

Текстовая расшифровка видео:

## КАК РАБОТАЮТ JOIN'Ы

### План:

- Задача: средняя оценка по предмету;
- MapJoin.

### Задача: средняя оценка по предмету

### Задача:

Есть табель и результат опроса студентов: «Какой предмет самый любимый». Как посчитать для каждого предмета среднюю оценку среди любителей этого предмета?

Нам может помочь **ReduceJoin**.

Делаем для каждого датасета составной ключ с типом значений. Считаем для каждого предмета среднюю оценку для каждого студента. Считаем среднюю оценку по всем студентам.

### Подробнее:

На входе у нас есть два датасета. Если мы можем определить по содержимому файлу, с чем работаем, то мы можем решить все в маппере либо сделать две Map Only-Job, каждая из которых будет читать свои файлы.

На входе у нас есть два датасета. Если по файлу нам известно содержание, то мы можем работать в маппере. Другой вариант – сделать две Map Only-Job, задача которых читать свои файлы.

Рассмотрим на примере:



```

meow-nofer@pikachu ~/Experiments/slurm/hadoop > cd 02-students
meow-nofer@pikachu ~/Experiments/slurm/hadoop/02-students > ls
excellent_reducer.py  join_mapper1.py  scores_subjects_average_reducer2.py  student_subj.txt
gen_fav_subj.py      join_mapper2.py  scores_subjects_average_reducer.py  subjects_mapper.py
gen_marks.py         scores_mapper.py  student_marks.txt
meow-nofer@pikachu ~/Experiments/slurm/hadoop/02-students > less student_marks.txt
meow-nofer@pikachu ~/Experiments/slurm/hadoop/02-students > less student_subj.txt
meow-nofer@pikachu ~/Experiments/slurm/hadoop/02-students >

```

```

Watkins Art
Wilson Physics
Hall Literature
Anderson Biology
Mendoza Biology
Rivera Physics
Wright Physics
Bauer Math
Turner Art
Gonzalez Chemistry
Cole Art
Henry Math
Weaver Art
Price Literature
Hernandez Chemistry
Weber Math
Green Math
Bell Biology
Schaefer Literature
Marks Physics
Knight PE
Ramsey PE
Salinas Physics
Butler Math
student_subj.txt
[0] 0:zsh 1:zsh 2:nvim- 3:zsh*

```

У нас есть файлы с оценками и любимыми предметами. Наша задача – их смэпить.

Когда мы читаем оценки студента в маппере, мы можем писать следующее:

```

" Press ? for help
.. (up a dir)
</Experiments/slurm/hadoop/
  01-wordcount/
    mapper2.py*
    reducer2.py*
  02-students/
    excellent_reducer.py
    gen_fav_subj.py
    gen_marks.py
    join_mapper1.py
    join_mapper2.py
    scores_mapper.py
    scores_subjects_average_
    scores_subjects_average_
    student_marks.txt
    student_subj.txt
    subjects_mapper.py
  notebooks/
    76-0.txt
    cleanup.sh*
</Experiments/slurm/hadoop
NORMAL 02-students/scores_mapper.py python ut
"02-students/scores_mapper.py" 7L, 145B
[0] 0:zsh 1:zsh 2:nvim* 3:zsh-

```

«для такого-то студента»; добавляем колонку с типом данных (указать, что это «score» (оценка)); через tab добавляем саму оценку.

Для любимых предметов действия практически идентичны:

