Last updated on ***Oct 8, 2025***

What is Auto Loader?

Auto Loader incrementally and efficiently processes new data files as they arrive in cloud storage without any additional setup.

How does Auto Loader work?

Auto Loader incrementally and efficiently processes new data files as they arrive in cloud storage. It provides a Structured Streaming source called `cloudFiles`. Given an input directory path on the cloud file storage, the `cloudFiles` source automatically processes new files as they arrive, with the option of also processing existing files in that directory. Auto Loader has support for both Python and SQL in Lakeflow Spark Declarative Pipelines.

You can use Auto Loader to process billions of files to migrate or backfill a table. Auto Loader scales to support near real-time ingestion of millions of files per hour.

Supported Auto Loader sources

Auto Loader can load data files from the following sources:

- Amazon S3 (`s3://`)
- Azure Data Lake Storage (ADLS, `abfss://`)
- Google Cloud Storage (GCS, `gs://`)
- Azure Blob Storage (`wasbs://`)

> **NOTE**

Ask Assistant

> The legacy Windows Azure Storage Blob driver (WASB) has been deprecated. ABFS has numerous benefits over WASB. See Azure documentation on ABFS. For documentation for working with the legacy WASB driver, see Connect to Azure Blob Storage with WASB (legacy).

- Databricks File System (DBFS, `dbfs:/`).

Auto Loader can ingest `JSON`, `CSV`, `XML`, `PARQUET`, `AVRO`, `ORC`, `TEXT`, and `BINARYFILE` file formats.

How does Auto Loader track ingestion progress?

As files are discovered, their metadata is persisted in a scalable key-value store (RocksDB) in the *checkpoint location* of your Auto Loader pipeline. This key-value store ensures that data is processed exactly once.

In case of failures, Auto Loader can resume from where it left off by information stored in the checkpoint location and continue to provide exactly-once guarantees when writing data into Delta Lake. You don't need to maintain or manage any state yourself to achieve fault tolerance or exactly-once semantics.

Incremental ingestion using Auto Loader with Lakeflow Spark Declarative Pipelines

Databricks recommends Auto Loader in Lakeflow Spark Declarative Pipelines for incremental data ingestion. Lakeflow Spark Declarative Pipelines extends functionality in Apache Spark Structured Streaming and allows you to write just a few lines of declarative Python or SQL to deploy a production-quality data pipeline with:

- Autoscaling compute infrastructure for cost savings
- Data quality checks with expectations
- Automatic schema evolution handling
- Monitoring via metrics in the event log

Ask Assistant

You do not need to provide a schema or checkpoint location because Lakeflow Spark Declarative Pipelines automatically manages these settings for your pipelines. See Load data in pipelines.

Databricks also recommends Auto Loader whenever you use Apache Spark Structured Streaming to ingest data from cloud object storage. APIs are available in Python and Scala.

Get started with Databricks Auto Loader

See the following articles to get started configuring incremental data ingestion using Auto Loader with Lakeflow Spark Declarative Pipelines:

- Tutorial: Build an ETL pipeline with Lakeflow Spark Declarative Pipelines
- Onboard data from Amazon S3
- Load data from cloud object storage into streaming tables using Auto Loader (Databricks SQL Editor)

Examples: Common Auto Loader patterns

For examples of common Auto Loader patterns, see Common data loading patterns.

Configure Auto Loader options

You can tune Auto Loader based on data volume, variety, and velocity.

- Configure schema inference and evolution in Auto Loader
- Configure Auto Loader for production workloads

For a full list of Auto Loader options, see:

- Auto Loader options

If you encounter unexpected performance, see the FAQ.

Ask Assistant

Configure Auto Loader file detection modes

Auto Loader supports two file detection modes. See:

- Auto Loader streams with directory listing mode
- Configure Auto Loader streams in file notification mode

Benefits of Auto Loader over using Structured Streaming directly on files

In Apache Spark, you can read files incrementally using `spark.readStream.format(fileFormat).load(directory)`. Auto Loader provides the following benefits over the file source:

- Scalability: Auto Loader can discover billions of files efficiently. Backfills can be performed asynchronously to avoid wasting any compute resources.
- Performance: The cost of discovering files with Auto Loader scales with the number of files that are being ingested instead of the number of directories that the files may land in. See Auto Loader streams with directory listing mode.
- Schema inference and evolution support: Auto Loader can detect schema drifts, notify you when schema changes happen, and rescue data that would have been otherwise ignored or lost. See How does Auto Loader schema inference work?.
- Cost: Auto Loader uses native cloud APIs to get lists of files that exist in storage. In addition, Auto Loader's file notification mode can help reduce your cloud costs further by avoiding directory listing altogether. Auto Loader can automatically set up file notification services on storage to make file discovery much cheaper.

Ask Assistant