



Last updated on **Dec 30, 2025**

# Auto Loader options

Configuration options specific to the `cloudFiles` source are prefixed with `cloudFiles` so that they are in a separate namespace from other Structured Streaming source options.

- [Common Auto Loader options](#)
- [Directory listing options](#)
- [File notification options](#)
- [File format options](#)
  - [Generic options](#)
  - [JSON options](#)
  - [CSV options](#)
  - [XML options](#)
  - [PARQUET options](#)
  - [AVRO options](#)
  - [BINARYFILE options](#)
  - [TEXT options](#)
  - [ORC options](#)
- [Cloud-specific options](#)
  - [AWS-specific options](#)
  - [Azure-specific options](#)
  - [Google-specific options](#)

## Common Auto Loader options

You can configure the following options for Auto Loader streams.

## Options

`cloudFiles.allowOverwrites`

Type: `Boolean`

Whether to allow input directory file changes to overwrite existing data.

For configuration caveats, see [Does Auto Loader process the file again when the file gets appended or overwritten?](#).

Default: `false`

`cloudFiles.backfillInterval`

Type: `Interval String`

Auto Loader can trigger asynchronous backfills at a given interval. For example `1 day` to backfill daily or `1 week` to backfill weekly. For more information, see [Trigger regular backfills using `cloudFiles.backfillInterval`](#).

Do not use when `cloudFiles.useManagedFileEvents` is set to `true`.

Default: None

`cloudFiles.cleanSource`

Type: `String`

Whether to automatically delete processed files from the input directory. When set to `OFF` (default), no files are deleted.

When set to `DELETE`, Auto Loader automatically deletes files 30 days after they are processed. To do this, Auto Loader must have write permissions to the source directory.

When set to `MOVE`, Auto Loader automatically moves files to the specified location in `cloudFiles.cleanSource.moveDestination` 30 days after they are processed. To do this, Auto



## Options

Loader must have write permissions to the source directory as well as to the move location.

A file is considered processed when it has a non-null value for `commit_time` in the result of the `cloud_files_state` table valued function. See [cloud\\_files\\_state table-valued function](#). The 30 day additional wait after processing can be configured using `cloudFiles.cleanSource.retentionDuration`.

**Note:** Databricks does not recommend using this option if there are multiple streams consuming data from the source location because the fastest consumer will delete the files and they will not be ingested in the slower sources.

**Note:** Enabling this feature requires Auto Loader to maintain additional state in its checkpoint, which incurs performance overhead but enables improved observability through the `cloud_files_state` table valued function. See [cloud\\_files\\_state table-valued function](#).

**Note:** `cleanSource` uses the current setting to decide whether to `MOVE` or `DELETE` a given file. For example, suppose that the setting was `MOVE` when the file was originally processed but was changed to `DELETE` when the file became a candidate for cleanup 30 days later. In this case, `cleanSource` will delete the file.

**Note:** `cleanSource` is best-effort. Files are *not* guaranteed to be deleted as soon as the `retentionDuration` period is over. Instead, they become candidates for cleanup and are deleted on a best-effort basis during regular stream processing. To save costs, `cleanSource` gracefully exits after stream processing is complete, even if there are candidates remaining for cleanup. The remaining candidates will be picked up during the next processing.

Available in Databricks Runtime 16.4 and above.

Default: OFF

`cloudFiles.cleanSource.retentionDuration`

Type: `Interval String`

 Ask Assistant

## Options

Amount of time to wait before processed files become candidates for archival with `cleanSource`. Must be greater than 7 days for `DELETE`. No minimum restriction for `MOVE`.

Available in Databricks Runtime 16.4 and above.

Default value: 30 days

`cloudFiles.cleanSource.moveDestination`

Type: `String`

Path to archive processed files to when `cloudFiles.cleanSource` is set to `MOVE`.

The move location is restricted in the following ways:

- Should not be a child of the source directory (this will cause the archived files to be ingested again)
- **S3**: Should be a directory in the same S3 bucket. Cross-bucket moves require files to be downloaded and re-uploaded to the new bucket, which can be expensive.
- **GCS**: Should be a directory in the same GCS bucket. Cross-bucket moves require files to be downloaded and re-uploaded to the new bucket, which can be expensive.
- **Azure**: Should be a directory in the same Azure container. Cross-container moves require files to be downloaded and re-uploaded to the new container, which can be expensive.

Auto Loader must have write permissions to this directory.

Available in Databricks Runtime 16.4 and above.

Default value: None

`cloudFiles.format`

Type: `String`

 Ask Assistant

## Options

The [data file format](#) in the source path. Allowed values include:

- `avro`: [Avro files](#)
- `binaryFile`: [Binary files](#)
- `csv`: [CSV files](#)
- `json`: [JSON files](#)
- `orc`: [ORC files](#)
- `parquet`: [Parquet files](#)
- `text`: [TXT files](#)
- `xml`: [XML files](#)

Default: None (required option)

`cloudFiles.includeExistingFiles`

Type: `Boolean`

Whether to include existing files in the stream processing input path or to only process new files arriving after initial setup. This option is evaluated only when you start a stream for the first time. Changing this option after restarting the stream has no effect.

Default: `true`

`cloudFiles.inferColumnTypes`

Type: `Boolean`

Whether to infer exact column types when leveraging schema inference. By default, columns are inferred as strings when inferring JSON and CSV datasets. See [schema inference](#) for more details.

