

# MapReduce, Hive – opis projektu

## Ogólny opis projektu

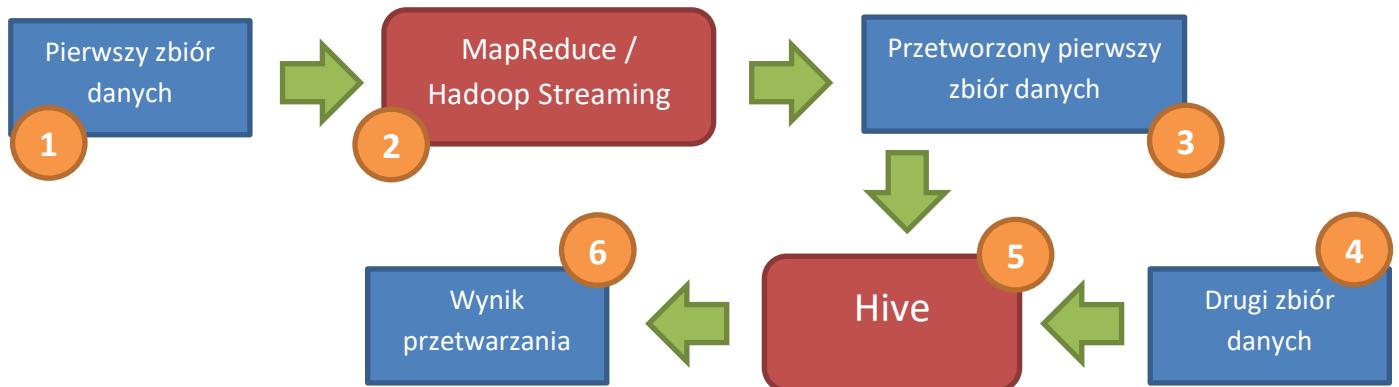
Celem projektu jest praktyczne wykorzystanie podstawowych platform przetwarzania danych stosowanych w środowiskach Big Data.

W ramach każdego z projektów będziemy przetwarzali dwa powiązane ze sobą zbiory danych.

Projekt będzie składał się z dwóch części.

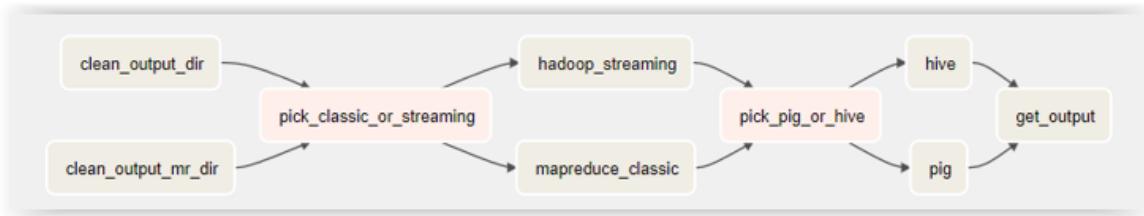
- W pierwszej części, za pomocą przetwarzania *MapReduce* w wariantie klasycznym (Java) lub *Hadoop Streaming*, będziemy przetwarzali pierwszy ze zbiorów danych.
- W drugiej części, za pomocą platformy *Hive*, będziemy przetwarzali wynik z pierwszej części oraz drugi ze zbiorów danych dokonując połączenia tych danych, dalszego przetwarzania.

Graficznie można projekt przedstawić następująco:



Technicznie projekt będzie składał się z:

1. Programu *MapReduce* w wariantie klasycznym (Java) lub *Hadoop Streaming* (2), który działając na pierwszym zbiorze danych (1) będzie generował wynik (3) umieszczając go w systemie plików HDFS.
2. Skryptu *Hive* (5), które działając na wyniku programu *MapReduce* (3) oraz drugim zbiorze danych (4), będzie generował ostateczny wynik przetwarzania (6) umieszczając go w systemie plików HDFS w formacie JSON (na poziomie każdego wynikowego rekordu).
3. Przepływu *Apache Airflow* uruchamianego z poziomu jego interfejsu sieciowego, który będzie:
  - a. przygotowywał system plików HDFS usuwając katalogi wynikowe z poprzednich uruchomień
  - b. uruchamiał program *MapReduce* (2)
  - c. uruchamiał program *Hive* (5)
  - d. pobierał gotowy wynik przetwarzania (6) do lokalnego systemu plików i prezentował jego zawartość



## Kilka wskazówek

Jeśli korzystasz z płatnych platform (np. GCP). Postaraj się tworzyć Twoje rozwiązania lokalnie. Oszczędzaj środki, które masz do dyspozycji. Jedynie ostateczne rozwiązania testuj na korzystając z docelowych płatnych platform.

Nie uruchamiaj początkowych wersji programów na pełnych zbiorach danych. Postaraj się sprawdzić swoje rozwiązania na próbce danych, dopiero kiedy Twój program będzie gotowy, przetestuj go na pełnym wolumenie danych.

Jeśli korzystasz z tymczasowych klastrów *Hadoop* (np. w GCP), nie ładuj źródłowych danych na klaster. Załaduj je na dostarczane przez daną platformę trwałe repozytorium (np. na zasobnik - *bucket*) i jeśli to możliwe przetwarzaj je bezpośrednio z tego repozytorium. Alternatywnie kopij je na klaster (`hadoop fs -copyToLocal gs://`) przed uruchomieniem przetwarzania.

