# Outline

Module 1：大數據簡介

Module 2：Hadoop Ecosystem介紹

Module 3：Hadoop 平台安裝

Module 4：Hadoop 分散式檔案系統（HDFS）

Module 5：Hadoop MapReduce

Module 6：Apache Hive

**Module 7：Sqoop與Flume**

Module 8：Apache Spark

Module 9：Spark 平台安裝

Module 10：RDD － Resilient distributed dataset

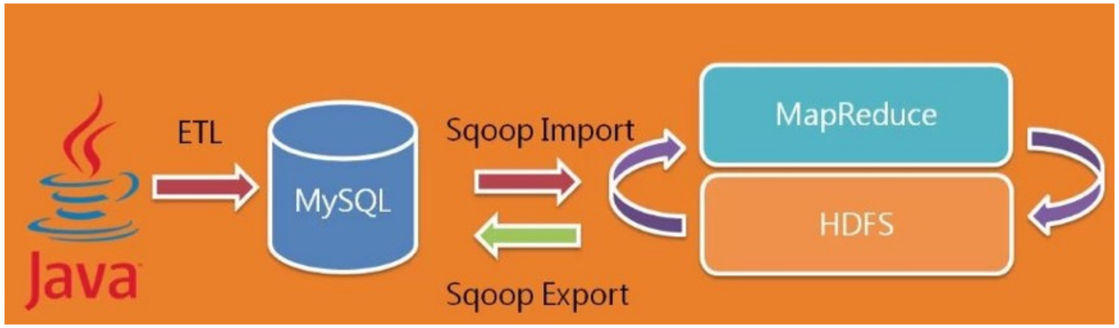
Module 11：Scala 程式開發基礎

Module 12：Spark SQL 及 DataFrame

Module 13：Spark 機器學習函式庫(MLlib)

# Sqoop介紹

- Sqoop = **SQL to Hadoop**
- 處理**結構化數據**
- 使用者透過 Sqoop **由關聯式資料庫(RDBMS)中擷取資料到 Hadoop**，供後續分析使用。
- 也能**將分析結果透過 Sqoop 匯入資料庫**，供其他用戶端程式使用。



