Data Ingestion with Apache Sqoop and Apache Flume

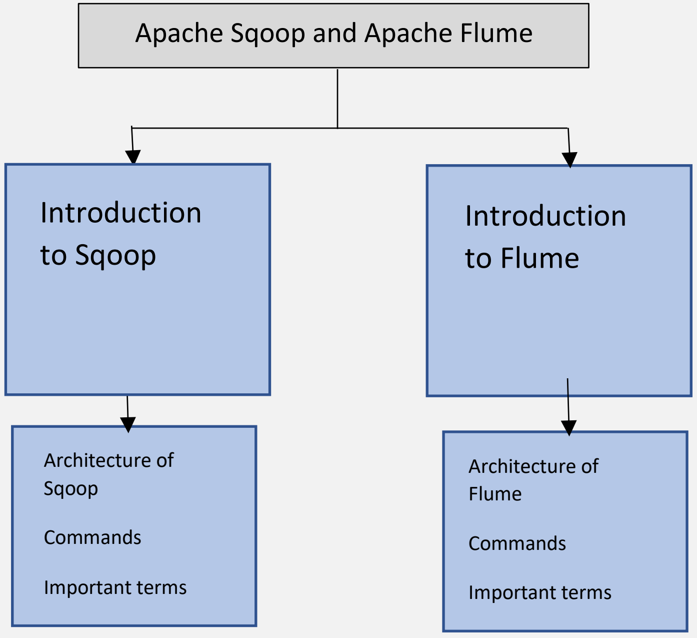
Sqoop is a tool designed to transfer data between Hadoop and relational database servers. Apache Flume is used to collect log data present in log files from web servers and aggregating it into HDFS for analysis.

As a part of Data Ingestion with Apache Sqoop and Apache Flume, you covered:

- Introduction to Sqoop
- Architecture and commands of Sqoop
- Introduction to Flume
- Flume Architecture and commands

Common Interview Questions:

1. What is the role of JDBC driver in a Sqoop set up?
2. How can you import only a subset of rows form a table?
3. How do you fetch data which is the result of join between two tables?
4. How can we slice the data to be imported to multiple parallel tasks?
5. How can Flume be used with HBase?
6. What is sink process?
7. What is flume agent and flume event?
8. What are use cases of Apache Flume?
9. What are possible types of Channel Selectors?



Data Ingestion with Apache Sqoop and Apache Flume

Apache Sqoop:

Sqoop is a tool designed to transfer data between Hadoop and relational database servers. It is used to import data from relational databases such as MySQL, Oracle to Hadoop HDFS, and export from Hadoop file system to relational databases.

Apache Flume:

