

Министерство образования и науки Российской Федерации

Калужский филиал
федерального государственного бюджетного образовательного
учреждения высшего образования
**«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»**
(КФ МГТУ им. Н.Э. Баумана)

Ю.С. Белов, С.С. Гришунов

ЯЗЫК PIG LATIN

Методические указания по выполнению лабораторной работы
по курсу «Технологии обработки больших данных»

Калуга - 2018

УДК 004.62
ББК 32.972.5
Б435

Методические указания составлены в соответствии с учебным планом КФ МГТУ им. Н.Э. Баумана по направлению подготовки 09.03.04 «Программная инженерия» кафедры «Программного обеспечения ЭВМ, информационных технологий и прикладной математики».

Методические указания рассмотрены и одобрены:

- Кафедрой «Программного обеспечения ЭВМ, информационных технологий и прикладной математики» (ФН1-КФ) протокол № 6 от «12» января 2018 г.


Зав. кафедрой ФН1-КФ  д.ф.-м.н., профессор Б.М. Логинов

- Методической комиссией факультета ФНК протокол № 1 от «30» 01 2018 г.

Председатель методической комиссии факультета ФНК  к.х.н., доцент К.Л. Анфилов

- Методической комиссией КФ МГТУ им.Н.Э. Баумана протокол № 1 от «06» 02 2018 г.

Председатель методической комиссии КФ МГТУ им.Н.Э. Баумана

 д.э.н., профессор О.Л. Перерва



Рецензент:

к.т.н., зав. кафедрой ЭИУ2-КФ

 И.В. Чухраев

Авторы

к.ф.-м.н., доцент кафедры ФН1-КФ
ассистент кафедры ФН1-КФ

 Ю.С. Белов
 С.С. Гришунов

Аннотация

Методические указания по выполнению лабораторной работы по курсу «Технологии обработки больших данных» содержат краткие основные понятия, краткое описание синтаксиса языка Pig Latin. Рассмотрен пример pig-скрипта для обработки логов web-сервер.

Предназначены для студентов 4-го курса бакалавриата КФ МГТУ им. Н.Э. Баумана, обучающихся по направлению подготовки 09.03.04 «Программная инженерия».

© Калужский филиал МГТУ им. Н.Э. Баумана, 2018 г.
© Ю.С. Белов, С.С. Гришунов, 2018 г.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ	4
ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ.....	5
КРАТКАЯ ХАРАКТЕРИСТИКА ОБЪЕКТА ИЗУЧЕНИЯ, ИССЛЕДОВАНИЯ	6
ПРИМЕР RIG-СКРИПТА.....	10
ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ	15
ТРЕБОВАНИЯ К РЕАЛИЗАЦИИ.....	15
ВАРИАНТЫ ЗАДАНИЙ.....	15
КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ	17
ФОРМА ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ	17
ОСНОВНАЯ ЛИТЕРАТУРА.....	18
ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА	18

ВВЕДЕНИЕ

Настоящие методические указания составлены в соответствии с программой проведения лабораторных работ по курсу «Технологии обработки больших данных» на кафедре «Программное обеспечение ЭВМ, информационные технологии и прикладная математика» факультета фундаментальных наук Калужского филиала МГТУ им. Н.Э. Баумана.

Методические указания, ориентированные на студентов 4-го курса направления подготовки 09.03.04 «Программная инженерия», содержат краткое описание синтаксиса языка Pig Latin, а также примеры решения задач и задание на выполнение лабораторной работы.

Методические указания составлены для ознакомления студентов с языком обработки данных Pig Latin. Для выполнения лабораторной работы студенту необходимы минимальные знания по программированию на высокоуровневом языке программирования (Java, Python или др.), базовые знания SQL.

ЦЕЛЬ И ЗАДАЧИ РАБОТЫ, ТРЕБОВАНИЯ К РЕЗУЛЬТАТАМ ЕЕ ВЫПОЛНЕНИЯ

Целью выполнения лабораторной работы является формирование практических навыков реализации pig-скриптов для обработки больших данных.

Основными задачами выполнения лабораторной работы являются:

1. Получить навыки обработки больших данных, используя Pig Latin.
2. Изучить принцип работы Pig Latin.
3. Изучить синтаксис Pig Latin.
4. Уметь писать запросы, комбинируя несколько источников данных.

Результатами работы являются:

- Входные файлы с данными
- Pig-скрипт работы с данными
- Выходные файлы с результатами вычислений
- Подготовленный отчет

КРАТКАЯ ХАРАКТЕРИСТИКА ОБЪЕКТА ИЗУЧЕНИЯ, ИССЛЕДОВАНИЯ

Создание приложений для MapReduce достаточно трудоемко. Написание всех функций, компилирование и упаковка занимают много времени. Чтобы облегчить работу компания Yahoo! разработала специализированный инструмент под названием Pig, повышающий уровень абстракции при обработке данных.