Default: `false`

## Options

`cloudFiles.maxBytesPerTrigger`

Type: `Byte String`

The maximum number of new bytes to be processed in every trigger. You can specify a byte string such as `10g` to limit each microbatch to 10 GB of data. This is a soft maximum. If you have files that are 3 GB each, Databricks processes 12 GB in a microbatch. When used together with `cloudFiles.maxFilesPerTrigger`, Databricks consumes up to the lower limit of `cloudFiles.maxFilesPerTrigger` or `cloudFiles.maxBytesPerTrigger`, whichever is reached first. This option has no effect when used with `Trigger.Once()` (`Trigger.Once()` is deprecated).

Default: None

`cloudFiles.maxFileAge`

Type: `Interval String`

How long a file event is tracked for deduplication purposes. Databricks does not recommend tuning this parameter unless you are ingesting data at the order of millions of files an hour. See the section on [File event tracking](#) for more details.

Tuning `cloudFiles.maxFileAge` too aggressively can cause data quality issues such as duplicate ingestion or missing files. Therefore, Databricks recommends a conservative setting for `cloudFiles.maxFileAge`, such as 90 days, which is similar to what comparable data ingestion solutions recommend.

Default: None

`cloudFiles.maxFilesPerTrigger`

Type: `Integer`

The maximum number of new files to be processed in every trigger. When used together with `cloudFiles.maxBytesPerTrigger`, Databricks consumes up to the lower limit of Ask Assistant

## Options

`cloudFiles.maxFilesPerTrigger` or `cloudFiles.maxBytesPerTrigger`, whichever is reached first. This option has no effect when used with `Trigger.Once()` (deprecated).

Default: 1000

`cloudFiles.partitionColumns`

Type: `String`

A comma separated list of Hive style partition columns that you would like inferred from the directory structure of the files. Hive style partition columns are key value pairs combined by an equality sign such as `<base-path>/a=x/b=1/c=y/file.format`. In this example, the partition columns are `a`, `b`, and `c`. By default these columns are automatically added to your schema if you are using schema inference and provide the `<base-path>` to load data from. If you provide a schema, Auto Loader expects these columns to be included in the schema. If you do not want these columns as part of your schema, you can specify `""` to ignore these columns. In addition, you can use this option when you want columns to be inferred the file path in complex directory structures, like the example below:

```
<base-path>/year=2022/week=1/file1.csv <base-path>/year=2022/month=2/day=3/file2.csv  
<base-path>/year=2022/month=2/day=4/file3.csv
```

Specifying `cloudFiles.partitionColumns` as `year,month,day` returns `year=2022` for `file1.csv`, but the `month` and `day` columns are `null`.

`month` and `day` are parsed correctly for `file2.csv` and `file3.csv`.

Default: None

`cloudFiles.schemaEvolutionMode`

Type: `String`

The mode for evolving the schema as new columns are discovered in the data. By default columns are inferred as strings when inferring JSON datasets. See [schema evolution](#) for

## Options

more details.

Default: `addNewColumns` when a schema is not provided, `none` otherwise

`cloudFiles.schemaHints`

Type: `String`

Schema information that you provide to Auto Loader during schema inference. See [schema hints](#) for more details.

Default: None

`cloudFiles.schemaLocation`

Type: `String`

The location to store the inferred schema and subsequent changes. See [schema inference](#) for more details.

Default: None (required to infer the schema)

`cloudFiles.useStrictGlobber`

Type: `Boolean`

Whether to use a strict globber that matches the default globbing behavior of other file sources in Apache Spark. See [Common data loading patterns](#) for more details. Available in Databricks Runtime 12.2 LTS and above.

Default: `false`

`cloudFiles.validateOptions`

 Ask Assistant

## Options

Type: `Boolean`

Whether to validate Auto Loader options and return an error for unknown or inconsistent options.

Default: `true`

# Directory listing options

The following options are relevant to directory listing mode.

## Options

`cloudFiles.useIncrementalListing` (deprecated)

Type: `String`

This feature has been deprecated. Databricks recommends using [file notification mode with file events](#) instead of `cloudFiles.useIncrementalListing`.

Whether to use the incremental listing rather than the full listing in directory listing mode. By default, Auto Loader makes the best effort to automatically detect if a given directory is applicable for the incremental listing. You can explicitly use the incremental listing or use the full directory listing by setting it as `true` or `false` respectively.

Incorrectly enabling incremental listing on a non-lexically ordered directory prevents Auto Loader from discovering new files.

Works with Azure Data Lake Storage (`abfss://`), S3 (`s3://`), and GCS (`gs://`).

Available in [Databricks Runtime 9.1 LTS](#) and above.

## Options

Default: `auto` on Databricks Runtime 17.2 and below, `false` on Databricks Runtime 17.3 and above

Available values: `auto`, `true`, `false`

# File notification options

The following options are relevant to file notification mode.

## Options

`cloudFiles.fetchParallelism`

Type: `Integer`

Number of threads to use when fetching messages from the queueing service.

Do not use when `cloudFiles.useManagedFileEvents` is set to `true`.

Default: 1

`cloudFiles.pathRewrites`

Type: A JSON string

Required only if you specify a `queueUrl` that receives file notifications from multiple S3 buckets and you want to leverage mount points configured for accessing data in these containers. Use this option to rewrite the prefix of the `bucket/key` path with the mount point. Only prefixes can be rewritten. For example, for the configuration `{"<databricks-mounted-bucket>/path": "dbfs:/mnt/data-warehouse"}`, the path `s3://<databricks-mounted-bucket>/path/2017/08/fileA.json` is rewritten to `dbfs:/mnt/data-warehouse/2017/08/fileA.json`



## Options

Do not use when `cloudFiles.useManagedFileEvents` is set to `true`.

