IT4931
Lưu trữ và phân tích dữ liệu lớn

IT4931

11/2021

Thanh-Chung Dao Ph.D.

Agenda

Zeppelin notebook

What and why we need it?

Installation using Docker

Usage

Load, inspect, and save data

Loading data from difference sources

Simple inspecting commands

Saving data

Zeppelin notebook

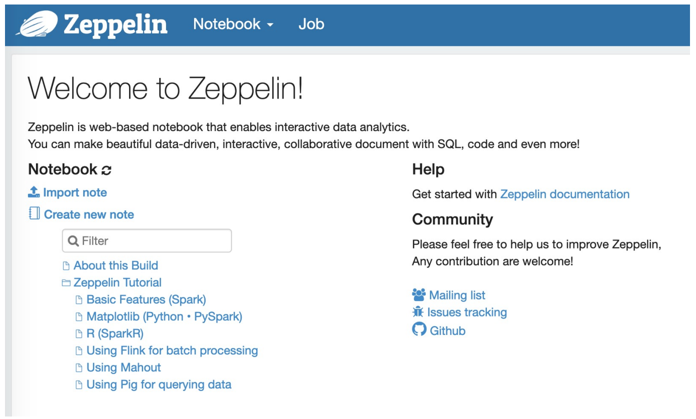
- A web-based interface for interactive data analytics
  - Easy to write and access your code
  - Support many programming languages
    - Scala (with Apache Spark), Python (with Apache Spark), SparkSQL, Hive, Markdown, Angular, and Shell
  - Data visualization
- Monitoring Spark jobs

Installation using Docker

- Install Docker and login
  - https://docs.docker.com/docker-for-windows/install/
  - https://docs.docker.com/docker-for-mac/install/
- Download lecture's git repository
  - https://github.com/bk-blockchain/big-data-class
- Run Zeppelin using docker-composer
  - docker-compose up -d --build spark_master
  - http://localhost

Zeppelin usage

- Run the first node: “About this Build”
  - Check Spark version



Zeppelin Notebook Job

Welcome to Zeppelin!

Zeppelin is web-based notebook that enables interactive data analytics.
You can make beautiful data-driven, interactive, collaborative document with SQL, code and even more!

Notebook

Import note
Create new note

Filter

About this Build
Zeppelin Tutorial
Basic Features (Spark)
Matplotlib (Python • PySpark)
R (SparkR)
Using Flink for batch processing
Using Mahout
Using Pig for querying data

Help

Get started with Zeppelin documentation

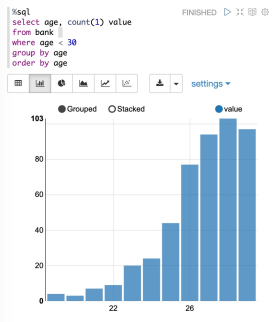
Community

Please feel free to help us to improve Zeppelin,
Any contribution are welcome!

Mailing list
Issues tracking
Github

- Check Spark running mode
  - http://localhost:4040
  - Need to start Spark first by running the first note

- Run the second node: “Tutorial/Basic Features (Spark)”
  - Load data into table
  - SQL example



```
%sql
select age, count(1) value
from bank
where age < 30
group by age
order by age
```

Useful Docker commands

- Login to a container
    - docker ps (get any container id)
    - docker exec -it container_id bash
- List all containers: docker ps -a
- Stop a container: docker stop container_id
- Start a stopped container: docker start container_id

Load, inspect, and save data

- Data is always huge that does not fit on a single machine
  - Data is distributed on many storage nodes
- Data scientists can likely focus on the format that their data is already in
  - Engineers may wish to explore more output formats
- Spark supports a wide range of input and output sources

