Pig/Hive/Jaql Fundamentals: Demo

Семинар курса «Управление разно-структурированными большими данными»

http://synthesis.ipi.ac.ru/synthesis/student/BigData/seminar-hadoop/hadoop2014

alexey.vovchenko@gmail.com

Начало работы

Для доступа к серверу IBM BigInsights необходимо выполнить инструкции из: «server_access.pdf». Если у Вас нет этого файла, необходимо написать мне на почту.

Pig

1. Перейдем в каталог с входными данными

```
cd /home/biadmin/labsData/hadoop_lab/pighivejaql/
```

2. Просмотрим содержимое

```
ls
```

3. Должны увидеть список файлов

```
googlebooks-1988.csv  googlebooks-1988.del
```

4. Такой же список файлов уже находится в Hadoop (в прошлой лабораторной данные записывались в Hadoop). Посмотреть содержимое в Hadoop можно командой

```
hadoop fs -ls vovchenko/hadoop_lab/pighivejaql
```

5. Как мы видим результат идентичен

```
-rw-r--r--   1 biadmin biadmin    9465807 2014-10-14 03:10
vovchenko/hadoop_lab/pighivejaql/googlebooks-1988.csv

-rw-r--r--   1 biadmin biadmin    9920381 2014-10-14 03:10
vovchenko/hadoop_lab/pighivejaql/googlebooks-1988.del
```

6. Теперь просмотрим содержимое файла

```
head -5 googlebooks-1988.csv
```

7. Запускаем Pig

```
cd $PIG_HOME/bin

./pig
```

8. Для вычисления абсолютных значений чисел используем внешнюю библиотеку

```
REGISTER /opt/ibm/biginsights/pig/contrib/piggybank/java/piggybank.jar;
```

9. Загружаем данные из файла в переменную Pig

```
records = LOAD 'vovchenko/hadoop_lab/pighivejaql/googlebooks-1988.csv' AS
(word:chararray, year:int, wordcount:int, pagecount:int, bookcount:int);
```

10. Группируем по длине слова

```
grouped = GROUP records by org.apache.pig.piggybank.evaluation.math.IntAbs(wordcount);
```

11. Суммируем

```
final = FOREACH grouped GENERATE group, SUM(records.wordcount);
```

12. Смотрим результат

```
DUMP final;
```

13. Выходим

```
quit
```

Hive

1. Запускаем Hive

```
cd $HIVE_HOME/bin

./hive
```

2. Создаем таблицу

```
CREATE TABLE vovchenko_wordlist (word STRING, year INT, wordcount INT, pagecount INT,
bookcount INT) ROW FORMAT DELIMITED FIELDS TERMINATED BY '\t';
```

3. Загружаем данные

```
LOAD DATA LOCAL INPATH '/home/biadmin/labsData/hadoop_lab/pighivejaql/googlebooks-
1988.csv' OVERWRITE INTO TABLE vovchenko_wordlist;
```

4. Создаем таблицу длинны слов

```
CREATE TABLE vovchenko_wordlengths (wordlength INT, wordcount INT);
```

5. Заполняем ее данными:

```
INSERT OVERWRITE TABLE vovchenko_wordlengths SELECT length(word), wordcount FROM
vovchenko_wordlist;
```

6. Строим гистограмму

```
SELECT wordlength, sum(wordcount) FROM vovchenko_wordlengths group by wordlength;
```

7. Завершаем работу

```
quit;
```

Jaql

Загружаем Jaql Shell

1. Меняем текущую папку

```
cd $BIGINSIGHTS_HOME/jaql/bin
```

2. Запускаем Jaql шел

```
./jaqlshell
```

Основы Jaql

1. Создадим массив значений

```
a1 = [1,2,3,4];
```

2. Просмотрим 3ий элемент массива.

```
a1[3];
```

3. Отобразим первые три элемента массива

```
a1[0:2];
```

4. Попробуем обновить значение 1го элемента массива. Запустим следующую команду.

```
a1[0]=9;

Возникает ошибка.
```

5. Правильный способ обновления значения

```
a1 = replaceElement(a1,0,9);

a1;
```

6. Создадим массив, a2, со значениями из диапазона от 1 to 20.

```
a2 = range(1,20);

a2;
```

7. Построим значения в обратном порядке

```
reverse(a2);
```

8. Посчитаем число элементов в массиве

```
count(a2);
```

Работа с записями

1. Создадим массив из нескольких записей:

```
a3=[{name: 'John', age: 40, children:['Katie','Will']}, {name:'Mary', age:21}];
```

2. Посмотрим, что произойдет, если обратится к атрибуту записи с именем “name”?

```
a3.name;
```

3. Оба имени будут показаны в результате. Можно аналогичный результат получить и другим образом:

```
a3[*].name;
```

