Spark + Hive 🦵🐝

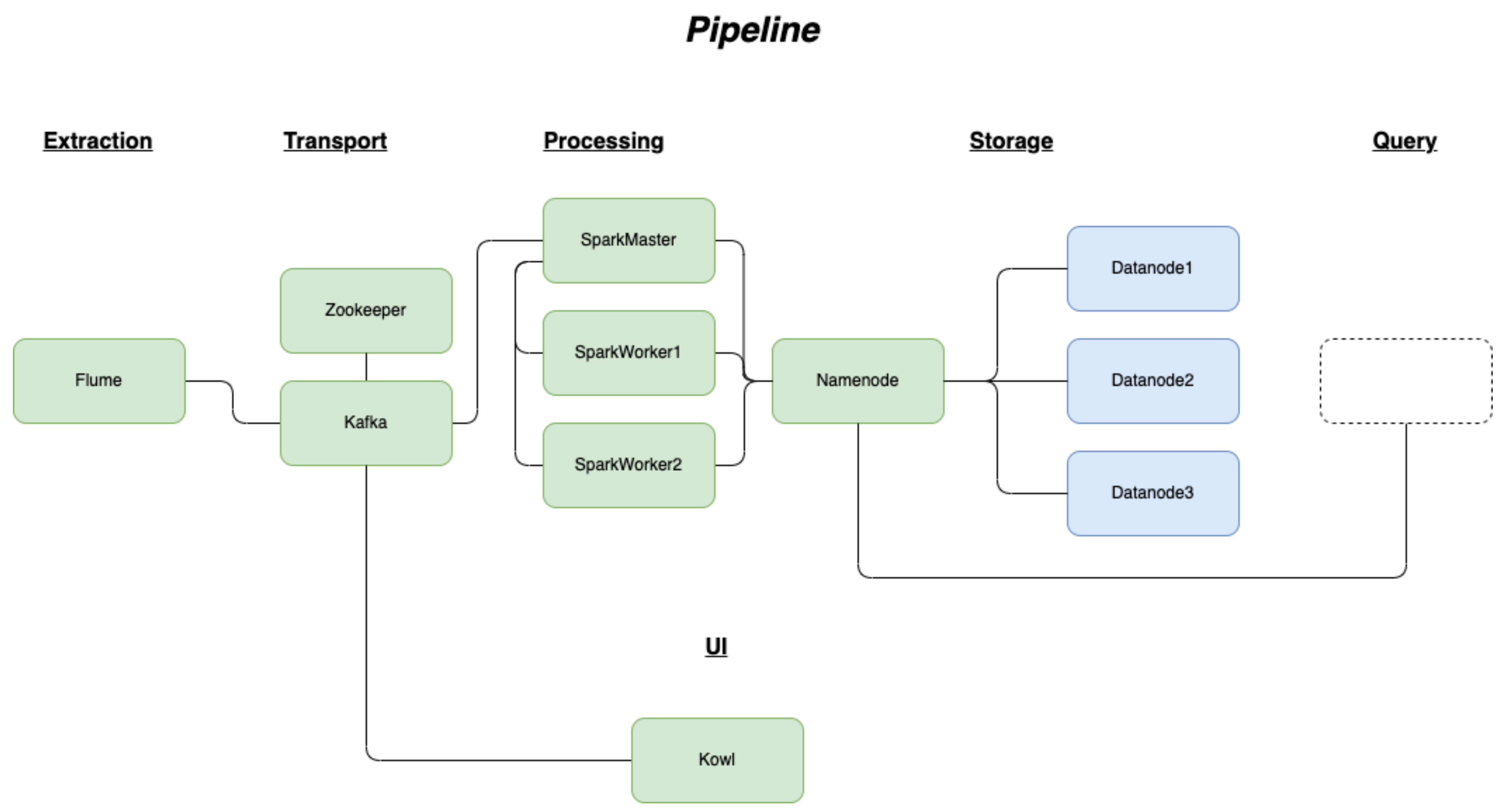
Big Data E22

Nikolai Emil Damm - nidam16@student.sdu.dk
Bende Siewertsen - besie18@student.sdu.dk

Context

What have we been doing?