Default: None

`cloudFiles.resourceTag`

Type: `Map(String, String)`

A series of key-value tag pairs to help associate and identify related resources, for example:

```
cloudFiles.option("cloudFiles.resourceTag.myFirstKey", "myFirstValue")
.option("cloudFiles.resourceTag.mySecondKey", "mySecondValue")
```

For more information on AWS, see [Amazon SQS cost allocation tags](#) and [Configuring tags for an Amazon SNS topic](#). (1)

For more information on Azure, see [Naming Queues and Metadata](#) and the coverage of `properties.labels` in [Event Subscriptions](#). Auto Loader stores these key-value tag pairs in JSON as labels. (1)

For more information on GCP, see [Reporting usage with labels](#). (1)

Do not use when `cloudFiles.useManagedFileEvents` is set to `true`. Instead set resource tags using the cloud provider console.

Default: None

`cloudFiles.useManagedFileEvents`

When set to `true`, Auto Loader uses the file events service to discover files in your external location. You can use this option only if the load path is in an external location with file events enabled. See [Use file notification mode with file events](#).

File events provide notifications-level performance in file discovery, because Auto Loader can discover new files since the last run. Unlike directory listing, this process  [does not need](#)

## Options

to list all files in the directory.

There are some situations when Auto Loader uses directory listing even though the file events option is enabled:

- During initial load, when `includeExistingFiles` is set to `true`, a full directory listing takes place to discover all of the files that were present in the directory before Auto Loader started.
- The file events service optimizes file discovery by caching the most recently created files. If Auto Loader runs infrequently, this cache can expire, and Auto Loader falls back to directory listing to discover files and update the cache. To avoid this scenario, invoke Auto Loader at least once every seven days.

See [When does Auto Loader with file events use directory listing?](#) for a comprehensive list of situations when Auto Loader uses directory listing with this option.

Available in Databricks Runtime 14.3 LTS and above.

`cloudFiles.useNotifications`

Type: `Boolean`

Whether to use file notification mode to determine when there are new files. If `false`, use directory listing mode. See [Compare Auto Loader file detection modes](#).

Do not use when `cloudFiles.useManagedFileEvents` is set to `true`.

Default: `false`

**(1)** Auto Loader adds the following key-value tag pairs by default on a best-effort basis:

- `vendor: Databricks`
- `path`: The location from where the data is loaded. Unavailable in GCP due to labeling limitations.

- `checkpointLocation`: The location of the stream's checkpoint. Unavailable in GCP due to labeling limitations.
- `streamId`: A globally unique identifier for the stream.

These key names are reserved and you cannot overwrite their values.

## File format options

With Auto Loader you can ingest `JSON`, `CSV`, `PARQUET`, `AVRO`, `TEXT`, `BINARYFILE`, and `ORC` files.

- [Generic options](#)
- [JSON options](#)
- [CSV options](#)
- [XML options](#)
- [PARQUET options](#)
- [AVRO options](#)
- [BINARYFILE options](#)
- [TEXT options](#)
- [ORC options](#)

## Generic options

The following options apply to all file formats.

Option
<code>ignoreCorruptFiles</code> Type: <code>Boolean</code>  Whether to ignore corrupt files. If true, the Spark jobs will continue to run when encountering corrupted files and the contents that have been read will still be returned. Observable as <code>numSkippedCorruptFiles</code> in the <code>operationMetrics</code> column of the Delta Lake history. Available in Databricks Runtime 11.3 LTS and above.

## Option

Default value: `false`

`ignoreMissingFiles`

Type: `Boolean`

Whether to ignore missing files. If true, the Spark jobs will continue to run when encountering missing files and the contents that have been read will still be returned.  
Available in Databricks Runtime 11.3 LTS and above.

Default value: `false` for Auto Loader, `true` for `COPY INTO` (legacy)

`modifiedAfter`

Type: `Timestamp String`, for example, `2021-01-01 00:00:00.000000 UTC+0`

An optional timestamp as a filter to only ingest files that have a modification timestamp after the provided timestamp.

Default value: None

`modifiedBefore`

Type: `Timestamp String`, for example, `2021-01-01 00:00:00.000000 UTC+0`

An optional timestamp as a filter to only ingest files that have a modification timestamp before the provided timestamp.

Default value: None

`pathGlobFilter` or `fileNamePattern`

Type: `String`

 Ask Assistant

## Option

A potential glob pattern to provide for choosing files. Equivalent to `PATTERN` in `COPY INTO` (legacy). `fileNamePattern` can be used in `read_files`.

Default value: None

### `recursiveFileLookup`

Type: `Boolean`

This option searches through nested directories even if their names do not follow a partition naming scheme like `date=2019-07-01`.

Default value: `false`

# JSON options

## Option

### `allowBackslashEscapingAnyCharacter`

Type: `Boolean`

Whether to allow backslashes to escape any character that succeeds it. If not enabled, only characters that are explicitly listed by the JSON specification can be escaped.

Default value: `false`

### `allowComments`

Type: `Boolean`

## Option

Whether to allow the use of Java, C, and C++ style comments ('/'\*, '\*' and '//' varieties) within parsed content or not.

Default value: `false`

### `allowNonNumericNumbers`

Type: `Boolean`

Whether to allow the set of not-a-number (`NaN`) tokens as legal floating number values.

Default value: `true`

### `allowNumericLeadingZeros`

Type: `Boolean`

Whether to allow integral numbers to start with additional (ignorable) zeroes (for example, `0000001`).

Default value: `false`

### `allowSingleQuotes`

Type: `Boolean`

Whether to allow use of single quotes (apostrophe, character `'\'`) for quoting strings (names and String values).