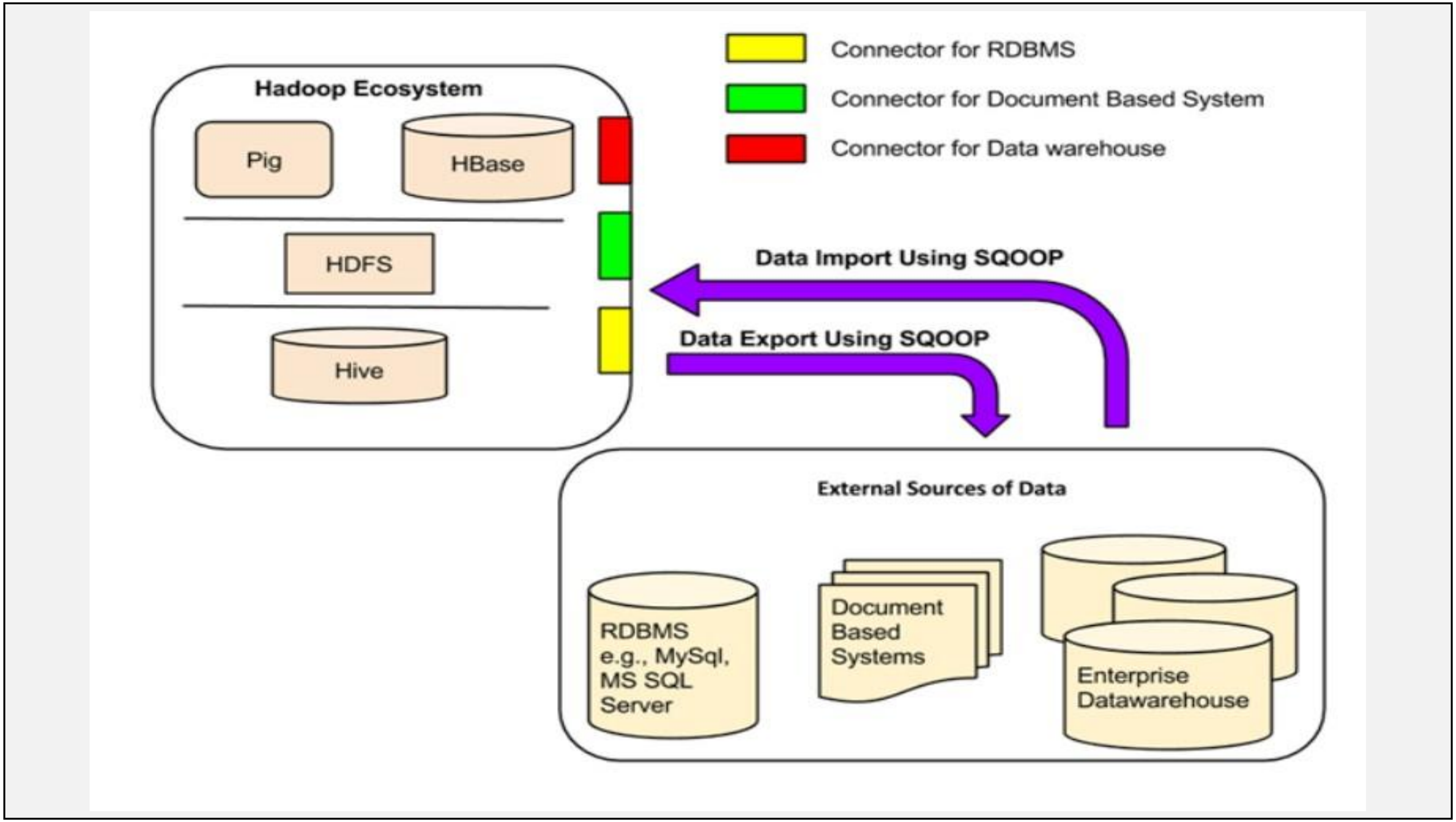
Apache Flume is a tool/service/data ingestion mechanism for collecting aggregating and transporting large amounts of streaming data such as log files, events (etc...) from various sources to a centralized data store.

Apache Sqoop VS Apache Flume:

Based on	Sqoop	Flume
Basic Nature	Sqoop works well with any RDBMS which has JDBC like Oracle, MySQL, Teradata, etc.	Flume works well for Streaming data source which is continuously generating such as logs, JMS, directory etc.
Data Flow	Specifically used for parallel data transfer. For this reason, the output could be in multiple files	Used for collecting and aggregating data because of its distributed nature.
Architecture	Follows connector-based architecture, which means connectors, knows how to connect to a different data source.	Follows agent-based architecture, where the code written in it is known as an agent that is responsible for fetching data.
Performance	Reduces excessive storage and processing loads by transferring them to other systems and has fast performance.	It is fault-tolerant, robust and has a tenable reliability mechanism for failover and recovery.

Data Ingestion with Apache Sqoop and Apache Flume

Sqoop Architecture:



Data Ingestion with Apache Sqoop and Apache Flume

Important Terminologies:

Sqoop Import:

The import tool imports individual tables from RDBMS to HDFS. Each row in a table is treated as a record in HDFS. All records are stored as text data in text files or as binary data in Avro and Sequence files.

Sqoop Export:

The export tool exports a set of files from HDFS back to an RDBMS. The files given as input to Sqoop contain records, which are called as rows in table. Those are read and parsed into a set of records and delimited with user-specified delimiter.