Pig состоит из двух частей:

1. язык для описания потоков Pig Latin;
2. исполнительная среда для запуска сценариев Pig Latin (доступны два варианта: запуск на локальной JVM или исполнение в кластере Hadoop).

Сценарий Pig включает серию операций (преобразований), которые необходимо применить к входным данным, чтобы получить выходные данные. Эти операции описывают поток данных, который затем преобразуется (компилируется) исполнительной средой Pig в исполняемое представление и запускается для выполнения. Во внутренней реализации Pig трансформирует преобразования в серию заданий MapReduce.

Программа на Pig Latin состоит из выражений, каждое из которых завершается символом «;».

Основной объект в Pig Latin – это «отношение». Именно с отношениями работают все операторы языка. В форме отношений представляются входные и выходные данные.

Каждое отношение представляет собой набор однотипных объектов — «кортежей» (tuples). Аналогии в БД: кортеж — это строка, отношение — это таблица.

Кортежи соответственно состоят из нумерованных или именованных объектов — «полей», произвольных базовых типов (число, строка и т.д.).

Итак, в Pig Latin результатом любого оператора является отношение, представляющее собой набор кортежей.

Обычно программу Pig Latin можно разделить на 3 части:

1. Оператор LOAD, который считывает данные из файловой системы.
2. Операторы преобразования для обработки данных
3. Оператор STORE, записывающий данные в файловую систему, либо оператор DUMP, выводящий результат на экран.

Оператор LOAD создает отношение из файлов в HDFS, например:

```
A = LOAD 'student' USING PigStorage() AS (name:chararray, age:int, gpa:float);
```

Доступ к полям может осуществляться по индексу поля в кортеже или по имени. Для приведенного выше отношения A обращения представлены в табл. 1, типы данных Pig Latin представлены в табл. 2.

Таблица 1

Свойства полей отношения

	Первое поле	Второе поле	Третье поле
Тип данных	chararray	int	float
Обращение по индексу	\$0	\$1	\$2
Обращение по имени	name	age	gpa

Таблица 2

Типы данных Pig Latin

Простые типы данных	Описание	Пример
int	32-битное целое со знаком	10
long	64-битное целое со знаком	10L
float	32-битное число с плавающей точкой	10.5f
double	64-битное число с плавающей точкой	10.5
Массивы		
chararray	Массив символов в формате UTF-8	hello world
bytearray	Массив байт (blob)	
Составные типы данных		
tuple	Упорядоченное множество полей	(19,2)
bag	Коллекция кортежей (tuple)	{(19,2), (18,1)}
map	Множество пар ключ-значение	[key#value]

Операторы отношений

Основной способ трансформации данных в Pig Latin– операторы отношений (табл. 3).

Таблица 3

Операторы отношений		
Оператор	Синтаксис	Описание
SPLIT	SPLIT alias INTO alias IF exp, alias IF exp [, alias IF exp ...];	Разделяет отношение на два и более отношений, основываясь на логических выражениях exp
UNION	alias = UNION alias, alias, [, alias ...];	Создаёт объединение двух и более отношений
FILTER	alias = FILTER alias BY exp;	Отбирает кортежи, основываясь на значении выражения exp
DISTINCT	alias = DISTINCT alias [PARALLEL n];	Удаляет повторяющиеся кортежи
SAMPLE	alias = SAMPLE alias factor;	Создает случайную выборку из кортежей. Размер выборки определяется коэффициентом factor
FOREACH	alias = FOREACH alias GENERATE expression [,expression ...] [AS schema];	Проходит по всем кортежам отношения и генерирует новые кортежи
JOIN	alias = JOIN alias BY field_alias, alias BY field_alias [, alias BY field_alias ...] [USING "replicated"] [PARALLEL n];	Выполняет операцию объединения (inner join).