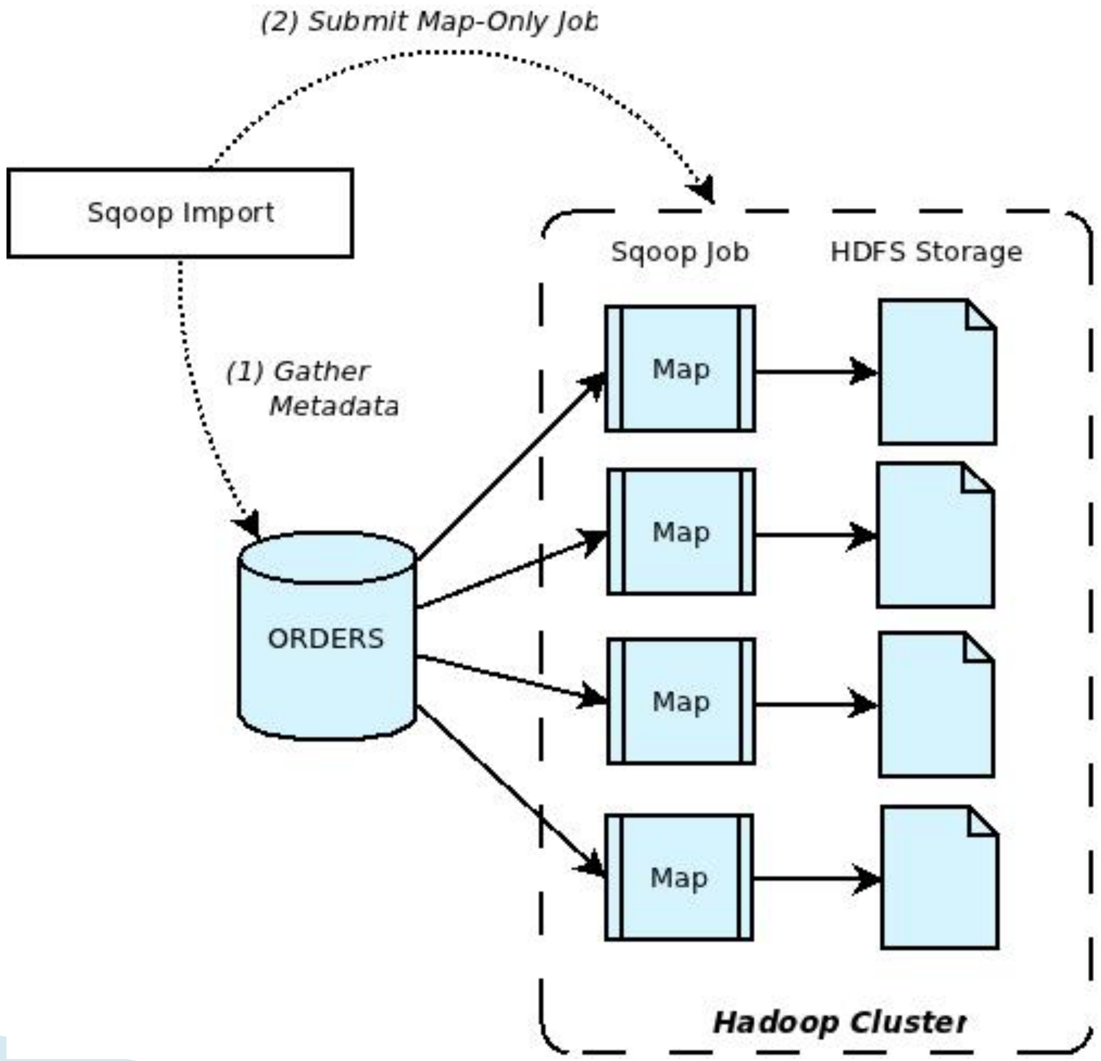
## Sqoop 1 介紹

- 可以交換資料庫及HDFS內的資料
  - 支援RDBMS、HBase等
  - 透過JDBC作連結
- 使用MapReduce中的Map來查詢或新增資料
  - 預設使手4個Mapper
  - 可限制佔用的頻寬
- 無法進行過濾或去除重覆資料等處理
- 通常是搭配排程(如Oozie)定期作匯出匯入動作

# Sqoop 使用語法

```
user@master ~ $ sqoop help
usage: sqoop COMMAND [ARGS]

Available commands:
  codegen            Generate code to interact with database records
  create-hive-table  Import a table definition into Hive
  eval               Evaluate a SQL statement and display the results
  export             Export an HDFS directory to a database table
  help               List available commands
  import             Import a table from a database to HDFS
  import-all-tables  Import tables from a database to HDFS
  job                Work with saved jobs
  list-databases     List available databases on a server
  list-tables        List available tables in a database
  merge              Merge results of incremental imports
  metastore          Run a standalone Sqoop metastore
  version            Display version information

See 'sqoop help COMMAND' for information on a specific command.
```