General Commands in Sqoop:

Command	Syntax
Sqoop Import Import data into HDFS.	$ sqoop import (generic-args) (import-args)
Sqoop Export To export data from HDFS to RDBMS database.	$ sqoop export (generic-args) (export-args)
Sqoop Job Creates and saves the import and export commands	$ sqoop job (generic-args) (job-args) [-- [subtool-name] (subtool-args)]
Codegen It generates DAO class in Java, based on the Table Schema structure	$ sqoop codegen (generic-args) (codegen-args)
Eval It allows users to execute user-defined queries against respective database servers.	$ sqoop eval (generic-args) (eval-args)

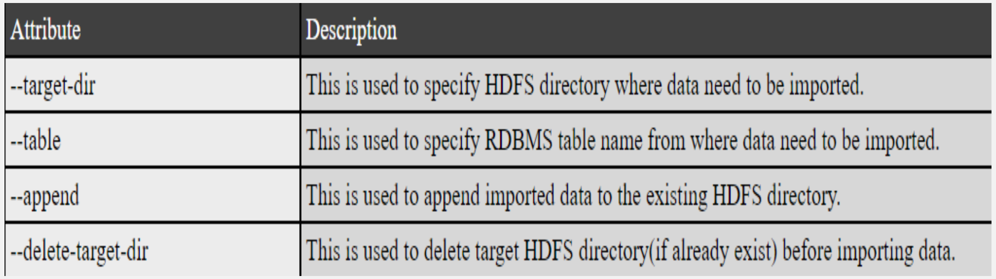
Data Ingestion with Apache Sqoop and Apache Flume

Sqoop Import:

Sqoop Import to HDFS:

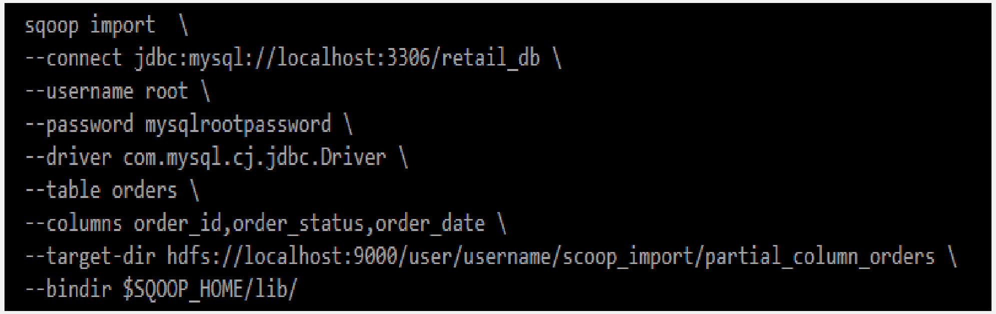
Sqoop can be used to import data seamlessly into HDFS from RDBMS systems.

Generic Arguments to import command:



Attribute	Description
--target-dir	This is used to specify HDFS directory where data need to be imported.
--table	This is used to specify RDBMS table name from where data need to be imported.
--append	This is used to append imported data to the existing HDFS directory.
--delete-target-dir	This is used to delete target HDFS directory(if already exist) before importing data.

Import Specific columns: "--columns" argument can be used to import specific columns.

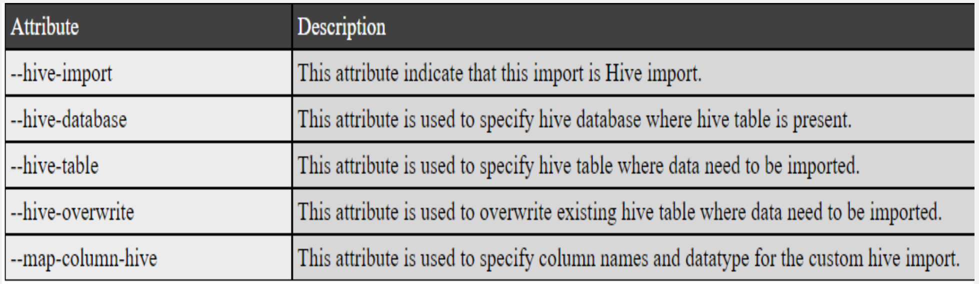


```
sqoop import  \
--connect jdbc:mysql://localhost:3306/retail_db \
--username root \
--password mysqlrootpassword \
--driver com.mysql.cj.jdbc.Driver \
--table orders \
--columns order_id,order_status,order_date \
--target-dir hdfs://localhost:9000/user/username/scoop_import/partial_column_orders \
--bindir $SQOOP_HOME/lib/
```