Таблица 3 (продолжение)

GROUP	alias = GROUP alias { [ALL] [BY {[field_alias [, field_ alias]] * [expression]} } [PARALLEL n];	В рамках одного отношения группирует кортежи с одинаковым ключом группировки. Для группировки всех кортежей в одну группу используется GROUP alias ALL
COGROUP	alias = COGROUP alias BY field_alias [INNER OUTER] , alias BY field_alias [INNER OUTER] [PARALLEL n];	Группирует кортежи из двух или более отношений.
CROSS	alias = CROSS alias, alias[,alias...] [PARALLEL n];	Вычисляет векторное произведение двух и более отношений
ORDER	alias = ORDER alias BY { * [ASC DESC] field_alias [ASC DESC] [, field_alias [ASC DESC] ...]} [PARALLEL n];	Сортирует отношение по одному или нескольким полям
STREAM	alias = STREAM alias [, alias ...] THROUGH {'command' cmd_alias } [AS schema] ;	Обработка отношения внешним скриптом

ПРИМЕР PIG-СКРИПТА

Для примера рассмотрим задачу обработки логов доступа к web-ресурсу. Необходимо определить:

1. общее количество запросов;
2. количество запросов с каждого уникального IP;
3. количество запросов на каждый уникальный URL;
4. объем данных, переданных по каждому URL.

Загрузка данных

Данные представлены в виде текстового файла:

```
08/Dec/2016:15:00:28 178.88.91.180 13600 http 200 4798 0
0 GET /public/cars/bmw7l/down.png HTTP/1.1

08/Dec/2016:15:00:29 193.110.115.45 64318 http 200 1594
0 GET /K1/img/top-nav-bg-default.jpg HTTP/1.1
```

Тогда команда для [загрузки данных](#) будет иметь вид:

```
records = LOAD '/log/flume/events/14-02-20/' USING PigStorage('\t')
AS (
date:chararray,
clientip:chararray,
clientport:chararray,
proto:chararray,
statuscode:int,
bytes:int,
sq:chararray,
bq:chararray,
request:chararray );
```

Оператор LOAD создает отношение records из файлов в HDFS из директории '/log/flume/events/14-02-20/', используя стандартный интерфейс PigStorage (также укажем, что разделителем в файлах является символ табуляции '\t'). Каждая строка из файлов предстанет кортежем в отношении. Секция AS присваивает полям в кортеже типы и имена, по которым нам будет удобнее к ним обращаться.

Далее необходимо провести [обработку данных](#).

Обработка данных

Посчитаем общее количество записей в логах с помощью оператора COUNT. Перед этим необходимо объединить все строки в records в одну группу операторами FOREACH и GROUP:

```
count_total = FOREACH (GROUP records ALL) GENERATE COUNT  
(records);
```

Теперь посчитаем количество запросов с уникальных адресов. В наших кортежах в отношении records в поле clientip содержатся IP-адреса, с которых выполнялись запросы. Сгруппируем кортежи в records по полю clientip и определим новое отношение, состоящее из двух полей:

1. поле ip, значение которого берется из названия группы в отношении records;
2. количество записей в группе — cnt, посчитанное оператором COUNT, то есть количество записей, соответствующих определенному IP-адресу в поле IP.

```
count_ip = FOREACH (GROUP records BY clientip) GENERATE grou  
p AS ip, COUNT(records) AS cnt;
```

Далее определяем еще одно отношение top_ip, состоящее из тех же данных, что и count_ip, но отсортированное по полю cnt оператором ORDER.

```
top_ip = ORDER count_ip BY cnt DESC;
```

После этого посчитаем количество успешных запросов на каждый URL, а также суммарный объем загруженных по каждому URL данных. Для этого сначала воспользуемся оператором фильтрации FILTER, отобрав только успешные запросы с HTTP кодами 200 OK и 206 Partial Content. Этот оператор определяет новое отношение filtered_req из отношения records, отфильтровав его по полю statuscode.

```
filtered_req = FILTER records BY statuscode == 200 OR statuscode ==  
206;
```

Далее аналогично подсчету IP-адресов посчитаем количество уникальных URL, группируя записи в отношении requests по полю request. Для нас также представляет интерес переданный объем данных по каждому URL: его можно рассчитать с помощью оператора SUM, складывающего поля bytes в сгруппированных записях отношения filtered_req.

```
count_req = FOREACH (GROUP filtered_req BY request) GENERATE  
group AS req, COUNT(filtered_req) AS cnt, SUM(filtered_req.bytes) AS  
bytes;
```

Теперь осуществим сортировку по полю bytes, определяя новое отношение top_req:

```
top_req = ORDER count_req BY bytes DESC;
```

Сохранение результатов

Команды для [сохранения результатов](#) имеют вид:

```
%declare DT `date +%y%m%dT%H%M`  
STORE count_total INTO '$DT/count_total';  
STORE top_ip INTO '$DT/top_ip';  
STORE top_req INTO '$DT/top_req';
```

Предпочтительно сохранять результаты каждого выполнения скрипта в отдельную директорию, имя которой включает дату и время исполнения. Для этого можно воспользоваться функцией вызова произвольной шелл-команды прямо из Pig-скрипта (ее нужно написать в обратных кавычках). В примере результат команды date заносится в переменную DT, которая затем подставляется в пути сохранения данных. Сохраняем результаты командой STORE: каждое отношение — в свой каталог.