# Sqoop Import (匯入)



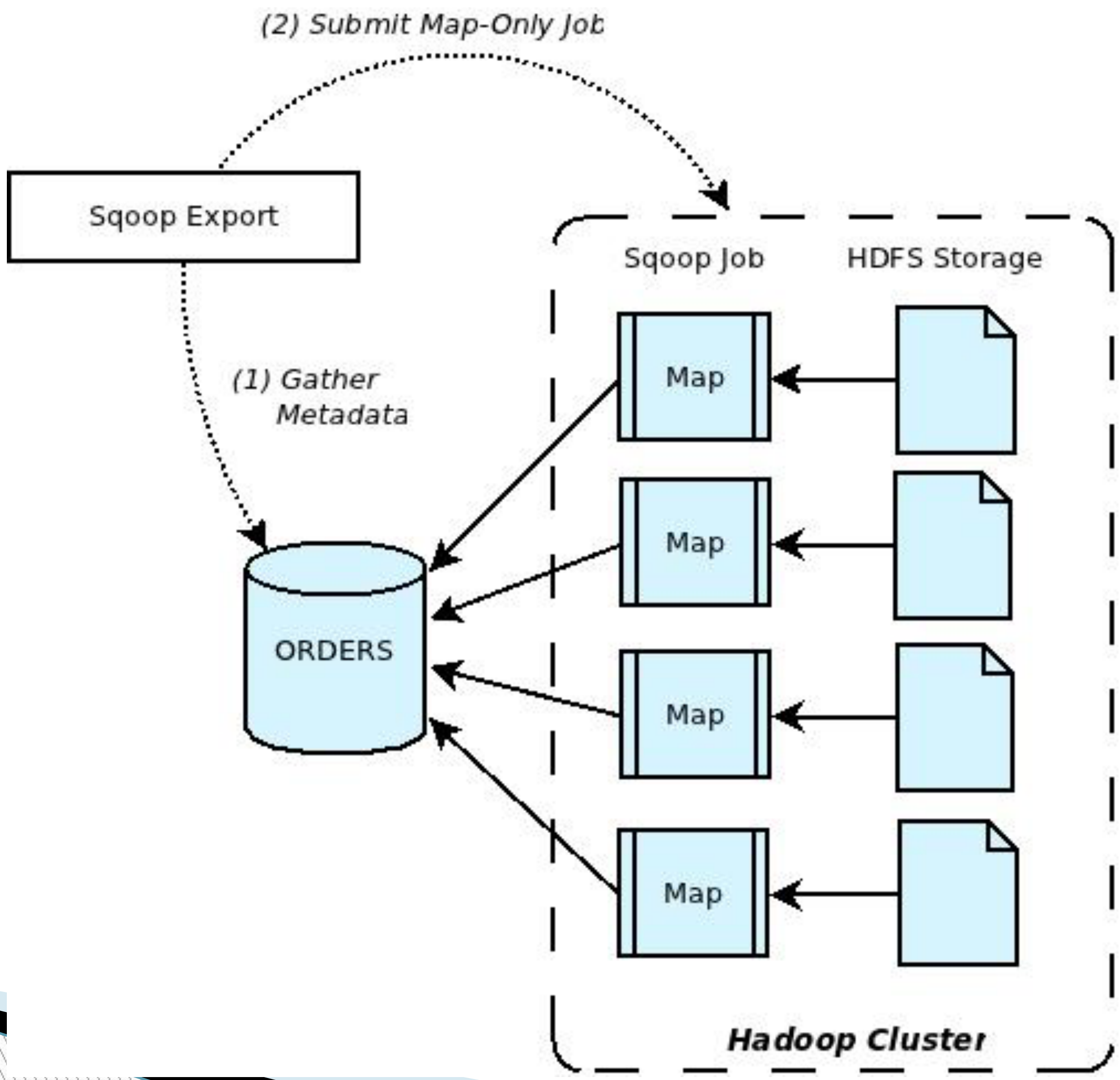
# Sqoop Import 語法 (1)

```
user@master ~ $ sqoop help import
usage: sqoop import [GENERIC-ARGS] [TOOL-ARGS]

Common arguments:

--connect [jdbc-uri]              Specify JDBC connect string
-P                                Read password from console password
--username [username]             Set authentication username

Import control arguments:

--append                          Imports data in append mode
--as-avrodatafile                 Imports data to Avro data files
--as-sequencefile                 Imports data to SequenceFiles
--as-textfile                     Imports data as plain text (default)
--columns [col,col,col...]        Columns to import from table
-e,--query [statement]            Import results of SQL 'statement'
-m,--num-mappers [n]              Use 'n' map tasks to import in parallel
--table [table-name]              Table to read
--target-dir [dir]                HDFS plain table destination
--where [where clause]            WHERE clause to use during import
```

# Sqoop Import 語法 (2)

```
Hive arguments:

--create-hive-table                  Fail if the target hive table exists
--hive-import                        Import tables into Hive
--hive-table [table-name]            Sets the table name to use when importing to hive

HBase arguments:

--column-family [family]             Sets the target column family for the import
--hbase-create-table                 If specified, create missing HBase tables
--hbase-row-key [col]                Specifies which input column to use as the row key
--hbase-table [table]                Import to [table] in HBase

At minimum, you must specify --connect and --table
```

