encoding errors produce error conditions. It is an alternative syntax for `REPLACE_INVALID_CHARACTERS`.

Values If set to `TRUE`, any invalid UTF-8 sequences are silently replaced with the Unicode character `U+FFFD` (i.e. “replacement character”).

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

Default `FALSE`

`PRESERVE_SPACE = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether the XML parser preserves leading and trailing spaces in element content.
Default	`FALSE`

`STRIP_OUTER_ELEMENT = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether the XML parser strips out the outer XML element, exposing 2nd level elements as separate documents.
Default	`FALSE`

`DISABLE_SNOWFLAKE_DATA = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether the XML parser disables recognition of Snowflake semi-structured data tags.
Default	`FALSE`

`DISABLE_AUTO_CONVERT = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether the XML parser disables automatic conversion of numeric and Boolean values from text to native representation.
Default	`FALSE`

`REPLACE_INVALID_CHARACTERS = TRUE | FALSE`

Use	Data loading and external table
Definition	Boolean that specifies whether to replace invalid UTF-8 characters with the Unicode replacement character (`�`). This option performs a one-to-one character replacement.

Values	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`

`SKIP_BYTE_ORDER_MARK = TRUE | FALSE`

Use	Data loading only
Definition	Boolean that specifies whether to skip any BOM (byte order mark) present in an input file. A BOM is a character code at the beginning of a data file that defines the byte order and encoding form. If set to `FALSE`, Snowflake recognizes any BOM in data files, which could result in the BOM either causing an error or being merged into the first column in the table.
Default	`TRUE`

Access control requirements

A role used to execute this SQL command must have the following privileges at a minimum:

Privilege	Object	Notes
CREATE FILE FORMAT	Schema	
OWNERSHIP	File format	A role must be granted or inherit the OWNERSHIP privilege on the object to create a temporary object that has the same name as the object that already exists in the schema. Note that in a managed access schema, only the schema owner (i.e. the role with the OWNERSHIP privilege on the schema) or a role with the MANAGE GRANTS privilege can grant or revoke privileges on objects in the schema, including future grants.

Note that operating on any object in a schema also requires the USAGE privilege on the parent database and schema.

For instructions on creating a custom role with a specified set of privileges, see Creating custom roles.

For general information about roles and privilege grants for performing SQL actions on securable objects, see Overview of Access Control.

Usage notes

Caution
Recreating a file format (using CREATE OR REPLACE FILE FORMAT) breaks the association between the file format and any external table that references it. This is because an external table links to a file format using a hidden ID rather than the name of the file format. Behind the scenes, the CREATE OR REPLACE syntax drops an object and recreates it with a different hidden ID.

If you must recreate a file format after it has been linked to one or more external tables, you must recreate each of the external tables (using CREATE OR REPLACE EXTERNAL TABLE) to reestablish the association. Call the GET_DDL function to retrieve a DDL statement to recreate each of the external tables.

- Conflicting file format values in a SQL statement produce an error. A conflict occurs when the same option is specified multiple times with different values (e.g. `...TYPE = 'CSV' ... TYPE = 'JSON'...`).
- Regarding metadata:

  Attention
  Customers should ensure that no personal data (other than for a User object), sensitive data, export-controlled data, or other regulated data is entered as metadata when using the Snowflake service. For more information, see Metadata fields in Snowflake.

- CREATE OR REPLACE *<object>* statements are atomic. That is, when an object is replaced, the old object is deleted and the new object is created in a single transaction.

Examples

Create a CSV file format named `my_csv_format` that defines the following rules for data files:

- Fields are delimited using the pipe character (`|`).
- Files include a single header line that will be skipped.

- The strings `NULL` and `null` will be replaced with NULL values.
- Empty strings will be interpreted as NULL values.
- Files will be compressed/decompressed using GZIP compression.

```
CREATE OR REPLACE FILE FORMAT my_csv_format
  TYPE = CSV
  FIELD_DELIMITER = '|'
  SKIP_HEADER = 1
  NULL_IF = ('NULL', 'null')
  EMPTY_FIELD_AS_NULL = true
  COMPRESSION = gzip;
```

Create a JSON file format named `my_json_format` that uses all the default JSON format options:

```
CREATE OR REPLACE FILE FORMAT my_json_format
  TYPE = JSON;
```

Create a PARQUET file format named `my_parquet_format` that does not compress unloaded data files using the Snappy algorithm:

```
CREATE OR REPLACE FILE FORMAT my_parquet_format
  TYPE = PARQUET
  COMPRESSION = SNAPPY;
```

Create a PARQUET file format named `my_parquet_format` that uses PARQUET logical types, instead of physical types or the legacy converted types.

```
CREATE OR REPLACE FILE FORMAT my_parquet_format
  TYPE = PARQUET
  USE_LOGICAL_TYPE = TRUE;
```