Еще одной важной особенностью языка Pig Latin является возможность использования в скрипте функций, написанных на другом языке, например, для разбора сложных входных данных с помощью python-скрипта:

```
REGISTER '/path/to/pigudf.py' using streaming_python as myfuncs;  
LOGS = LOAD 'wasb:///example/data/sample.log' as (LINE:chararray);  
LOG = FILTER LOGS by LINE is not null;
```

```
DETAILS = FOREACH LOG GENERATE
myfuncs.create_structure(LINE);
DUMP DETAILS;
```

При этом сама функция имеет вид:

```
@outputSchema("log: {(date:chararray, time:chararray,
classname:chararray, level:chararray, detail:chararray)}")
def create_structure(input):
    if (input.startswith('java.lang.Exception')):
        input = input[21:len(input)] + ' - java.lang.Exception'
    date, time, classname, level, detail = input.split(' ', 4)
    return date, time, classname, level, detail
```

Листинг примера целиком

```
records = LOAD '/log/flume/events/14-02-20/' USING PigStorage('\t')
AS (
date:chararray,
clientip:chararray,
clientport:chararray,
proto:chararray,
statuscode:int,
bytes:int,
sq:chararray,
bq:chararray,
request:chararray );
```

```
count_total = FOREACH (GROUP records ALL) GENERATE COUNT
(records);
```

```
count_ip = FOREACH (GROUP records BY clientip) GENERATE grou
p AS ip, COUNT(records) AS cnt;
top_ip = ORDER count_ip BY cnt DESC;
```

```
filtered_req = FILTER records BY statuscode == 200 OR statuscode ==
206;
```

```
count_req = FOREACH (GROUP filtered_req BY request) GENERATE  
group AS req, COUNT(filtered_req) AS cnt, SUM(filtered_req.bytes) AS  
bytes;
```

```
top_req = ORDER count_req BY bytes DESC;
```

```
%declare DT `date +%y%m%dT%H%M`  
STORE count_total INTO '$DT/count_total';  
STORE top_ip INTO '$DT/top_ip';  
STORE top_req INTO '$DT/top_req';
```

Как видно из примера, язык Pig Latin позволяет быстро и интуитивно понятно оперировать данными, предоставляя программистам вместо возможности написания «вручную» map и reduce функций более высокоуровневый SQL-подобный синтаксис.

ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ

Задание 1

Для всех вариантов:

Выполнить задание из лабораторной работы №2, используя язык Pig Latin

Задание 2

База данных твитов состоит из двух файлов. Выполнить задание по варианту, используя Pig Latin.

Файл tweets.csv имеет формат:

tweet_id, tweet, login

Файл users.csv имеет формат:

login, user_name, state

Файлы: [tweets.csv](#), [users.csv](#)

ТРЕБОВАНИЯ К РЕАЛИЗАЦИИ

Скрипты должны выводить данные в папку с номером задания и текущей датой в названии. Для демонстрации возможно использовать режим работы с локальной файловой системой.

ВАРИАНТЫ ЗАДАНИЙ

1. Найти все твиты, содержащие слово “favorite” и подсчитать их количество. Найти все твиты пользователей из штата NY, сгруппировав их по имени пользователя.
2. Написать программу на Java, используя MapReduce, для выполнения операции join для двух файлов tweets.csv и users.csv. Реализовать операцию join с помощью Pig Latin.
3. Найти все твиты пользователей из штата NY, сгруппировав их по имени пользователя. Отсортировать результат по активности пользователя (пользователи с наибольшим числом твитов должны быть вверху списка). Сохранить два файла:

Первый в формате:

user_name, number of tweets

Второй в формате:

user_name, (tweet, tweet ...)

4. Вывести имена пользователей, опубликовавших хотя бы 2 твита. Отсортировать результат по активности пользователя (пользователи с наибольшим числом твитов должны быть вверху списка).
5. Вывести имена пользователей, неопубликовавших ни одного твита. Подсчитать в каком штате наибольшее число таких пользователей.
6. В каждом штате найти пользователя с наибольшим числом твитов. Подсчитать среднее количество твитов для этих пользователей
7. Выбрать все твиты пользователей из штата NY. Вывести список 20 самых часто используемых слов в их твитах.
8. Найти все твиты о дожде (должны содержать слова rain, rainy). Вывести список штатов и количество твитов о дожде пользователей из каждого штата.
9. Выбрать все твиты пользователей из штата NY и штата MI. Найти список 20 самых часто используемых слов в твитах для обоих штатов. В один файл вывести те слова из списков, которые совпали для обоих штатов, во второй файл вывести несовпавшие слова.
10. Найти всех пользователей, написавших менее 3 твитов. Подсчитать общее количество твитов, написанных этими пользователями. Вычислить долю, которую составляют эти твиты от общего количества твитов в базе.