# Sqoop Export (匯出)



# Sqoop Export 語法

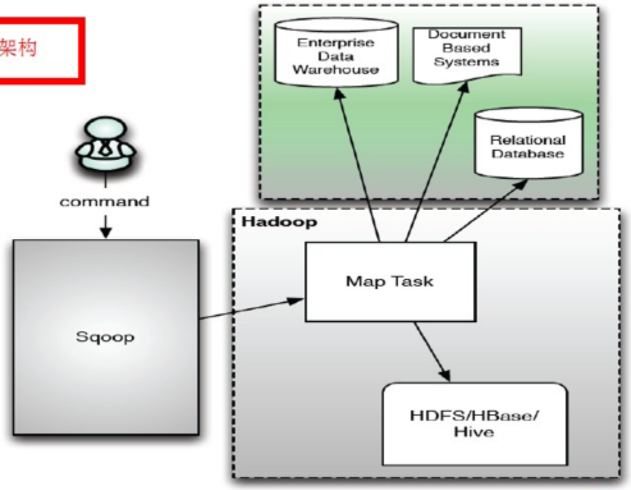
```
user@master ~ $ sqoop help export
usage: sqoop export [GENERIC-ARGS] [TOOL-ARGS]

Common arguments:

   --connect [jdbc-uri]            Specify JDBC connect string
-P                                 Read password from console
   --username [username]           Set authentication username

Export control arguments:

   --columns [col,col,col...]      Columns to export to table
   --export-dir [dir]              HDFS source path for the export
-m,--num-mappers [n]               Use 'n' map tasks to export in
parallel
   --table [table-name]            Table to populate

At minimum, you must specify --connect, --export-dir, and --table
```

# Sqoop1 vs Sqoop2

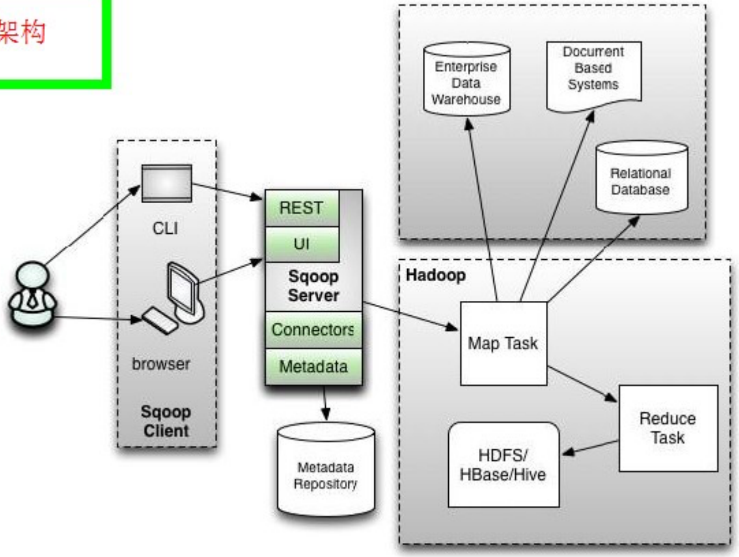
- **Sqoop1**
  - Client Model(Connector及Driver都安裝於Client)
  - 只啟動Map Task
  - 安全性較差(連線帳密包含在client端指令中)
  - 1.4.X版
- **Sqoop2**
  - Server Model(Connector及Driver都安裝於Server)
  - 可使用Map Task及Reduce Task
  - 提供更佳安全性(連線由Server控管、權限管理)
  - 1.99.X版