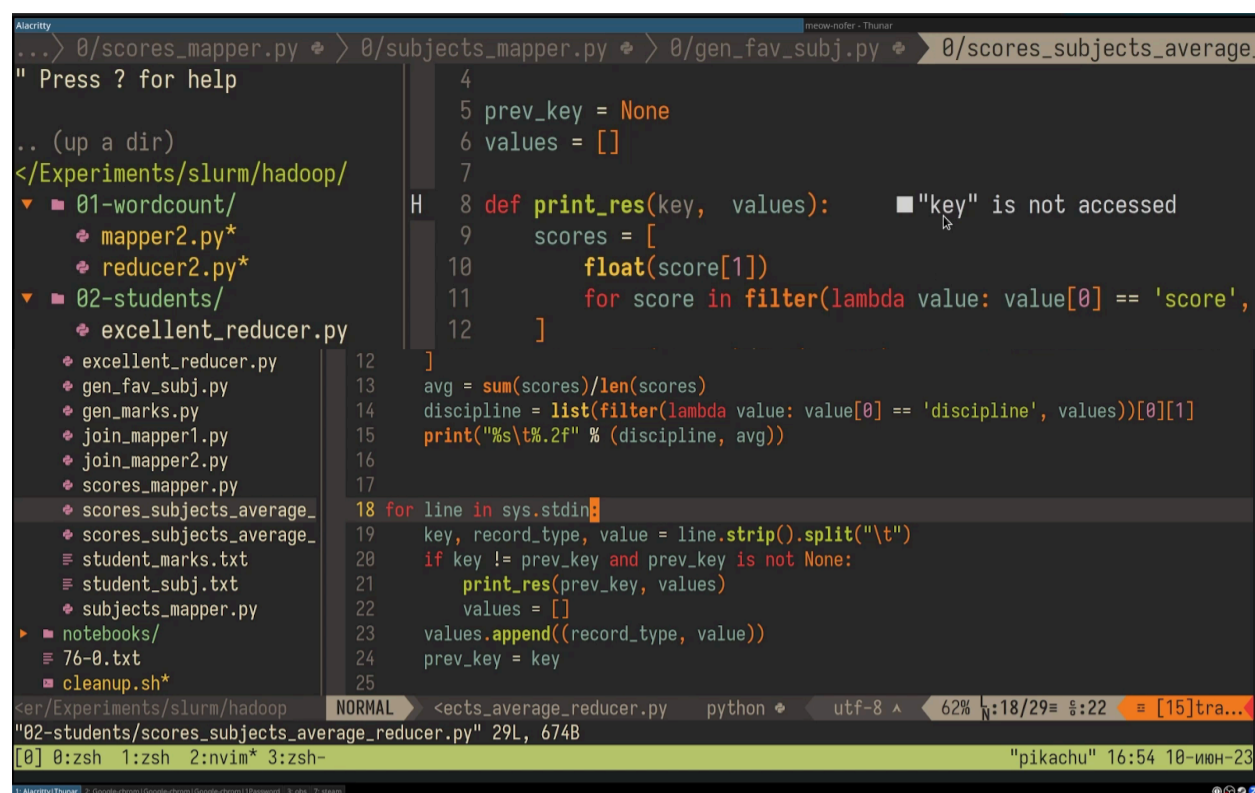
```

" Press ? for help
.. (up a dir)
</Experiments/slurm/hadoop/
  01-wordcount/
    mapper2.py*
    reducer2.py*
  02-students/
    excellent_reducer.py
    gen_fav_subj.py
    gen_marks.py
    join_mapper1.py
    join_mapper2.py
    scores_mapper.py
    scores_subjects_average_
    scores_subjects_average_
    student_marks.txt
    student_subj.txt
    subjects_mapper.py
  notebooks/
    76-0.txt
    cleanup.sh*
</Experiments/slurm/hadoop
NORMAL 02-students/subjects_mapper.py python ut
"02-students/subjects_mapper.py" 7L, 151B
[0] 0:zsh 1:zsh 2:nvim* 3:zsh-

```

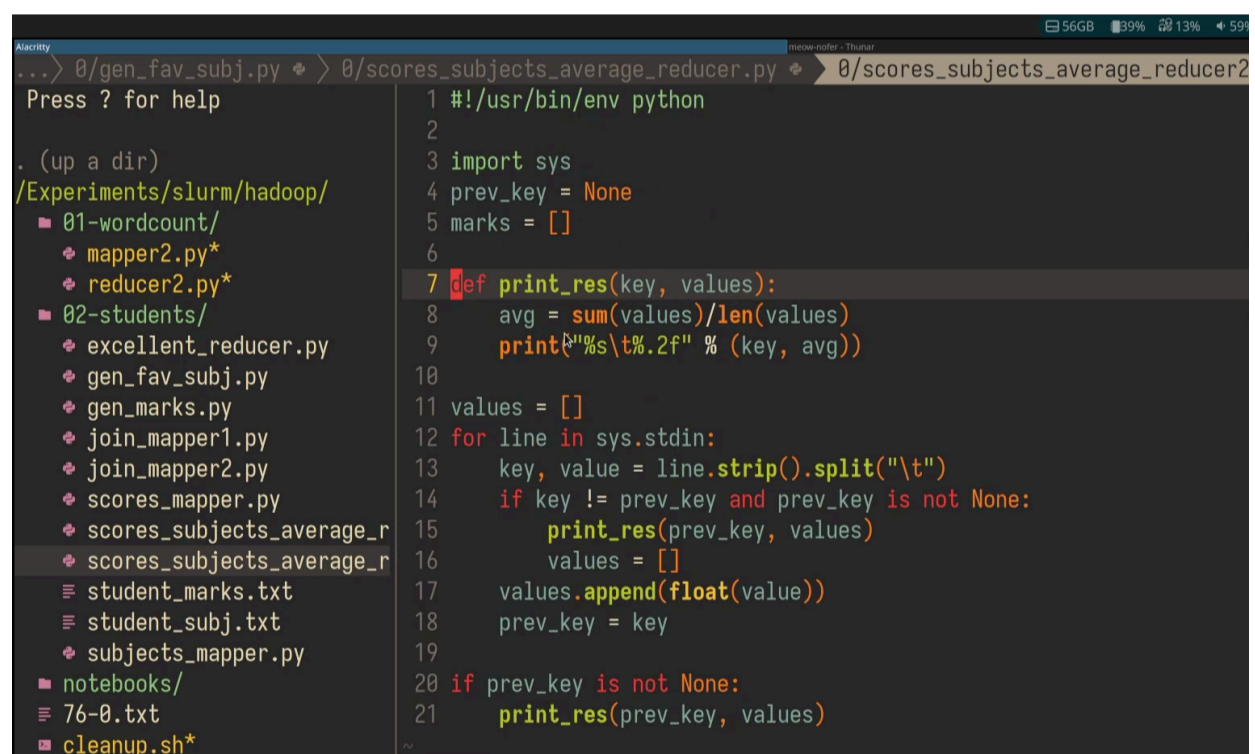
Мы делаем маппер, в котором формат данных такой же, но тип данных будет другой (будет «discipline»). Таким образом, при чтении в job'ах мы будем знать, какие данные из «score», а какие – из «discipline».