Default value: `true`

### `allowUnquotedControlChars`

Type: `Boolean`

## Option

Whether to allow JSON strings to contain unescaped control characters (ASCII characters with value less than 32, including tab and line feed characters) or not.

Default value: `false`

### `allowUnquotedFieldNames`

Type: `Boolean`

Whether to allow use of unquoted field names (which are allowed by JavaScript, but not by the JSON specification).

Default value: `false`

### `badRecordsPath`

Type: `String`

The path to store files for recording the information about bad JSON records.

Using the `badRecordsPath` option in a file-based data source has the following limitations:

- It is non-transactional and can lead to inconsistent results.
- Transient errors are treated as failures.

Default value: None

### `columnNameOfCorruptRecord`

Type: `String`

The column for storing records that are malformed and cannot be parsed. If the `mode` for parsing is set as `DROPMALFORMED`, this column will be empty.

## Option

Default value: `_corrupt_record`

### `dateFormat`

Type: `String`

The format for parsing date strings.

Default value: `yyyy-MM-dd`

### `dropFieldIfAllNull`

Type: `Boolean`

Whether to ignore columns of all null values or empty arrays and structs during schema inference.

Default value: `false`

### `encoding` or `charset`

Type: `String`

The name of the encoding of the JSON files. See `java.nio.charset.Charset` for list of options.

You cannot use `UTF-16` and `UTF-32` when `multiline` is `true`.

Default value: `UTF-8`

### `inferTimestamp`

Type: `Boolean`

Whether to try and infer timestamp strings as a `TimestampType`. When set to `true`, schema inference might take noticeably longer. You must enable `cloudFiles.inferColumnTypes` to use

## Option

with Auto Loader.

Default value: `false`

### `lineSep`

Type: `String`

A string between two consecutive JSON records.

Default value: None, which covers `\r`, `\r\n`, and `\n`

### `locale`

Type: `String`

A `java.util.Locale` identifier. Influences default date, timestamp, and decimal parsing within the JSON.

Default value: `US`

### `mode`

Type: `String`

Parser mode around handling malformed records. One of `PERMISSIVE`, `DROPMALFORMED`, or `FAILFAST`.

Default value: `PERMISSIVE`

### `multiLine`

Type: `Boolean`

## Option

Whether the JSON records span multiple lines.

Default value: `false`

`prefersDecimal`

Type: `Boolean`

Attempts to infer strings as `DecimalType` instead of float or double type when possible. You must also use schema inference, either by enabling `inferSchema` or using `cloudFiles.inferColumnTypes` with Auto Loader.

Default value: `false`

`primitivesAsString`

Type: `Boolean`

Whether to infer primitive types like numbers and booleans as `StringType`.

Default value: `false`

`readerCaseSensitive`

Type: `Boolean`

Specifies the case sensitivity behavior when `rescuedDataColumn` is enabled. If true, rescue the data columns whose names differ by case from the schema; otherwise, read the data in a case-insensitive manner. Available in Databricks Runtime 13.3 and above.

Default value: `true`

`rescuedDataColumn`

 Ask Assistant

## Option

Type: `String`

Whether to collect all data that can't be parsed due to a data type mismatch or schema mismatch (including column casing) to a separate column. This column is included by default when using Auto Loader. For more details, refer to [What is the rescued data column?](#).

`COPY INTO` (legacy) does not support the rescued data column because you cannot manually set the schema using `COPY INTO`. Databricks recommends using Auto Loader for most ingestion scenarios.

Default value: None

### `singleVariantColumn`

Type: `String`

Whether to ingest the entire JSON document, parsed into a single Variant column with the given string as the column's name. If disabled, the JSON fields will be ingested into their own columns.

Default value: None

### `timestampFormat`

Type: `String`

The format for parsing timestamp strings.

Default value: `yyyy-MM-dd'T'HH:mm:ss[.sss][XXX]`

### `timeZone`

Type: `String`

## Option

The `java.time.ZoneId` to use when parsing timestamps and dates.

Default value: None

# CSV options

## Option

`badRecordsPath`

Type: `String`

The path to store files for recording the information about bad CSV records.

Default value: None

`charToEscapeQuoteEscaping`

Type: `Char`

The character used to escape the character used for escaping quotes. For example, for the following record: `[ " a\\\", b ]`:

- If the character to escape the `'\'` is undefined, the record won't be parsed. The parser will read characters: `[a],[\],["],[,],[ ] ,[b]` and throw an error because it cannot find a closing quote.
- If the character to escape the `'\'` is defined as `'\'`, the record will be read with 2 values: `[a\]` and `[b]`.

Default value: `'\0'`

`columnNameOfCorruptRecord`

## Option

Supported for Auto Loader. Not supported for `COPY INTO` (legacy).

Type: `String`

The column for storing records that are malformed and cannot be parsed. If the `mode` for parsing is set as `DROPMALFORMED`, this column will be empty.

Default value: `_corrupt_record`

### comment

Type: `Char`

Defines the character that represents a line comment when found in the beginning of a line of text. Use '`\0`' to disable comment skipping.

Default value: `'\u0000'`

### dateFormat

Type: `String`

The format for parsing date strings.

Default value: `yyyy-MM-dd`

### emptyValue

Type: `String`

String representation of an empty value.

Default value: `""`

## Option

`encoding` or `charset`

Type: `String`

The name of the encoding of the CSV files. See `java.nio.charset.Charset` for the list of options. `UTF-16` and `UTF-32` cannot be used when `multiline` is `true`.

Default value: `UTF-8`

`enforceSchema`

Type: `Boolean`

Whether to forcibly apply the specified or inferred schema to the CSV files. If the option is enabled, headers of CSV files are ignored. This option is ignored by default when using Auto Loader to rescue data and allow schema evolution.