Niniejszy dokument opisuje kwestie dotyczące programu *MapReduce* oraz skryptu *Hive*. Sposób przygotowania trzeciego składnika projektu - przepływu *Apache Airflow* – opisuje oddzielnny dokument.

W przypadku wątpliwości odnośnie interpretacji danych zgłaszaj je i sprawdzaj na forum. Ustalenia, które tam będą miały miejsce są obowiązujące i mogą mieć wpływ na uznanie wyniku/przetwarzania za poprawny/poprawne. Dużo kwestii zostało także rozwiązane w opisach poszczególnych zestawów zadań, które znajdziesz poniżej w sekcji Zestawy danych.

Hierarchia ważności ustaleń: ten dokument, forum, ustalenia z prowadzącym, inne.

## Ogólne wymogi

### Program *MapReduce*

- Program ma być parametryzowany
  - `input_dir1` – katalogiem danych źródłowych (1)
  - `output_dir3` – katalogiem danych wynikowych (3)
- Format plików wynikowych to `TextOutputFormat`. Separatory użyte w przypadku wynikowych kluczy oraz wartości złożonych (wielowartościowych) mogą być dowolne.
- W wyniku implementacji powinny powstać:
  - **program *MapReduce***
    - w przypadku *MapReduce Classic*
      - plik jar zawierający program *MapReduce*
      - pliki źródłowe zaimplementowanych klas tworzących program *MapReduce*
    - w przypadku *Hadoop Streaming*
      - plik zawierający skrypt mappera np. `mapper.py`
      - plik zawierający skrypt reduktora np. `reducer.py`
      - plik zawierający skrypt aggregatatora łączącego (*combinera*) np. `combiner.py`  
(o ile nie został wykorzystany jeden z powyższych)
  - **skrypt uruchamiający `run_mr.sh`** zawierający polecenie uruchamiające program *MapReduce*,
    - ma być uruchamiany za pomocą dwóch parametrów wskazanych powyżej

```
run_mr.sh input_dir1 output_dir3
```

- ma zawierać polecenia, które umożliwią jego wielokrotne i powtarzalne wywoływanie, przykładowo powinien usuwać katalog z danymi wynikowymi (o ile istnieje) przed wywołaniem zadania *MapReduce*

## Program dla Apache Hive

- Program ma być parametryzowany
  - `input_dir3` – katalogiem wejściowym dla przetworzonego pierwszego zbioru danych (wynik przetwarzania *MapReduce*) (3)
  - `input_dir4` – katalogiem wejściowym dla drugiego zbioru danych (4)
  - `output_dir6` – katalogiem wyjściowym, który będzie zawierał ostateczny wynik całości przetwarzania (6)
- Format plików wynikowych to pliki tekstowe z danymi w formacie JSON (na poziomie każdego wynikowego rekordu)
- W wyniku implementacji powinny powstać:
  - skrypt zawierający komplet poleceń **HQL** przeznaczonych do wykonania przez *Apache Hive* np. `hive.hql`
    - tworząc struktury danych i inne definicje należy wykorzystywać bazę danych `default`
    - po zakończonym wywołaniu mają pozostać definicje tabel pozwalające na sięgnięcie do danych źródłowych
    - tabela dla danych źródłowych będących wynikiem działania programu *MapReduce* ma mieć atrybuty o nazwach zgodnych z określonymi w treści projektu dla zadania *MapReduce*
    - skrypt powinien być powtarzalny, tak aby można było go wielokrotnie uruchamiać, przykładowo zanim zostanie utworzona jakaś trwała struktura lub definicja, jej uprzednia powinna zostać usunięta jeśli istnieje
  - skrypt uruchamiający `run_hive.sh` zawierający polecenie uruchamiające powyższy skrypt obsługiwany przez **Apache Hive**, uruchamiany za pomocą trzech parametrów wskazanych powyżej

```
run_hive.sh input_dir3 input_dir4 output_dir6
```

## Punktacja i terminy projektu

Punktacja poszczególnych składowych projektu jest opisana na stronach kursu.

Terminy oddawania projektu wynikają z aktywności obsługującej projekt. Oddajemy projekty i oceniamy je w sposób wynikający z dostępnych w tym celu aktywności Moodle.

## Zestawy danych

Wszystkie zestawy danych pobieramy ze strony wskazanej na stronie kursu niezależnie od ich oryginalnego źródła pochodzenia.

Pobieranie danych źródłowych, rozpakowanie plików i ładowanie ich do miejsca źródłowego nie należy do projektu – nie należy uzupełniać przepływu *Apache Airflow* o dodatkowe operatory, lub polecenia, które wykonują powyższe operacje. Operacje te należy wykonać wcześniej. W założeniu projektu dane źródłowe mają znajdować się w katalogach wskazywanych za pomocą parametrów przepływu *Apache Airflow*. Nie mogą być one na stałe określone w ramach projektu.

## Opis zestawów danych

Opisy poszczególnych zbiorów danych dostępne są w oddzielnych dokumentach.