# Sqoop1 vs Sqoop2

sqoop1架构

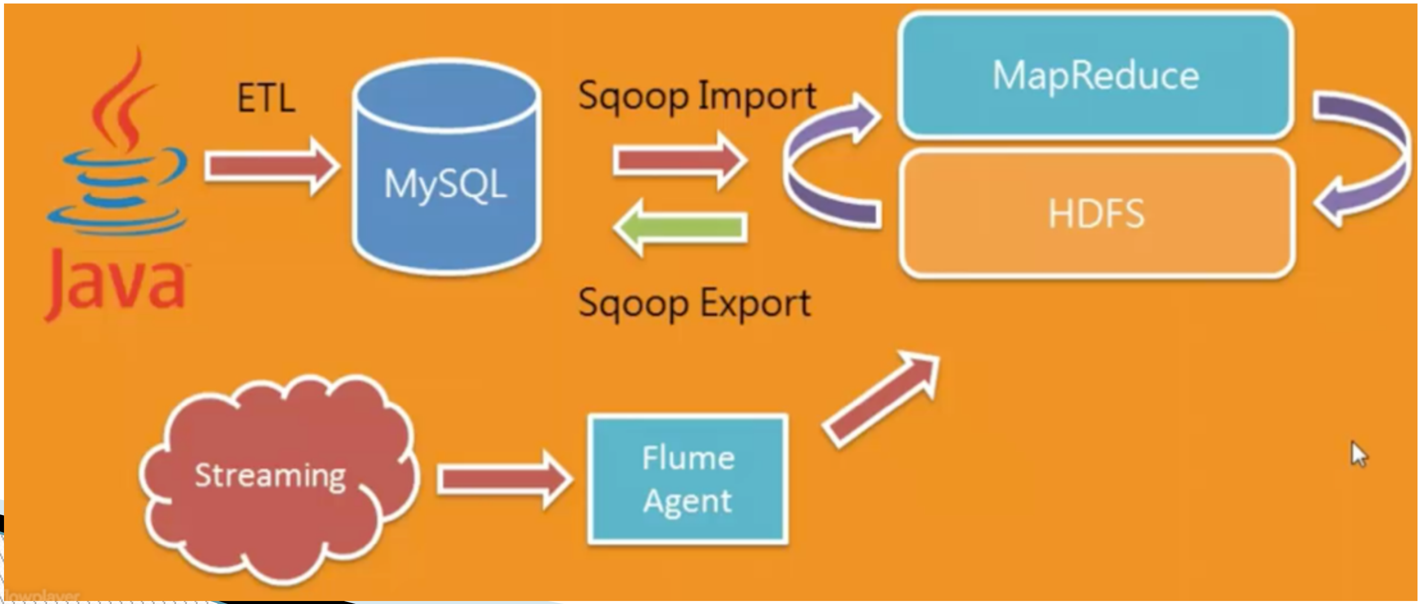


sqoop2架构



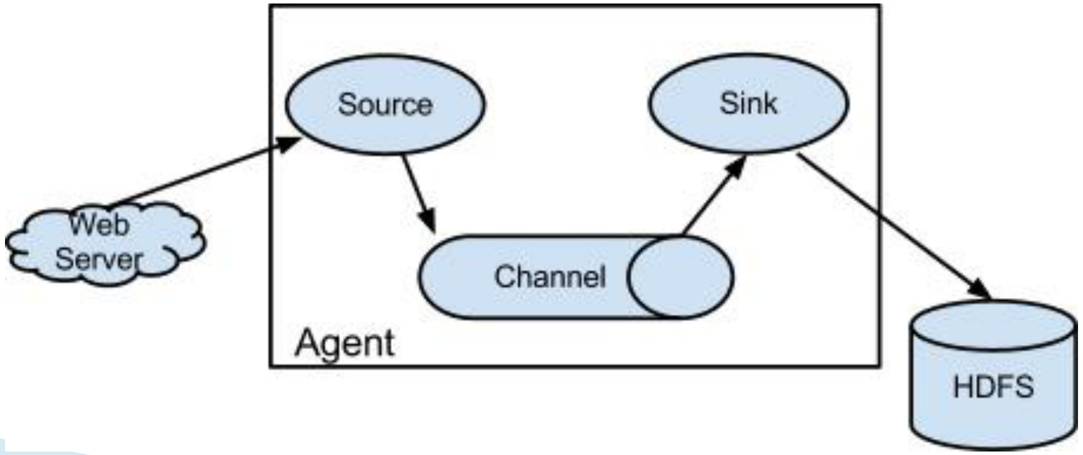
## Flume介紹

- 由Cloudera推出的高效能、高可用性的分散式Log收集系統，現由apache維護
- 在Hadoop Ecosystem中被用來**處理非結構化資料**(Streaming、Log等)