Default value: `true`

`escape`

Type: `Char`

The escape character to use when parsing the data.

Default value: `'\'`

`header`

Type: `Boolean`

Whether the CSV files contain a header. Auto Loader assumes that files have headers when inferring the schema.

## Option

Default value: `false`

`ignoreLeadingWhiteSpace`

Type: `Boolean`

Whether to ignore leading whitespaces for each parsed value.

Default value: `false`

`ignoreTrailingWhiteSpace`

Type: `Boolean`

Whether to ignore trailing whitespaces for each parsed value.

Default value: `false`

`inferSchema`

Type: `Boolean`

Whether to infer the data types of the parsed CSV records or to assume all columns are of `StringType`. Requires an additional pass over the data if set to `true`. For Auto Loader, use `cloudFiles.inferColumnTypes` instead.

Default value: `false`

`lineSep`

Type: `String`

A string between two consecutive CSV records.

## Option

Default value: None, which covers `\r`, `\r\n`, and `\n`

### locale

Type: `String`

A `java.util.Locale` identifier. Influences default date, timestamp, and decimal parsing within the CSV.

Default value: `us`

### maxCharsPerColumn

Type: `Int`

Maximum number of characters expected from a value to parse. Can be used to avoid memory errors. Defaults to `-1`, which means unlimited.

Default value: `-1`

### maxColumns

Type: `Int`

The hard limit of how many columns a record can have.

Default value: `20480`

### mergeSchema

Type: `Boolean`

Whether to infer the schema across multiple files and to merge the schema of each file.  Ask Assistant  
Enabled by default for Auto Loader when inferring the schema.

## Option

Default value: `false`

`mode`

Type: `String`

Parser mode around handling malformed records. One of `'PERMISSIVE'`, `'DROPIMALFORMED'`, and `'FAILFAST'`.

Default value: `PERMISSIVE`

`multiLine`

Type: `Boolean`

Whether the CSV records span multiple lines.

Default value: `false`

`nanValue`

Type: `String`

The string representation of a non-a-number value when parsing `FloatType` and `DoubleType` columns.

Default value: `"NaN"`

`negativeInf`

Type: `String`

## Option

The string representation of negative infinity when parsing `FloatType` or `DoubleType` columns.

Default value: `"-Inf"`

### `nullValue`

Type: `String`

String representation of a null value.

Default value: `""`

### `parserCaseSensitive` (deprecated)

Type: `Boolean`

While reading files, whether to align columns declared in the header with the schema case sensitively. This is `true` by default for Auto Loader. Columns that differ by case will be rescued in the `rescuedDataColumn` if enabled. This option has been deprecated in favor of `readerCaseSensitive`.

Default value: `false`

### `positiveInf`

Type: `String`

The string representation of positive infinity when parsing `FloatType` or `DoubleType` columns.

Default value: `"Inf"`

### `preferDate`

 Ask Assistant

## Option

Type: Boolean

Attempts to infer strings as dates instead of timestamp when possible. You must also use schema inference, either by enabling `inferSchema` or using `cloudFiles.inferColumnTypes` with Auto Loader.

Default value: true

### quote

Type: Char

The character used for escaping values where the field delimiter is part of the value.

Default value: "

### readerCaseSensitive

Type: Boolean

Specifies the case sensitivity behavior when `rescuedDataColumn` is enabled. If true, rescue the data columns whose names differ by case from the schema; otherwise, read the data in a case-insensitive manner.

Default value: true

### rescuedDataColumn

Type: String

Whether to collect all data that can't be parsed due to: a data type mismatch, and schema mismatch (including column casing) to a separate column. This column is included by default when using Auto Loader. For more details refer to [What is the rescued data column?](#)

## Option

`COPY INTO` (legacy) does not support the rescued data column because you cannot manually set the schema using `COPY INTO`. Databricks recommends using Auto Loader for most ingestion scenarios.

Default value: None

`sep` or `delimiter`

Type: `String`

The separator string between columns.

Default value: `","`

`skipRows`

Type: `Int`

The number of rows from the beginning of the CSV file that should be ignored (including commented and empty rows). If `header` is true, the header will be the first unskipped and uncommented row.

Default value: `0`

`timestampFormat`

Type: `String`

The format for parsing timestamp strings.

Default value: `yyyy-MM-dd'T'HH:mm:ss[.sss][XXX]`

`timeZone`

 Ask Assistant

## Option

Type: `String`

The `java.time.ZoneId` to use when parsing timestamps and dates.

Default value: None

## `unescapeQuoteHandling`

Type: `String`

The strategy for handling unescaped quotes. Allowed options:

- `STOP_AT_CLOSING_QUOTE`: If unescaped quotes are found in the input, accumulate the quote character and proceed parsing the value as a quoted value, until a closing quote is found.
- `BACK_TO_DELIMITER`: If unescaped quotes are found in the input, consider the value as an unquoted value. This will make the parser accumulate all characters of the current parsed value until the delimiter defined by `sep` is found. If no delimiter is found in the value, the parser will continue accumulating characters from the input until a delimiter or line ending is found.
- `STOP_AT_DELIMITER`: If unescaped quotes are found in the input, consider the value as an unquoted value. This will make the parser accumulate all characters until the delimiter defined by `sep`, or a line ending is found in the input.
- `SKIP_VALUE`: If unescaped quotes are found in the input, the content parsed for the given value will be skipped (until the next delimiter is found) and the value set in `nullValue` will be produced instead.
- `RAISE_ERROR`: If unescaped quotes are found in the input, a `TextParseException` will be thrown.