Sqoop Hive Import:

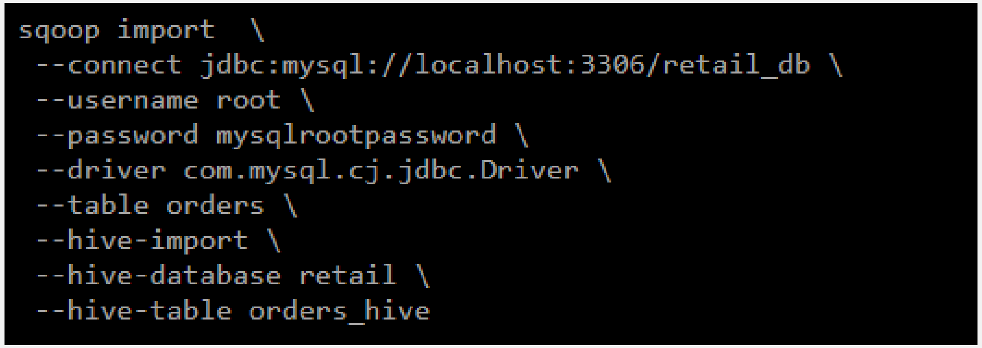
Sqoop can be used to import data seamlessly into Hive tables from RDBMS systems.

Generic Arguments to Hive Import command:



Attribute	Description
--hive-import	This attribute indicate that this import is Hive import.
--hive-database	This attribute is used to specify hive database where hive table is present.
--hive-table	This attribute is used to specify hive table where data need to be imported.
--hive-overwrite	This attribute is used to overwrite existing hive table where data need to be imported.
--map-column-hive	This attribute is used to specify column names and datatype for the custom hive import.

Simple Hive Import: This will import data to Hive table and import utility will create table if not present.



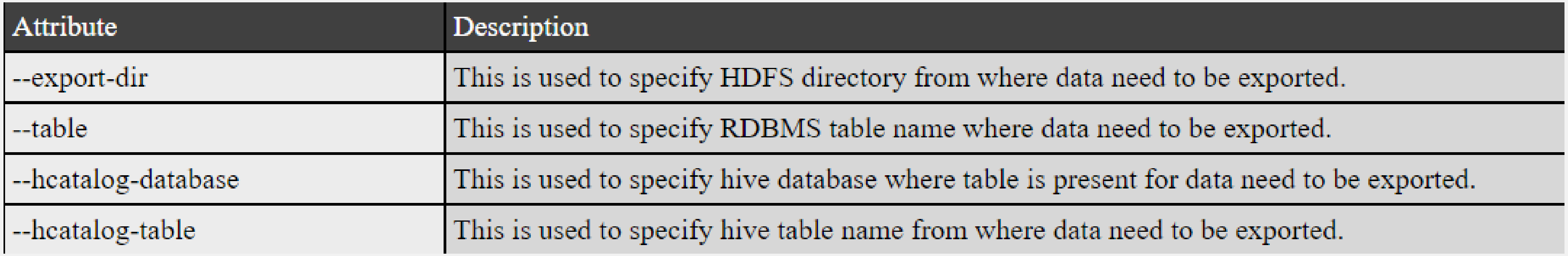
```
sqoop import  \
 --connect jdbc:mysql://localhost:3306/retail_db \
 --username root \
 --password mysqlrootpassword \
 --driver com.mysql.cj.jdbc.Driver \
 --table orders \
 --hive-import \
 --hive-database retail \
 --hive-table orders_hive
```

Sqoop Export:

Sqoop export to HDFS and Hive:

Sqoop can be used to export data seamlessly from HDFS into RDBMS systems and Hive to RDBMS.