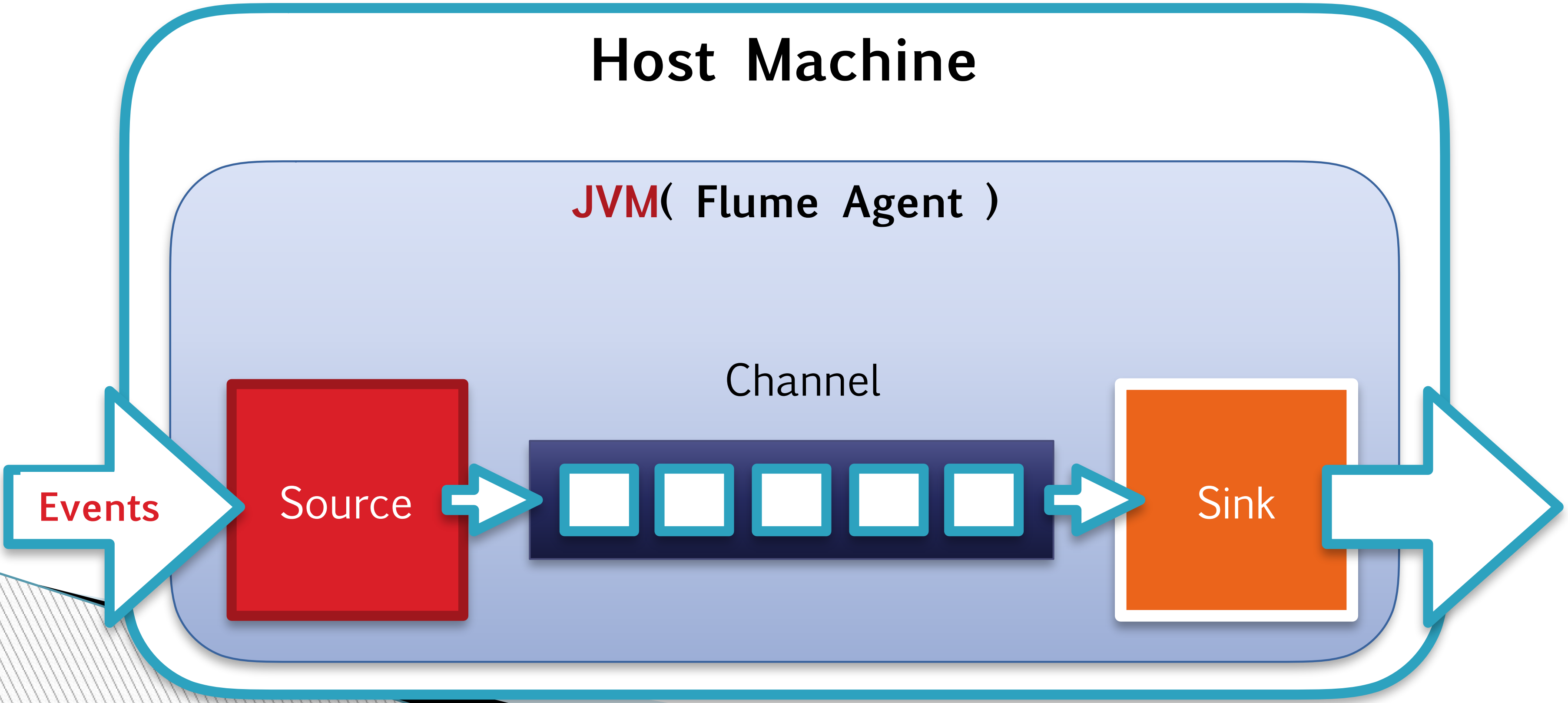
# Flume的組成元件

- **Agent**
  - Flume的執行單位。每台機器運行一個agent，但一個agent可包含多個sources和sinks
- **Client**
  - 資料產生來源，如Web  Server
- **Source**
  - 從Client收集資料，傳遞給Channel
- **Sink**
  - 從Channel收集數據
- **Channel**
  - sources和sinks的buffer
  - 以**Queue**的概念實作