Default value: `STOP_AT_DELIMITER`

Option	Description	Scope
<code>rowTag</code>	<p>The row tag of the XML files to treat as a row. In the example XML <code>&lt;books&gt; &lt;book&gt;&lt;book&gt;... &lt;books&gt;</code>, the appropriate value is <code>book</code>. This is a required option.</p>	read
<code>samplingRatio</code>	<p>Defines a fraction of rows used for schema inference. XML built-in functions ignore this option. Default: <code>1.0</code>.</p>	read
<code>excludeAttribute</code>	<p>Whether to exclude attributes in elements. Default: <code>false</code>.</p>	read

Option	Description	Scope
<code>mode</code>	<p>Mode for dealing with corrupt records during parsing.</p> <p><code>PERMISSIVE</code>: For corrupted records, puts the malformed string into a field configured by <code>columnNameOfCorruptRecord</code>, and sets malformed fields to <code>null</code>. To keep corrupt records, you can set a <code>string</code> type field named <code>columnNameOfCorruptRecord</code> in a user-defined schema. If a schema does not have the field, corrupt records are dropped during parsing. When inferring a schema, the parser implicitly adds a <code>columnNameOfCorruptRecord</code> field in an output schema.</p> <p><code>DROPMALFORMED</code>: Ignores corrupted records. This mode is unsupported for XML built-in functions.</p> <p><code>FAILFAST</code>: Throws an exception when the parser meets corrupted records.</p>	read
<code>inferSchema</code>	<p>If <code>true</code>, attempts to infer an appropriate type for each resulting DataFrame column. If <code>false</code>, all resulting columns are of <code>string</code> type. Default: <code>true</code>. XML built-in functions ignore this option.</p>	read
<code>columnNameOfCorruptRecord</code>	<p>Allows renaming the new field that contains a malformed string created by <code>PERMISSIVE</code> mode. Default: <code>spark.sql.columnNameOfCorruptRecord</code>.</p>	read

Option	Description	Scope
<code>attributePrefix</code>	The prefix for attributes to differentiate attributes from elements. This will be the prefix for field names. Default is <code>_</code> . Can be empty for reading XML, but not for writing.	read, write
<code>valueTag</code>	The tag used for the character data within elements that also have attribute(s) or child element(s) elements. User can specify the <code>valueTag</code> field in the schema or it will be added automatically during schema inference when character data is present in elements with other elements or attributes. Default: <code>_VALUE</code>	read,write
<code>encoding</code>	For reading, decodes the XML files by the given encoding type. For writing, specifies encoding (charset) of saved XML files. XML built-in functions ignore this option. Default: <code>UTF-8</code> .	read, write
<code>ignoreSurroundingSpaces</code>	Defines whether surrounding white spaces from values being read should be skipped. Default: <code>true</code> . Whitespace-only character data are ignored.	read
<code>rowValidationXSDPath</code>	Path to an optional XSD file that is used to validate the XML for each row individually. Rows that fail to validate are treated like parse errors as above. The XSD does not otherwise affect the schema provided, or inferred.	read

Option	Description	Scope
<code>ignoreNamespace</code>	<p>If <code>true</code>, namespaces' prefixes on XML elements and attributes are ignored. Tags <code>&lt;abc:author&gt;</code> and <code>&lt;def:author&gt;</code>, for example, are treated as if both are just <code>&lt;author&gt;</code>. Namespaces cannot be ignored on the <code>rowTag</code> element, only its read children. XML parsing is not namespace-aware even if <code>false</code>. Default: <code>false</code>.</p>	read
<code>timestampFormat</code>	<p>Custom timestamp format string that follows the <a href="#">datetime pattern</a> format. This applies to <code>timestamp</code> type. Default: <code>yyyy-MM-dd'T'HH:mm:ss[.SSS][XXX]</code>.</p>	read, write
<code>timestampNTZFormat</code>	<p>Custom format string for timestamp without timezone that follows the datetime pattern format. This applies to <code>TimestampNTZType</code> type. Default: <code>yyyy-MM-dd'T'HH:mm:ss[.SSS]</code></p>	read, write
<code>dateFormat</code>	<p>Custom date format string that follows the <a href="#">datetime pattern</a> format. This applies to date type. Default: <code>yyyy-MM-dd</code>.</p>	read, write
<code>locale</code>	<p>Sets a locale as a language tag in IETF BCP 47 format. For instance, <code>locale</code> is used while parsing dates and timestamps. Default: <code>en-US</code>.</p>	read
<code>rootTag</code>	<p>Root tag of the XML files. For example, in <code>&lt;books&gt;&lt;book&gt;&lt;book&gt;...&lt;/books&gt;</code>, the appropriate value is <code>books</code>. You can include basic attributes by</p>	write

Option	Description	Scope
	specifying a value like <code>books foo="bar"</code> . Default: <code>ROWS</code> .	
<code>declaration</code>	<p>Content of XML declaration to write at the start of every output XML file, before the <code>rootTag</code>. For example, a value of <code>foo</code> causes <code>&lt;?xml foo?&gt;</code> to be written. Set to an empty string to suppress.</p> <p>Default: <code>version="1.0" encoding="UTF-8" standalone="yes"</code>.</p>	write
<code>arrayElementName</code>	<p>Name of XML element that encloses each element of an array-valued column when writing.</p> <p>Default: <code>item</code>.</p>	write
<code>nullValue</code>	<p>Sets the string representation of a null value.</p> <p>Default: string <code>null</code>. When this is <code>null</code>, the parser does not write attributes and elements for fields.</p>	read, write
<code>compression</code>	<p>Compression code to use when saving to file.</p> <p>This can be one of the known case-insensitive shortened names (<code>none</code>, <code>bzip2</code>, <code>gzip</code>, <code>lz4</code>, <code>snappy</code>, and <code>deflate</code>). XML built-in functions ignore this option. Default: <code>none</code>.</p>	write
<code>validateName</code>	<p>If true, throws an error on XML element name validation failure. For example, SQL field names can have spaces, but XML element names cannot. Default: <code>true</code>.</p>	write