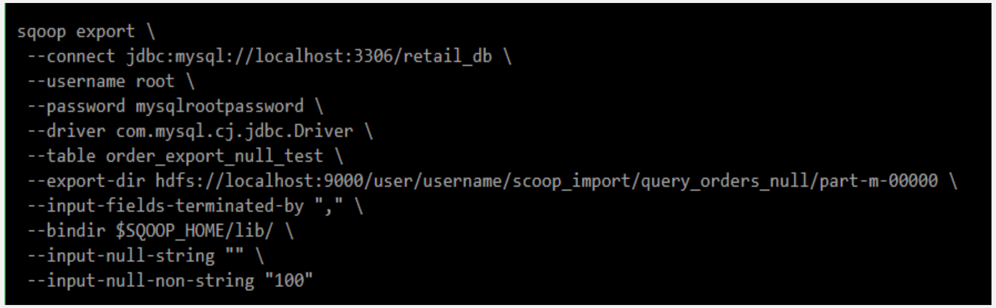
Generic Arguments to export command:



Attribute	Description
--export-dir	This is used to specify HDFS directory from where data need to be exported.
--table	This is used to specify RDBMS table name where data need to be exported.
--hcatalog-database	This is used to specify hive database where table is present for data need to be exported.
--hcatalog-table	This is used to specify hive table name from where data need to be exported.

Sqoop Export from HDFS:

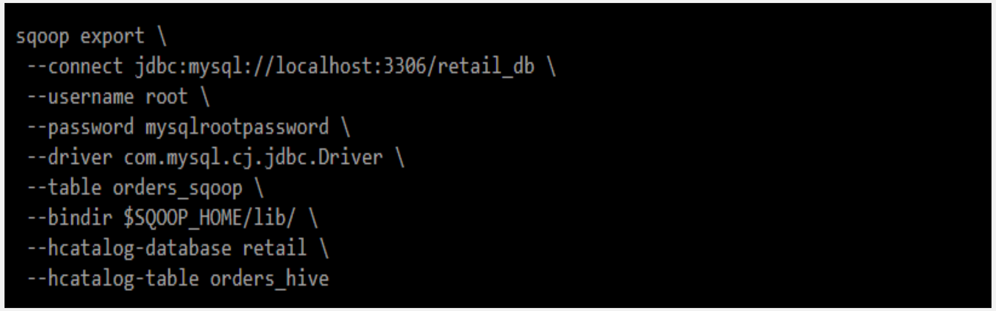
Export Data with Nulls: If nulls are not handled properly then null data may be exported as blank string for string columns. There are different arguments to handle nulls in string and number. Both " --input-null-string" & "--input-null-non-string" clauses can be used in a single export.



```
sqoop export \
 --connect jdbc:mysql://localhost:3306/retail_db \
 --username root \
 --password mysqlrootpassword \
 --driver com.mysql.cj.jdbc.Driver \
 --table order_export_null_test \
 --export-dir hdfs://localhost:9000/user/username/scoop_import/query_orders_null/part-m-00000 \
 --input-fields-terminated-by "," \
 --bindir $SQOOP_HOME/lib/ \
 --input-null-string "" \
 --input-null-non-string "100"
```

Sqoop Export from Hive:

Export Hive table: "--hcatalog-database" and "--hcatalog-table" attributes can be used to specify hive database and tablename from where data need to be exported.



```
sqoop export \
 --connect jdbc:mysql://localhost:3306/retail_db \
 --username root \
 --password mysqlrootpassword \
 --driver com.mysql.cj.jdbc.Driver \
 --table orders_sqoop \
 --bindir $SQOOP_HOME/lib/ \
 --hcatalog-database retail \
 --hcatalog-table orders_hive
```