На входе все данные будут отсортированы. В reducer считаем среднюю оценку по всем студентам, далее проводим join на стороне reduce:



```
0/scores_mapper.py > 0/subjects_mapper.py > 0/gen_fav_subj.py > 0/scores_subjects_average_
" Press ? for help
.. (up a dir)
</Experiments/slurm/hadoop/
  01-wordcount/
    mapper2.py*
    reducer2.py*
  02-students/
    excellent_reducer.py
    excellent_reducer.py
    gen_fav_subj.py
    gen_marks.py
    join_mapper1.py
    join_mapper2.py
    scores_mapper.py
    scores_subjects_average_
    scores_subjects_average_
    student_marks.txt
    student_subj.txt
    subjects_mapper.py
notebooks/
76-0.txt
cleanup.sh*
4
5 prev_key = None
6 values = []
7
8 def print_res(key, values):
9     scores = [
10         float(score[1])
11         for score in filter(lambda value: value[0] == 'score',
12 ]
13 ]
14 avg = sum(scores)/len(scores)
15 discipline = list(filter(lambda value: value[0] == 'discipline', values))[0][1]
16 print("%s\t%.2f" % (discipline, avg))
17
18 for line in sys.stdin:
19     key, record_type, value = line.strip().split("\t")
20     if key != prev_key and prev_key is not None:
21         print_res(prev_key, values)
22         values = []
23     values.append((record_type, value))
24     prev_key = key
25
```

Мы проходимся по данным, которые поступили нам в редюсере. Если данные «score» и «discipline», то записываем и сохраняем. В данной ситуации мы считаем среднюю оценку (суммируем по score, считаем среднее и возвращаем). Далее, высчитываем среднее по любимому предмету. Мы делаем еще один reduce и в нем делим одно на другое:



```
0/gen_fav_subj.py > 0/scores_subjects_average_reducer.py > 0/scores_subjects_average_reducer2
Press ? for help
. (up a dir)
/Experiments/slurm/hadoop/
  01-wordcount/
    mapper2.py*
    reducer2.py*
  02-students/
    excellent_reducer.py
    gen_fav_subj.py
    gen_marks.py
    join_mapper1.py
    join_mapper2.py
    scores_mapper.py
    scores_subjects_average_r
    scores_subjects_average_r
    student_marks.txt
    student_subj.txt
    subjects_mapper.py
notebooks/
76-0.txt
cleanup.sh*
1 #!/usr/bin/env python
2
3 import sys
4 prev_key = None
5 marks = []
6
7 def print_res(key, values):
8     avg = sum(values)/len(values)
9     print("%s\t%.2f" % (key, avg))
10
11 values = []
12 for line in sys.stdin:
13     key, value = line.strip().split("\t")
14     if key != prev_key and prev_key is not None:
15         print_res(prev_key, values)
16         values = []
17     values.append(float(value))
18     prev_key = key
19
20 if prev_key is not None:
21     print_res(prev_key, values)
```

### Подытожим:

Мы учитываем факт того, что данные сортируются по ключам. Если типы данных разные, то можно их обогатить дополнительной колонкой с этим же типом данных, в reduce сделать дополнительно join.