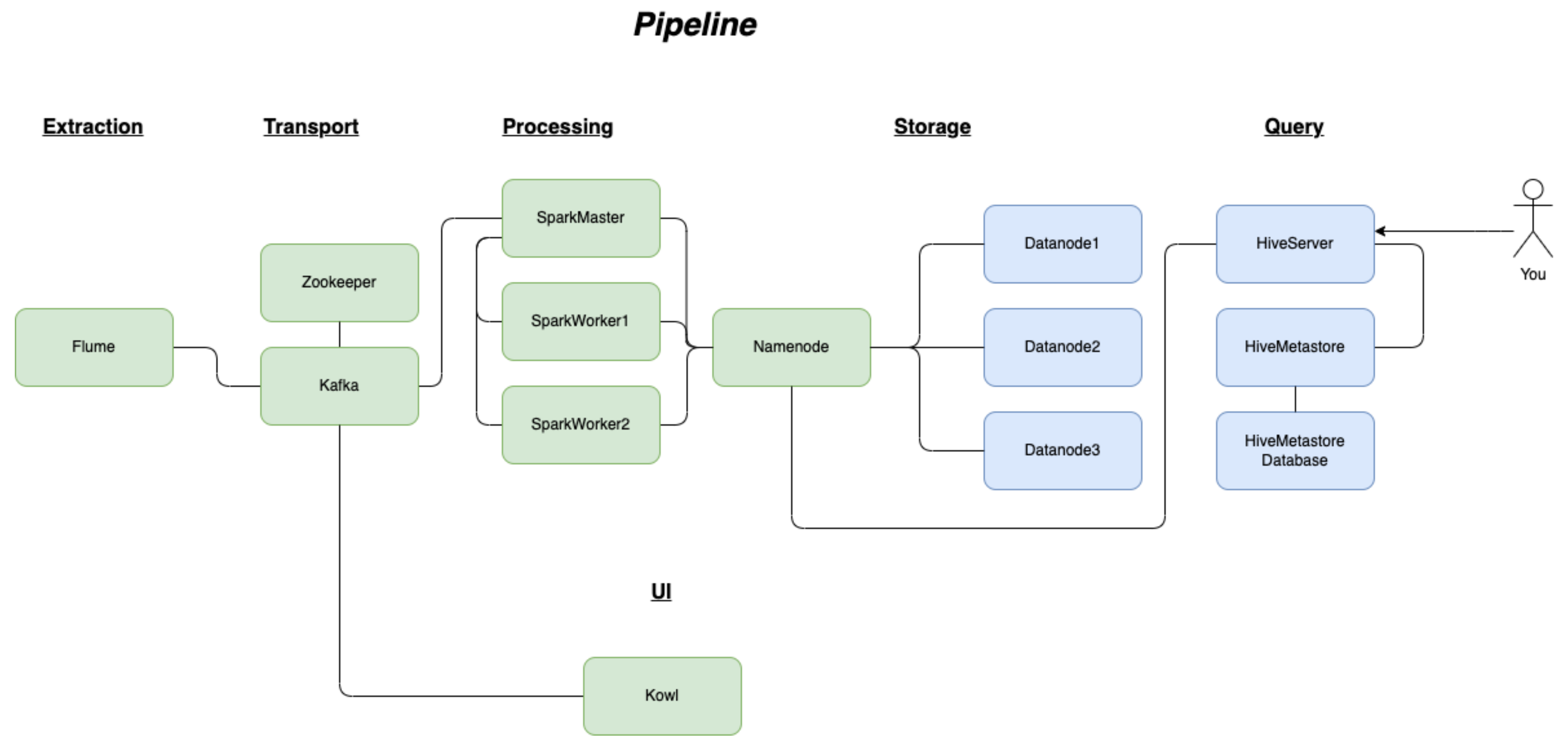
- A data-pipeline to ingest, process and store data.
- Last time we introduced distributed processing.



Context

What are we doing today?

- Revisit Spark Streaming
- Adding hive to our existing cluster
    - Query the wonderland!



Exercise 1

Composing a cluster with Hive

- To work with Hive we need to setup a cluster with a Hive server and a Hiver metastore.

- To do this you need to do the following.