Data Ingestion with Apache Sqoop and Apache Flume

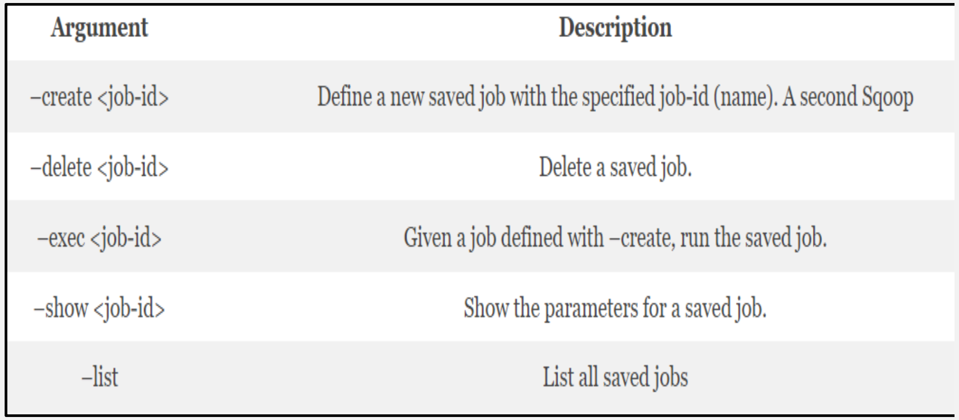
Sqoop Job:

Sqoop Job:

Syntax of Sqoop Job

$ sqoop job (generic-args) (job-args) [– [subtool-name] (subtool-args)]

$ sqoop-job (generic-args) (job-args) [– [subtool-name] (subtool-args)]



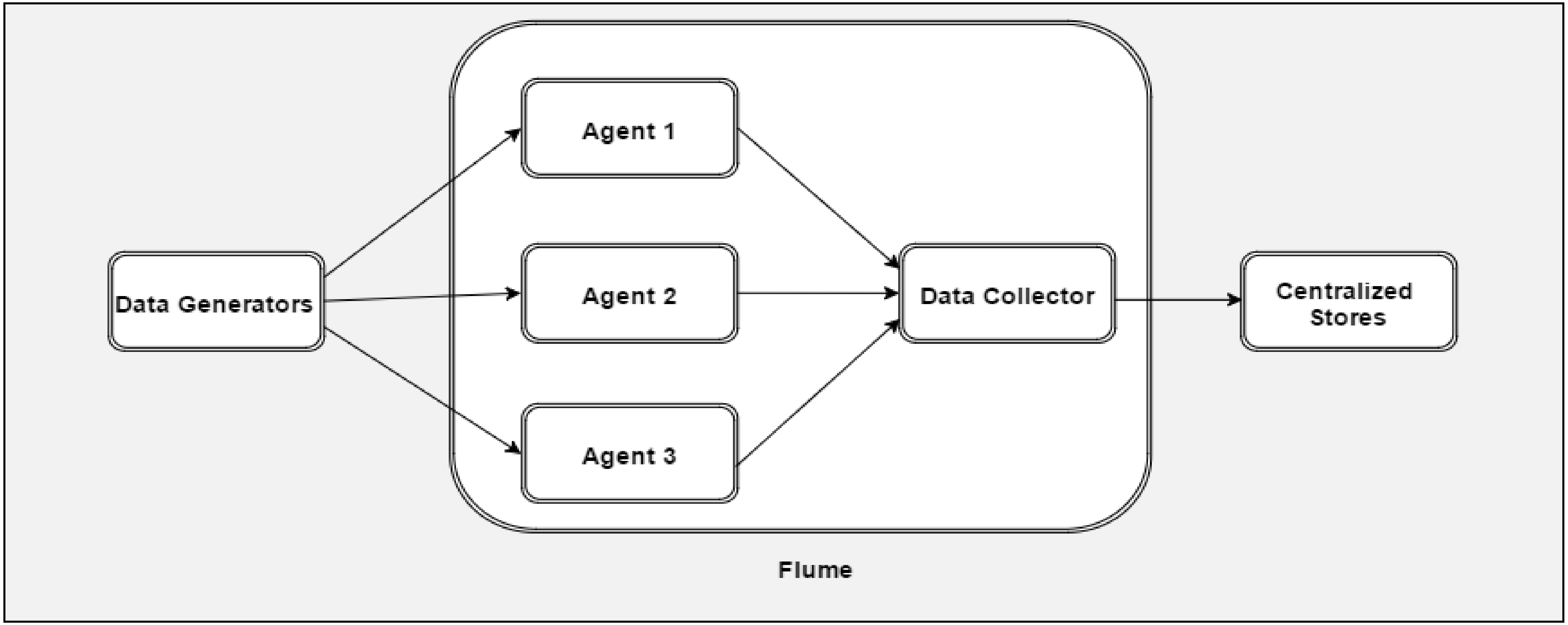
Argument	Description
–create <job-id>	Define a new saved job with the specified job-id (name). A second Sqoop
–delete <job-id>	Delete a saved job.
–exec <job-id>	Given a job defined with –create, run the saved job.
–show <job-id>	Show the parameters for a saved job.
–list	List all saved jobs