Option	Description	Scope
<code>readerCaseSensitive</code>	<p>Specifies the case sensitivity behavior when <code>rescuedDataColumn</code> is enabled. If true, rescue the data columns whose names differ by case from the schema; otherwise, read the data in a case-insensitive manner. Default: <code>true</code>.</p>	read
<code>rescuedDataColumn</code>	<p>Whether to collect all data that can't be parsed due to a data type mismatch and schema mismatch (including column casing) to a separate column. This column is included by default when using Auto Loader. For more details, see <a href="#">What is the rescued data column?</a>.</p> <p><code>COPY INTO</code> (legacy) does not support the rescued data column because you cannot manually set the schema using <code>COPY INTO</code>. Databricks recommends using Auto Loader for most ingestion scenarios.</p> <p>Default: None.</p>	read
<code>singleVariantColumn</code>	<p>Specifies the name of the single variant column. If this option is specified for reading, parse the entire XML record into a single Variant column with the given option string value as the column's name. If this option is provided for writing, write the value of the single Variant column to XML files. Default: <code>none</code>.</p>	read, write

## PARQUET options

## Option

`datetimeRebaseMode`

Type: `String`

Controls the rebasing of the DATE and TIMESTAMP values between Julian and Proleptic Gregorian calendars. Allowed values: `EXCEPTION`, `LEGACY`, and `CORRECTED`.

Default value: `LEGACY`

`int96RebaseMode`

Type: `String`

Controls the rebasing of the INT96 timestamp values between Julian and Proleptic Gregorian calendars. Allowed values: `EXCEPTION`, `LEGACY`, and `CORRECTED`.

Default value: `LEGACY`

`mergeSchema`

Type: `Boolean`

Whether to infer the schema across multiple files and to merge the schema of each file.

Default value: `false`

`readerCaseSensitive`

Type: `Boolean`

Specifies the case sensitivity behavior when `rescuedDataColumn` is enabled. If true, rescue the data columns whose names differ by case from the schema; otherwise, read the data in a case-insensitive manner.

## Option

Default value: `true`

`rescuedDataColumn`

Type: `String`

Whether to collect all data that can't be parsed due to: a data type mismatch, and schema mismatch (including column casing) to a separate column. This column is included by default when using Auto Loader. For more details refer to [What is the rescued data column?](#)

`COPY INTO` (legacy) does not support the rescued data column because you cannot manually set the schema using `COPY INTO`. Databricks recommends using Auto Loader for most ingestion scenarios.

Default value: None

# AVRO options

## Option

`avroSchema`

Type: `String`

Optional schema provided by a user in Avro format. When reading Avro, this option can be set to an evolved schema, which is compatible but different with the actual Avro schema. The deserialization schema will be consistent with the evolved schema. For example, if you set an evolved schema containing one additional column with a default value, the read result will contain the new column too.

Default value: None

## Option

### datetimeRebaseMode

Type: `String`

Controls the rebasing of the DATE and TIMESTAMP values between Julian and Proleptic Gregorian calendars. Allowed values: `EXCEPTION`, `LEGACY`, and `CORRECTED`.

Default value: `LEGACY`

### mergeSchema

Type: `Boolean`

Whether to infer the schema across multiple files and to merge the schema of each file. `mergeSchema` for Avro does not relax data types.

Default value: `false`

### readerCaseSensitive

Type: `Boolean`

Specifies the case sensitivity behavior when `rescuedDataColumn` is enabled. If true, rescue the data columns whose names differ by case from the schema; otherwise, read the data in a case-insensitive manner.

Default value: `true`

### rescuedDataColumn

Type: `String`

Whether to collect all data that can't be parsed due to: a data type mismatch, and schema mismatch (including column casing) to a separate column. This column is included by



## Option

default when using Auto Loader.

`COPY INTO` (legacy) does not support the rescued data column because you cannot manually set the schema using `COPY INTO`. Databricks recommends using Auto Loader for most ingestion scenarios.

For more details refer to [What is the rescued data column?](#).

Default value: None

## BINARYFILE options

Binary files do not have any additional configuration options.

## TEXT options

### Option

#### encoding

Type: `String`

The name of the encoding of the TEXT file line separator. For a list of options, see `java.nio.charset.Charset`.

The content of the file is not affected by this option and is read as-is.

Default value: `UTF-8`

#### lineSep

Type: `String`

 Ask Assistant

## Option

A string between two consecutive TEXT records.

Default value: None, which covers `\r`, `\r\n` and `\n`

### wholeText

Type: Boolean

Whether to read a file as a single record.

Default value: `false`

## ORC options

### Option

#### mergeSchema

Type: Boolean

Whether to infer the schema across multiple files and to merge the schema of each file.

Default value: `false`

## Cloud-specific options

Auto Loader provides a number of options for configuring cloud infrastructure.