4. Если же мы хотим получить имя только первой записи то это можно сделать так:

```
a3[0].name;
```

5. Для вложенных массивов можно также получить доступ:

```
a3[0].children[0];
```

6. Отобразить только имя и возраст у первой записи.

```
a3[0]{.name,.age};
```

7. Добавим новое поле ко второй записи gender и зададим значение ‘F’;

```
{a3[1].*,gender:'F'};
```

8. Удалим поле возраста age из первой записи.

```
a3[0]{*-.age};
```

9. Удалим поле возраста и одновременно добавим новое поле пола со значением ‘M’;

```
{a3[0]{*-.age},gender:'M'};
```

10. Отобразим список атрибутов у первой записи;

```
names(a3[0]);
```

Основные операции

1. Загрузим JSON файл содержащий названия книг, их авторов, год издания и некоторое описание. Прочитаем данные из файла и отобразим их на экране.

```
books=read(file("/home/biadmin/labsData/hadoop_lab/jaql/bookreviews.json"));

books;
```

2. Попробуем считать данные из нескольких файлов. Создадим для этого функцию читающую данные по имени файла.

```
readjson=fn(filename) read(file("/home/biadmin/labsData/hadoop_lab/jaql/" +
filename));
```

3. Проверим работоспособность функции на файле booksreview.json.

```
readjson("bookreviews.json");
```

4. Файл bookreviews содержит книги двух авторов: J. K. Rowlings и David Baldacci. Выведем в результат только книги написанные David Baldacci.

```
books -> filter $.author == 'David Baldacci';
```

5. Отобразим только те книги, которые были изданы до 2001го года.

```
books -> filter $.published < 2001;
```

6. Посчитаем общее число книг каждого из авторов и отобразим для каждого автора число написанных им книг.

```
books-> group by author = $.author into {author, num_books: count($)};
```

7. Отобразим только книги, написанные в 2000 году, и для них отобразим только авторов и названия.

```
books->filter $.published == 2000 ->transform {$.author, $.title};
```

8. У нас также есть и другой файл: /home/biadmin/labsData/hadoop_lab/jaql/books.csv, в котором содержится информация о каждой книге. В каждой записи есть номер книги, автор, название и год издания. Считаем этот файл.

```
booksonly=read(del("file:///home/biadmin/labsData/hadoop_lab/jaql/books.csv"));

booksonly;
```

9. Пусть по какой-то причине (т.к. это лабораторная, то разумность причины нас мало интересует), нам хочется чтобы отдельные массивы в booksonly были помещены в один массив. Это можно сделать следующей командой.

```
booksonly -> expand $;
```

10. По умолчанию данные считываются без схемы. Например, поле «год» было считано как строка. Попробуем прочитать файл явно задав схему.

```
booksonly=read(del("file:///home/biadmin/labsData/hadoop_lab/jaql/books.csv",

schema=schema{booknum:long,author:string,title:string,published:long}));

booksonly;
```

11. Нам также доступен еще один файл reviews.csv в той же папке. Каждая запись содержит номер книги, имя рецензента, и рейтинг (кол-во звезд). Прочитаем этот файл.

```
reviews=read(del("file:///home/biadmin/labsData/hadoop_lab/jaql/reviews.csv",

schema=schema{booknum:long,name:string,stars:long}));
```

```
reviews;
```

12. Соединим массивы booksonly и reviews по номеру книги (booknum), возвращая в результате название книги и ее рейтинг.

```
booksandstars=join b in booksonly, r in reviews where b.booknum == r.booknum into
{b.title, r.stars};

booksandstars;
```

13. Посчитаем теперь среднее кол-во звезд для каждой книги.

```
booksandstars->group by book = $.title into {book, avg_stars: avg($.stars)};
```

14. Последние две команды можно сделать и одним правилом группировки-соединения.

```
group booksonly by g = $.booknum as bs, reviews by g = $.booknum as rs into
{title:bs[0].title, avg_stars: avg(rs.stars)};
```

15. Командой Explain можно посмотреть переписывается ли последнее выражение в MapRedcue или нет

```
explain group booksonly by g = $.booknum as bs, reviews by g = $.booknum as rs into
{title:bs[0].title, avg_stars: avg(rs.stars)};
```

Использование Jaql SQL

1. Сделаем группировку и подсчет среднего рейтинга с использованием Jaql SQL.

```
select b.title, avg(r.stars) as avg_stars from booksonly b, reviews r group by
b.title;
```

2. Посмотрим план выполнения этой команды

```
explain select b.title, avg(r.stars) as avg_stars from booksonly b, reviews r group by
b.title;
```

3. Если сделать explain старых двух операций join и groupby то увидим тоже самое что и в реляционном виде.

```
explain booksandstars -> group by book = $.title into {book, avg_stars: avg($.stars)};
```