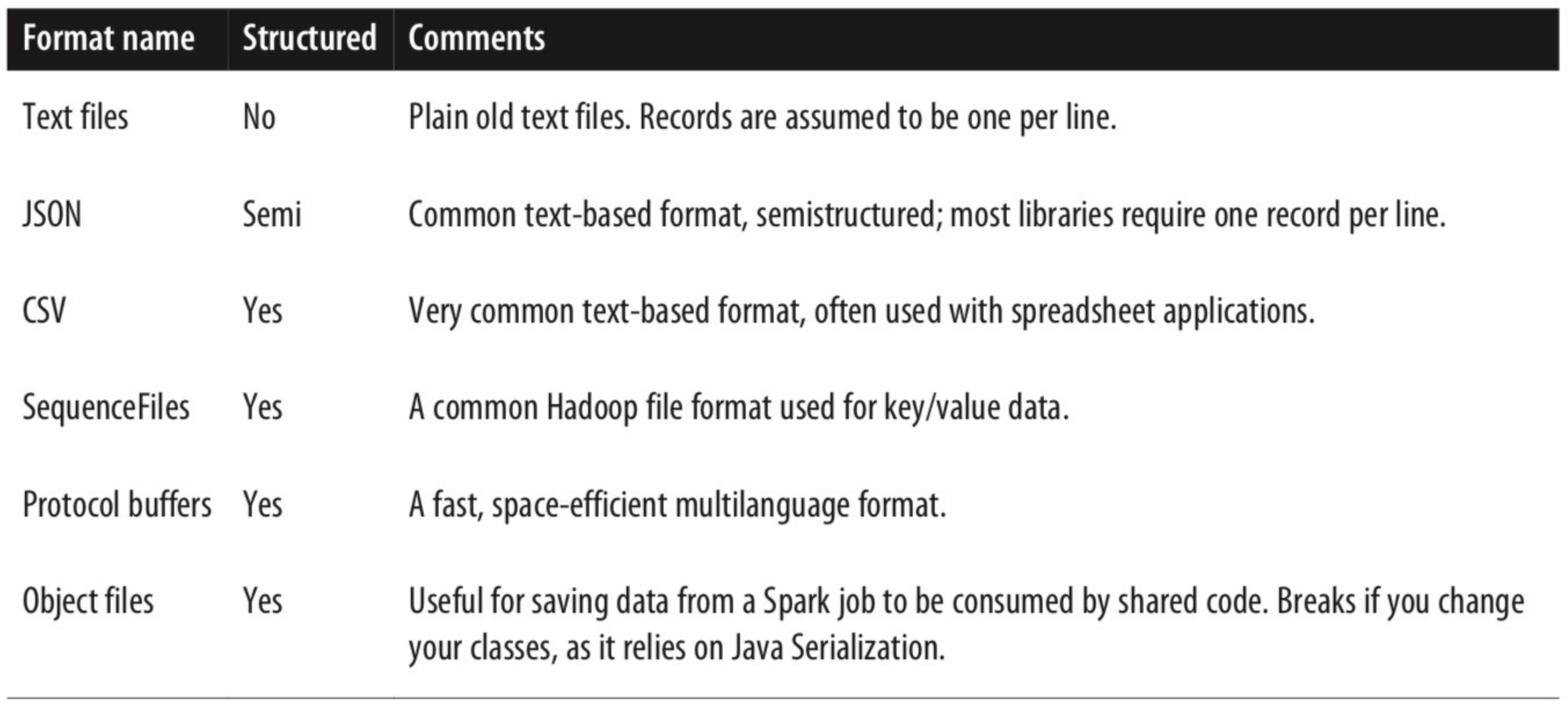
Data sources

- File formats and filesystems
  - Local or distributed filesystem, such as NFS, HDFS, or Amazon S3
  - File formats including text, JSON, SequenceFiles, and protocol buffers
- Structured data sources through Spark SQL
  - Apache Hive
  - Parquet
  - JSON
  - From RDDs
- Databases and key/value stores
  - Cassandra, HBase, Elasticsearch, and JDBC dbs

File Formats

- Formats range from unstructured, like text, to semistructured, like JSON, to structured, like SequenceFiles

Table 5-1. Common supported file formats



Format name	Structured	Comments
Text files	No	Plain old text files. Records are assumed to be one per line.
JSON	Semi	Common text-based format, semistructured; most libraries require one record per line.
CSV	Yes	Very common text-based format, often used with spreadsheet applications.
SequenceFiles	Yes	A common Hadoop file format used for key/value data.
Protocol buffers	Yes	A fast, space-efficient multilanguage format.
Object files	Yes	Useful for saving data from a Spark job to be consumed by shared code. Breaks if you change your classes, as it relies on Java Serialization.

From Learning Spark [1]

Lab: loading, inspecting, and saving data

- On the Zeppelin notebook
    - http://localhost:8080/#/notebook/2EAMFFAH7

References

- [1] Karau, Holden, et al. *Learning spark: lightning-fast big data analysis*. " O'Reilly Media, Inc.", 2015.