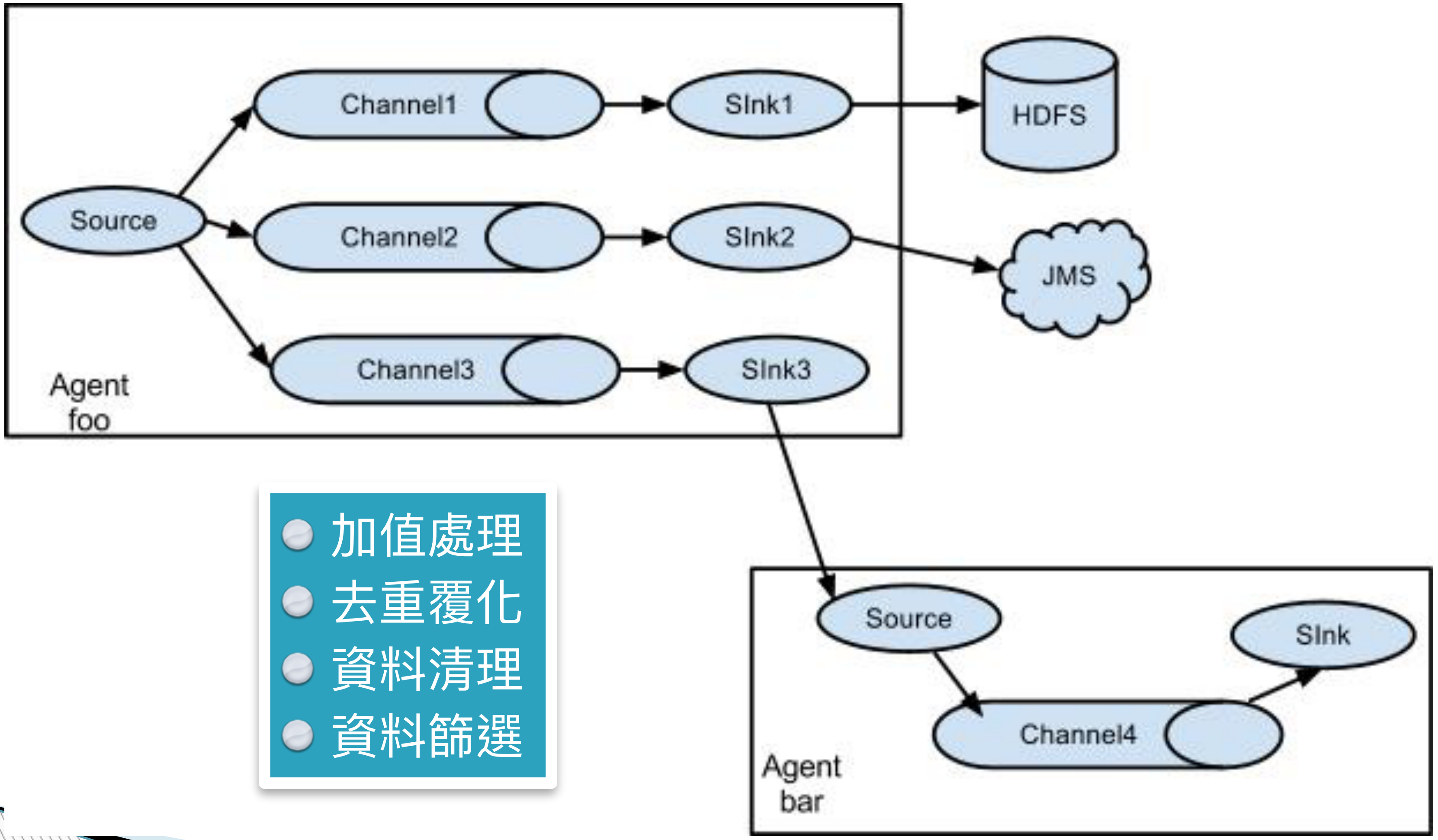


# Flume的組成元件

- **Events**
  - 代表輸入的資料來源，可以是Web Log、JMS、 avro 對象等
  - Events方向為**單向(Input Only**，與Sqoop雙向不同)



# Flume的佈署架構



- 加值處理
- 去重覆化
- 資料清理
- 資料篩選

# Source、Sink、Channel

- **Source**
  - 以Event Driven或Polling based方式取得資料
  - 由特定目錄或資料取得資料(tail -f file)
  - 由特定連結取得資料(Socket等)
- **Sink**
  - Polling based運作方式
  - 轉接至其它服務(例如Kafka)
  - 轉存至儲存服務(例如HDFS、S3)
- **Channel**
  - 以Memory、Disk作為暫存媒體

## 使用Flume

- 在Terminal中以**flume-ng**指令啟動flume agent
  - flume-ng位於‹flume安裝目錄›/bin底下
  - 指令格式
    - flume-ng agent **--conf-file** ‹config-file path› **--name** ‹config-name›

## Flume Config 格式

```
# list the sources, sinks and channels for the agent
<Agent>.sources = <Source>
<Agent>.sinks = <Sink>
<Agent>.channels = <Channel1> <Channel2>

# properties for sources
<Agent>.sources.<Source>.<someProperty> = <someValue>
# set channel for source
<Agent>.sources.<Source>.channels = <Channel1> <Channel2> ...

# properties for sinks
<Agent>.sources.<Sink>.<someProperty> = <someValue>
# set channel for sink
<Agent>.sinks.<Sink>.channel = <Channel1>

# properties for channels
<Agent>.channel.<Channel>.<someProperty> = <someValue>
```