  1. Teardown previous clusters e.g. Hadoop and Kafka. You can use docker system prune after stopping the stacks.

  2. cd ./lecture05-exercises/

  3. Examine the docker-compose.yml file

  4. Run docker compose up -d. Remove the argument -d if you need to see what happens.

Exercise 2

Upload Alice in Wonderland to HDFS (if you haven’t already)

- docker exec –ti namenode bash
- apt update
- apt install wget
- wget -O alice-in-wonderland.txt https://www.gutenberg.org/files/11/11-0.txt
- hdfs dfs -mkdir /txt
- hdfs dfs -put alice-in-wonderland.txt /txt/

Exercise 3

Count words in Alice in Wonderland with Hive!

- Exec into the hive server (beeline is the name of the Hive CLI), and connect to the CLI to the server:

  1. `docker exec -ti hive-server beeline`
  2. `!connect jdbc:hive2://localhost:10000`
  3. username and password is "hive"

- Now lets show existing tables, create a new table, and load some data into it!

  4. `SHOW TABLES;`
  5. `CREATE TABLE lines (line STRING);`
  6. `LOAD DATA INPATH 'hdfs://namenode:9000/txt/alice-in-wonderland.txt' OVERWRITE INTO TABLE lines;`

- Awesome! Now we want structure our new data, so lets create another table for word counts.

  7.
     ```
     CREATE TABLE word_counts
         AS SELECT word, count(1)
         AS count FROM (SELECT explode(split(line, ' ')) AS word FROM lines) w
         GROUP BY word
         ORDER BY word;
     ```

- Finally we should query the word counts and see how they look!

  8. `SELECT * FROM word_counts ORDER BY count DESC LIMIT 10;`

Exercise 4

Check out HDFS

- Exec into the namenode.
- List files in the folder you put the alice-in-wonderland.txt in.
- Is the file there? If yes why? If no why not?
  - Hive is hungry beast 😉 What are the benefits and disadvantages of the behaviour you discovered?

Exercise 5 Part 1

Hive external tables

- Why should we use Hive external tables?
  - Loose coupling with the data.
  - Data can be managed by more that Hive.
  - To avoid that dropping tables in Hive deletes data.
- Lets start out by cleaning up our Hive tables!
  - DROP TABLE word_counts, lines
  - SHOW TABLES; - verify the tables are gone 👀
- Upload alice-in-wonderland.txt to HDFS again. You can follow Exercise 1 if in doubt on how to do that.

https://docs.cloudera.com/HDPDocuments/DLM1/DLM-1.5.1/administration/content/dlm_managed_tables_external_tables.html

Exercise 5 Part 2

Hive external tables

- Lets create an external table!
  - CREATE EXTERNAL TABLE lines (line string) LOCATION 'hdfs://namenode:9000/txt';
- Now lets recreate the word_counts table
  - See previous slide.
  - Verify that it is recreated with SELECT * FROM word_counts ORDER BY count DESC LIMIT 10;
- Now lets add another book into HDFS and query from them both!
  - hdfs dfs -put alice-in-wonderland.txt /txt/alice-in-wonderland2.txt on the namenode
  - SELECT COUNT(*) FROM lines;
  - SELECT INPUT__FILE__NAME FROM lines GROUP BY INPUT__FILE__NAME;
- What happened? What results did you see?

https://docs.cloudera.com/HDPDocuments/DLM1/DLM-1.5.1/administration/content/dlm_managed_tables_external_tables.html

Exercise 6

Sentiment exercise with Hive!

- Next we want to query sentiment scores with Hive. To do this we should first add the sentiment files to HDFS. The files are located in lecture05-exercises/sentiment-files, and should be added into a HDFS folder as usual. (See Exercise 2 if in doubt)
- Now we need to load these files into Hive:
  - Use the LOAD DATA query as in exercise 3.
  - The files should be loaded into two tables - positive_words_raw and negative_words_raw.
- Now we need to structure the data so it is easier to work with:
  - For both positive and negative words, you must now manipulate the data into two new tables consisting of a single column with a unique word in each row.
  - The tables should be called positive_words and negative_words.
- Finally we want to do a query where we join these tables with the word_counts.
  - Try different queries, and see if you can figure out how to join the positive and negative words with the word counts.