Sqoop $CONDITIONS:

Sqoop performs highly efficient data transfers by inheriting Hadoop's parallelism.

- To help Sqoop split your query into multiple chunks that can be transferred in parallel, you need to include the $CONDITIONS placeholder in the where clause of your query.
- Sqoop will automatically substitute this placeholder with the generated conditions specifying which slice of data should be transferred by each individual task.
- While you could skip $CONDITIONS by forcing Sqoop to run only one job using the --num-mappers 1 param- eter, such a limitation would have a severe performance impact.

Flume Architecture:



Data generators (such as Facebook, Twitter) generate data which gets collected by individual **Flume agents** running on them. Thereafter, a **data collector** (which is also an agent) collects the data from the agents which is aggregated and pushed into a centralized store such as HDFS or HBase.

Data Ingestion with Apache Sqoop and Apache Flume

Important Terminologies:

Flume Agent:

An agent is an independent daemon process (JVM) in Flume. It receives the data (events) from clients or other agents and forwards it to its next destination (sink or agent). Flume may have more than one agent.

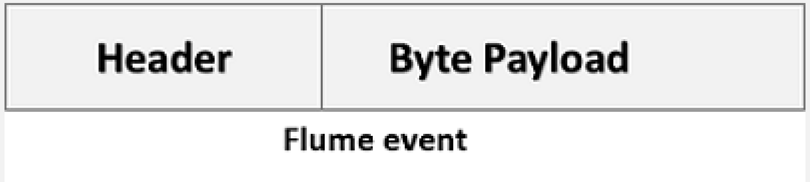
Source: A source is the component of an Agent which receives data from the data generators and transfers it to one or more channels in the form of Flume events.

Channel: A channel is a transient store which receives the events from the source and buffers them till they are consumed by sinks. Channel selectors types are default and multiplexing channel selector.

Sink: A sink stores the data into centralized stores like HBase and HDFS. It consumes the data (events) from the channels and delivers it to the destination. A sink processor is used to invoke a particular sink from the selected group of sinks.

Flume Event:

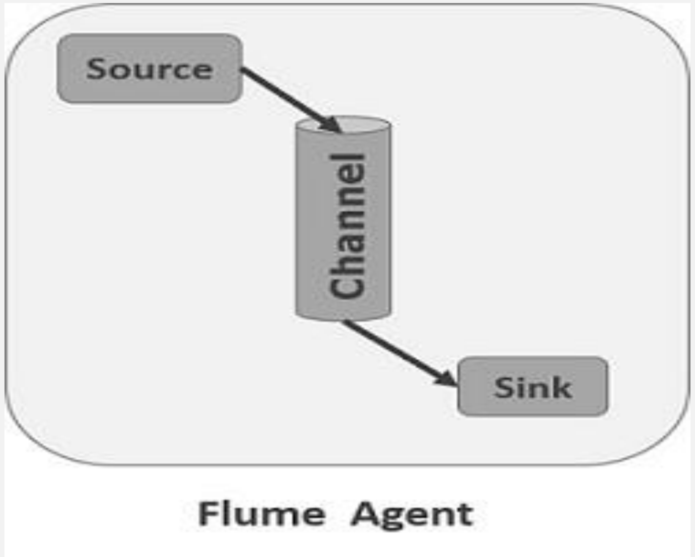
An event is the basic unit of the data transported inside Flume. It contains a payload of byte array that is to be transported from the source to the destination accompanied by optional headers. A typical Flume event would have the following structure:



Header	Byte Payload

Flume event

Structure of Flume Agent:

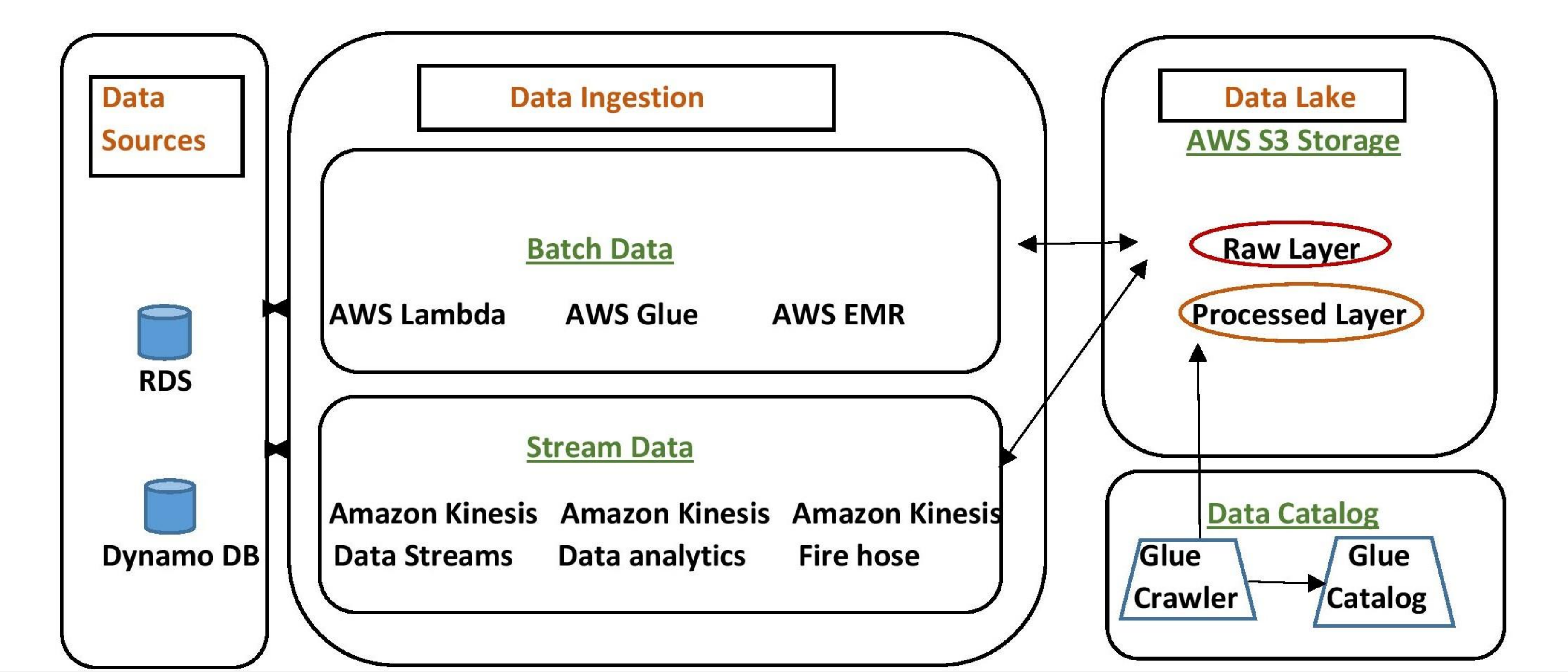


Flume Agent

AWS Glue and Data Ingestion:

AWS Glue is a serverless data integration service that makes it easier to discover, prepare, move, and integrate data from multiple sources for analytics, machine learning (ML), and application development.

End to End workflow for Data Ingestion using Amazon Web


Data Sources
RDS
Dynamo DB
Data Ingestion
Batch Data
AWS Lambda
AWS Glue
AWS EMR
Stream Data
Amazon Kinesis Data Streams
Amazon Kinesis Data analytics
Amazon Kinesis Fire hose
Data Lake
AWS S3 Storage
Raw Layer
Processed Layer
Data Catalog
Glue Crawler
Glue Catalog