## MapJoin

Довольно часто датасет, с которым мы джойним, небольшой (например, справочник в несколько гб). Мы можем его целиком залить в память на worker'ах и там джойнить.

Так, например, мы считаем среднюю оценку без усреднения по людям. Мы берем список любимых предметов и заливаем целиком, заджойнив с тем, что лежит в памяти, не стаякая reduce job'ы.

Выглядеть это может следующим образом:

```
meow-nofer@pikachu ~/Experiments/slurm/hadoop
0/mapper2.py > 0/reducer2.py > 0/join_mapper1.py > 0/scores_mapper.py > 0/subjects_mapper.py
" Press ? for help
.. (up a dir)
</Experiments/slurm/hadoop/
  01-wordcount/
    mapper2.py*
    reducer2.py*
  02-students/
    excellent_reducer.py
    gen_fav_subj.py
    gen_marks.py
    join_mapper1.py
    join_mapper2.py
    scores_mapper.py
    scores_subjects_average_
    scores_subjects_average_
    student_marks.txt
    student_subj.txt
    subjects_mapper.py
  notebooks/
    76-0.txt
    cleanup.sh*
</Experiments/slurm/hadoop NORMAL 02-students/join_mapper1.py python utf-8
"02-students/join_mapper1.py" 12 lines --8%--
[0] 0:zsh 1:zsh 2:nvim* 3:zsh-
```

```
1 #!/usr/bin/env python
2
3 import sys
4
5 student_subjects = {}
6 for line in open('student_subj.txt'):
7     student, subject = line.strip().split("\t")
8     student_subjects[student] = subject
9
10 for line in sys.stdin:
11     student, mark = line.strip().split('\t')
12     print(student_subjects[student] + "\t" + mark)
```

Мы читаем файл, сохраняем информацию о том, у кого какой любимый предмет. Далее, читаем данные и автоматом достаём, что хотим.

**Вопрос:** как прочитать данный файл на всех worker'ах, чтобы он был доступен?

**Ответ:** вспомним синтаксис запуска команд:

```
meow-nofer@pikachu ~/Experiments/slurm/hadoop
Yarn jar $HADOOP_STREAMING_JAR \
-input /user/meow-nofer/input \
-output /user/meow-nofer/output \
-mapper "python mapper2.py" \
-reducer "python reducer2.py" \
-file 01-wordcount/mapper2.py \
-file 01-wordcount/reducer2.py
failing bck-i-search: yat_
```

У нас есть ключ (files). Используем distributed cache. Более того, файл с данными, который хотим переслать, мы можем заархивировать, а в мапперах читать его из архива.

Так, мы можем указать через файл список любимых предметов. Он зальётся в distributed cache и будет доступен на каждом воркере локально. Таким образом мы сделаем join на этапе маппера:

```
meow-nofer@pikachu ~/Experiments/slurm/hadoop
0/mapper2.py > 0/reducer2.py > 0/join_mapper1.py > 0/scores_mapper.py > 0/subjects_mapper.py
" Press ? for help
.. (up a dir)
</Experiments/slurm/hadoop/
  01-wordcount/
    mapper2.py*
    reducer2.py*
  02-students/
    excellent_reducer.py
    gen_fav_subj.py
    gen_marks.py
    join_mapper1.py
    join_mapper2.py
    scores_mapper.py
    scores_subjects_average_
    scores_subjects_average_
    student_marks.txt
    student_subj.txt
    subjects_mapper.py
  notebooks/
    76-0.txt
    cleanup.sh*
</Experiments/slurm/hadoop NORMAL 02-students/join_mapper1.py python utf-8
"02-students/join_mapper1.py" 12 lines --8%--
[0] 0:zsh 1:zsh 2:nvim* 3:zsh-
```

```
1 #!/usr/bin/env python
2
3 import sys
4
5 student_subjects = {}
6 for line in open('student_subj.txt'):
7     student, subject = line.strip().split("\t")
8     student_subjects[student] = subject
9
10 for line in sys.stdin:
11     student, mark = line.strip().split('\t')
12     print(student_subjects[student] + "\t" + mark)
```

**Важно:** это работает только тогда, когда у нас датасет, с которым мы джойним достаточно маленький для того, чтобы поместился в памяти целиком.

**Вопрос:** зачем нам об этом знать?

**Ответ:** данная концепция универсальна. Это основополагающие положения, о которых необходимо знать.

Как вам урок?



Изучил, далее >

Слёрм ©

[+7 \(495\) 248-05-80](tel:+7(495)248-05-80)

[Лицензия №ДЛ-1368 от 22.08.2019](#)

[Политика конфиденциальности](#)

[Публичная оферта](#)