- [AWS-specific options](#)
- [Azure-specific options](#)

- Google-specific options

## AWS-specific options

Provide the following options only if you choose `cloudFiles.useNotifications` = `true` and you want Auto Loader to set up the notification services for you:

Option
<code>cloudFiles.region</code>
Type: <code>String</code>
The region where the source S3 bucket resides and where the AWS SNS and SQS services will be created.
Default: The region of the EC2 instance.
<code>cloudFiles.restrictNotificationSetupToSameAWSAccountId</code>
<b>Type: Boolean</b> Only allow event notifications from AWS S3 buckets in the same account as the SNS topic. When true, Auto Loader only accepts event notifications from AWS S3 buckets in the same account as the SNS topic.
When <code>false</code> , the access policy does not restrict cross-account bucket and SNS topic setups. This is useful when the SNS topic and bucket path are associated with different accounts.
Available in Databricks Runtime 17.2 and above.
Default: <code>false</code>

Provide the following option only if you choose `cloudFiles.useNotifications` = `true` and you want Auto Loader to use a queue that you have already set up:

## Option

```
cloudFiles.queueUrl
```

Type: `String`

The URL of the SQS queue. If provided, Auto Loader directly consumes events from this queue instead of setting up its own AWS SNS and SQS services.

Default: None

## AWS authentication options

Provide the following authentication option to use a Databricks service credential:

## Option

```
databricks.serviceCredential
```

Type: `String`

The name of your Databricks [service credential](#). Available in Databricks Runtime 16.1 and above.

Default: None

When Databricks service credentials or IAM roles are not available, you can provide the following authentication options instead:

## Options

```
cloudFiles.awsAccessKey
```

## Options

Type: `String`

The AWS access key ID for the user. Must be provided with `cloudFiles.awsSecretKey`.

Default: None

`cloudFiles.awsSecretKey`

Type: `String`

The AWS secret access key for the user. Must be provided with `cloudFiles.awsAccessKey`.

Default: None

`cloudFiles.roleArn`

Type: `String`

The ARN of an IAM role to assume, if needed. The role can be assumed from your cluster's instance profile or by providing credentials with `cloudFiles.awsAccessKey` and `cloudFiles.awsSecretKey`.

Default: None

`cloudFiles.roleExternalId`

Type: `String`

An identifier to provide while assuming a role using `cloudFiles.roleArn`.

Default: None

`cloudFiles.roleSessionName`

 Ask Assistant

## Options

Type: `String`

An optional session name to use while assuming a role using `cloudFiles.roleArn`.

Default: None

`cloudFiles.stsEndpoint`

Type: `String`

An optional endpoint to provide for accessing AWS STS when assuming a role using

`cloudFiles.roleArn`.

Default: None

## Azure-specific options

You must provide values for all of the following options if you specify

`cloudFiles.useNotifications` = `true` and you want Auto Loader to set up the notification services for you:

## Options

`cloudFiles.resourceGroup`

Type: `String`

The Azure Resource Group under which the storage account is created.

Default: None

`cloudFiles.subscriptionId`

## Options

Type: `String`

The Azure Subscription ID under which the resource group is created.

Default: None

`databricks.serviceCredential`

Type: `String`

The name of your Databricks [service credential](#). Available in Databricks Runtime 16.1 and above.

Default: None

If a Databricks service credential is not available, you can provide the following authentication options instead:

## Options

`cloudFiles.clientId`

Type: `String`

The client ID or application ID of the service principal.

Default: None

`cloudFiles.clientSecret`

Type: `String`

The client secret of the service principal.

## Options

Default: None

`cloudFiles.connectionString`

Type: `String`

The connection string for the storage account, based on either account access key or shared access signature (SAS).

Default: None

`cloudFiles.tenantId`

Type: `String`

The Azure Tenant ID under which the service principal is created.

Default: None

### ⓘ IMPORTANT

Automated notification setup is available in Azure China and Government regions with Databricks Runtime 9.1 and later. You must provide a `queueName` to use Auto Loader with file notifications in these regions for older Databricks Runtime versions.

Provide the following option only if you choose `cloudFiles.useNotifications = true` and you want Auto Loader to use a queue that you have already set up:

## Option

`cloudFiles.queueName`

 Ask Assistant

## Option

Type: `String`

The name of the Azure queue. If provided, the cloud files source directly consumes events from this queue instead of setting up its own Azure Event Grid and Queue Storage services. In that case, your `databricks.serviceCredential` or `cloudFiles.connectionString` requires only read permissions on the queue.

Default: None

## Google-specific options

Auto Loader can automatically set up notification services for you by leveraging Databricks service credentials. The service account created with the Databricks service credential will require the permissions specified in [Configure Auto Loader streams in file notification mode](#).

### Options

`cloudFiles.projectId`

Type: `String`

The id of the project that the GCS bucket is in. The Google Cloud Pub/Sub subscription will also be created within this project.

Default: None

`databricks.serviceCredential`

Type: `String`

The name of your Databricks service credential. Available in Databricks Runtime 16.1 and above.

## Options

Default: None

If a Databricks service credential is not available, you can use Google Service Accounts directly. You can either configure your cluster to assume a service account by following [Google service setup](#) or provide the following authentication options directly:

## Options

`cloudFiles.client`

Type: `String`

The client ID of the Google Service Account.

Default: None

`cloudFiles.clientEmail`

Type: `String`

The email of the Google Service Account.

Default: None

`cloudFiles.privateKey`

Type: `String`

The private key that's generated for the Google Service Account.

Default: None

## Options

`cloudFiles.privateKeyId`

Type: `String`

The ID of the private key that's generated for the Google Service Account.

Default: None

Provide the following option only if you choose `cloudFiles.useNotifications = true` and you want Auto Loader to use a queue that you have already set up:

## Option

`cloudFiles.subscription`

Type: `String`

The name of the Google Cloud Pub/Sub subscription. If provided, the cloud files source consumes events from this queue instead of setting up its own GCS Notification and Google Cloud Pub/Sub services.

Default: None