Ref：https://flume.apache.org/FlumeUserGuide.html

# Flume Config－Source設定

| 類型 | 描述 |
|---|---|
| avro source | ‣ 建立avro服務，接收其它avro用戶端發送的訊息<br>‣ 一般**在Agent間的傳輸會使用avro配置** |
| exec source | ‣ 執行Unix指令，以stdout為資料來源<br>‣ 指令可以<agent>.<source>.command配置，如tail -f log.txt |
| netcat source | ‣ 監聽指定port，將每一行封裝成一個Event<br>‣ 預設每行最大長度為512 |
| http source | ‣ 支援HTTP的POST或GET作為資料輸入來源 |
| Kafka Source | ‣ 從Apache Kafka的topic讀取訊息 |
| Spooling Directory Source | ‣ 可監控指定的目錄下新增的文件，並將文件中的內容讀取出來作為Flume的輸入 |
| JMS Source | ‣ 從JMS的queue或者topic讀取訊息 |

## Flume Config－Source設定範例

```
#Avro Source設定範例
a1.sources = r1
a1.channels = c1

a1.sources.r1.type = avro
a1.sources.r1.channels = c1
a1.sources.r1.bind = loclahost
a1.sources.r1.port = 5566
```

```
#Exec Source設定範例
a1.sources = r1
a1.channels = c1

a1.sources.r1.type = exec
a1.sources.r1.command = tail -F /var/log/secure
a1.sources.r1.channels = c1
```

# Flume Config－Channel設定

| 類型 | 描述 |
|---|---|
| Memory Channel | 儲存Event在Memory Queue中，若機器毀損可能會造成數據的丟失，但效能最佳。 |
| JDBC Channel | 將Event儲存在資料庫中，可提高容錯性，目前僅支援內建的Derby。 |
| Kafka Channel | 將Event儲存在Kafka Cluster(必須單獨部署)，Kafka提供了高可用和複製性，所以Kafka或者Agent損毀，資料也不會丟失，建置最複雜，但容錯性最高。 |
| File Channel | 將Event資料儲存在local檔案系統裡，效能較差，但亦可透過設置checkpoint目錄確保容錯性。 |

# Flume Config－Channel設定範例

```
#Memory Channel設定範例
a1.channels = c1
a1.channels.c1.type = memory
a1.channels.c1.capacity = 10000
a1.channels.c1.transactionCapacity = 10000
a1.channels.c1.byteCapacityBufferPercentage = 20
a1.channels.c1.byteCapacity = 800000 #bytes
```

```
#Kafka Channel設定範例
a1.channels.channel1.type =
org.apache.flume.channel.kafka.KafkaChannel
a1.channels.channel1.kafka.bootstrap.servers =
kafka-1:9092,kafka-2:9092,kafka-3:9092
a1.channels.channel1.kafka.topic = channel1
a1.channels.channel1.kafka.consumer.group.id = flume-consumer
```

# Flume Config－Sink設定

| 類型 | 描述 |
|---|---|
| HDFS Sink | 將數據寫入到HDFS |
| Hive Sink | 將文本或者JSON數據用分隔符分割，直接變成Hive的表，或者是Partitation |
| Logger Sink | 記錄Event的Info級別日誌，通常用於測試。 |
| Avro Sink | 採用Avro Sink接收到的Event，發送到另外一個Avro Source |
| File Roll Sink | 將Event存放到本地檔案系統，可依據時間或者大小來作Rolling。 |
| Null Sink | 丟棄所有從Channel獲取的Event。 |
| HBaseSink | 寫入數據到Hbase |
| AsyncHBaseSink | 採用非同步的方式寫入資料到Hbase |
| ElasticSearchSink | 將數據寫入ElasticSearch Cluster |
| Kafka Sink | 將數據寫入Kafka Cluster |

# Flume Config－Sink設定範例

```
#HDSF Sink設定範例
a1.channels = c1
a1.sinks = k1

a1.sinks.k1.type = hdfs
a1.sinks.k1.channel = c1
a1.sinks.k1.hdfs.path = /flume/events/%y-%m-%d/%H%M/%S
a1.sinks.k1.hdfs.filePrefix = events-
```

```
#Avro Sink設定範例
a1.channels = c1
a1.sinks = k1
a1.sinks.k1.type = avro
a1.sinks.k1.channel = c1
a1.sinks.k1.hostname = localhost
a1.sinks.k1.port = 5566
```