КОНТРОЛЬНЫЕ ВОПРОСЫ И ЗАДАНИЯ

1. Перечислите основные части, из которых состоит программа, написанная на Pig Latin.
2. Дайте определение отношению.
3. Дайте определение кортежу.
4. Приведите команду для загрузки данные.
5. Приведите команду для обращения к полям отношений.
6. Перечислите основные типы данных Pig Latin.
7. Перечислите основные операции отношений.
8. Приведите команду для сохранения полученных данных в файл.
9. Приведите команду для вывода на экран полученных данных.
10. Приведите метод использования в скрипте функций, написанных на другом языке.

ФОРМА ОТЧЕТА ПО ЛАБОРАТОРНОЙ РАБОТЕ

На выполнение лабораторной работы отводится 2 занятия (4 академических часа: 3 часа на выполнение и сдачу лабораторной работы и 1 час на подготовку отчета).

Номер варианта студенту выдается преподавателем.

Отчет на защиту предоставляется в печатном виде.

Структура отчета (на отдельном листе(-ах)): титульный лист, формулировка задания (вариант), этапы выполнения работы (со скриншотами), результаты выполнения работы. выводы.

ОСНОВНАЯ ЛИТЕРАТУРА

1. Федин Ф.О. Анализ данных. Часть 1. Подготовка данных к анализу [Электронный ресурс] : учебное пособие / Ф.О. Федин, Ф.Ф. Федин. — Электрон. текстовые данные. — М. : Московский городской педагогический университет, 2012. — 204 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/26444.html>
2. Федин Ф.О. Анализ данных. Часть 2. Инструменты Data Mining [Электронный ресурс] : учебное пособие / Ф.О. Федин, Ф.Ф. Федин. — Электрон. текстовые данные. — М. : Московский городской педагогический университет, 2012. — 308 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/26445.html>
3. Чубукова, И.А. Data Mining [Электронный ресурс] : учеб. пособие — Электрон. дан. — Москва : , 2016. — 470 с. — Режим доступа: <https://e.lanbook.com/book/100582>. — Загл. с экрана.
4. Воронова Л.И. Big Data. Методы и средства анализа [Электронный ресурс] : учебное пособие / Л.И. Воронова, В.И. Воронов. — Электрон. текстовые данные. — М. : Московский технический университет связи и информатики, 2016. — 33 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/61463.html>
5. Юре, Л. Анализ больших наборов данных [Электронный ресурс] / Л. Юре, Р. Ананд, Д.У. Джефффри. — Электрон. дан. — Москва : ДМК Пресс, 2016. — 498 с. — Режим доступа: <https://e.lanbook.com/book/93571>. — Загл. с экрана.

ДОПОЛНИТЕЛЬНАЯ ЛИТЕРАТУРА

6. Волкова Т.В. Разработка систем распределенной обработки данных [Электронный ресурс] : учебно-методическое пособие / Т.В. Волкова, Л.Ф. Насейкина. — Электрон. текстовые данные. — Оренбург: Оренбургский государственный университет, ЭБС АСВ, 2012. — 330 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/30127.html>
7. Кухаренко Б.Г. Интеллектуальные системы и технологии [Электронный ресурс] : учебное пособие / Б.Г. Кухаренко. —

- Электрон. текстовые данные. — М. : Московская государственная академия водного транспорта, 2015. — 116 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/47933.html>
8. Воронова Л.И. Интеллектуальные базы данных [Электронный ресурс] : учебное пособие / Л.И. Воронова. — Электрон. текстовые данные. — М. : Московский технический университет связи и информатики, 2013. — 35 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/63324.html>
9. Николаев Е.И. Базы данных в высокопроизводительных информационных системах [Электронный ресурс] : учебное пособие / Е.И. Николаев. — Электрон. текстовые данные. — Ставрополь: Северо-Кавказский федеральный университет, 2016. — 163 с. — 2227-8397. — Режим доступа: <http://www.iprbookshop.ru/69375.html>

Электронные ресурсы:

10. <http://hadoop.apache.org/> (англ.)
11. <https://pig.apache.org/> (англ.)