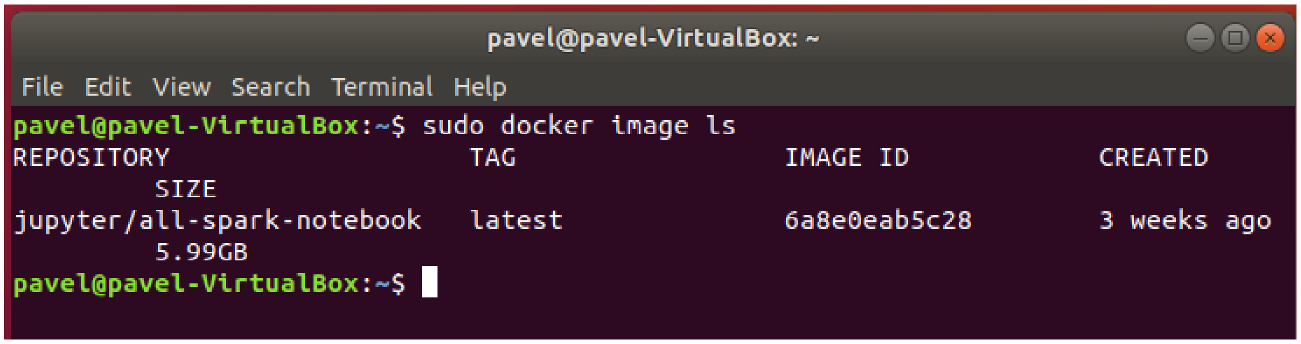
Jupyter homework.

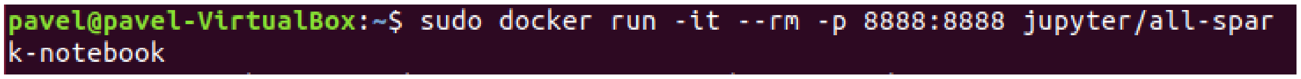
First, I downloaded the all-spark-notebook image, using the docker pull jupyter/all-spark-notebook command.

Here's the list of images that I have:


pavel@pavel-VirtualBox: ~
File Edit View Search Terminal Help

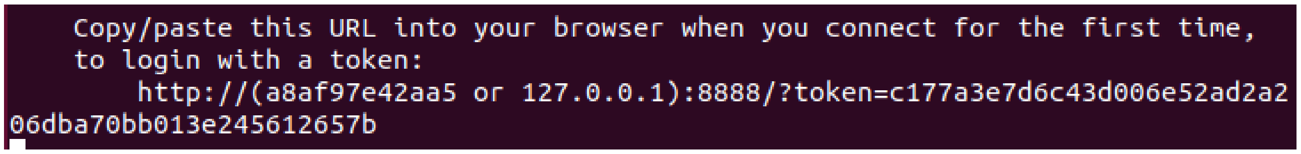
```
pavel@pavel-VirtualBox:~$ sudo docker image ls
REPOSITORY                   TAG                 IMAGE ID            CREATED
         SIZE
jupyter/all-spark-notebook   latest              6a8e0eab5c28        3 weeks ago
         5.99GB
pavel@pavel-VirtualBox:~$ 
```

Then, in order to start a container of this image, I used this command:



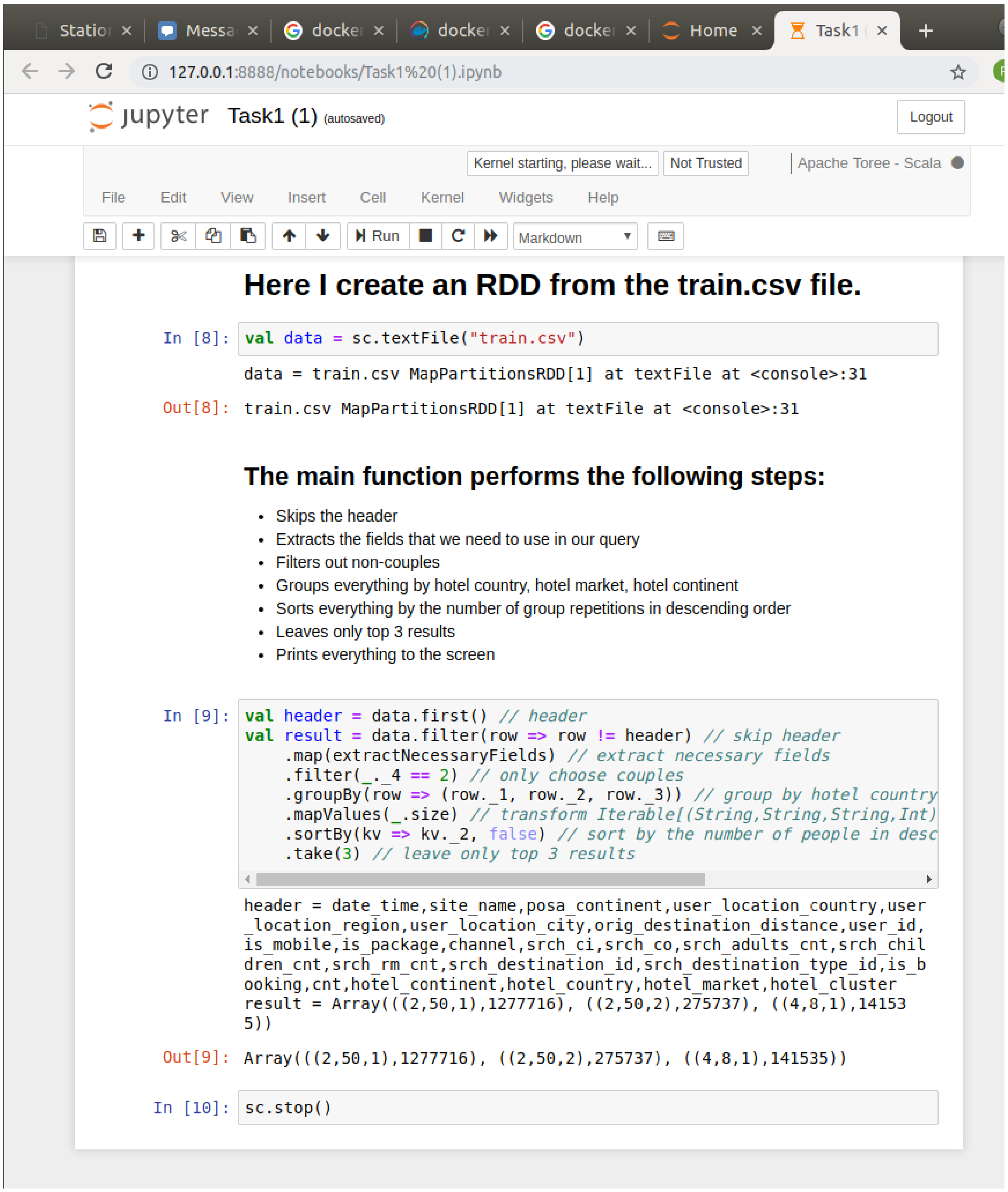
```
pavel@pavel-VirtualBox:~$ sudo docker run -it --rm -p 8888:8888 jupyter/all-spar
k-notebook
```

Which gives you a url you can connect to and use your jupyter notebook:



```
    Copy/paste this URL into your browser when you connect for the first time,
    to login with a token:
        http://(a8af97e42aa5 or 127.0.0.1):8888/?token=c177a3e7d6c43d006e52ad2a2
06dba70bb013e245612657b
```

Here's my code inside the notebook:



Here I create an RDD from the train.csv file.

In [8]:
```
val data = sc.textFile("train.csv")
```

```
data = train.csv MapPartitionsRDD[1] at textFile at <console>:31
```

Out[8]:
```
train.csv MapPartitionsRDD[1] at textFile at <console>:31
```

The main function performs the following steps:

- Skips the header
- Extracts the fields that we need to use in our query
- Filters out non-couples
- Groups everything by hotel country, hotel market, hotel continent
- Sorts everything by the number of group repetitions in descending order
- Leaves only top 3 results
- Prints everything to the screen

In [9]:
```
val header = data.first() // header
val result = data.filter(row => row != header) // skip header
    .map(extractNecessaryFields) // extract necessary fields
    .filter(_._4 == 2) // only choose couples
    .groupBy(row => (row._1, row._2, row._3)) // group by hotel country
    .mapValues(_.size) // transform Iterable[(String,String,String,Int)
    .sortBy(kv => kv._2, false) // sort by the number of people in desc
    .take(3) // leave only top 3 results
```

```
header = date_time,site_name,posa_continent,user_location_country,user
_location_region,user_location_city,orig_destination_distance,user_id,
is_mobile,is_package,channel,srch_ci,srch_co,srch_adults_cnt,srch_chil
dren_cnt,srch_rm_cnt,srch_destination_id,srch_destination_type_id,is_b
ooking,cnt,hotel_continent,hotel_country,hotel_market,hotel_cluster
result = Array(((2,50,1),1277716), ((2,50,2),275737), ((4,8,1),14153
5))
```

Out[9]:
```
Array(((2,50,1),1277716), ((2,50,2),275737), ((4,8,1),141535))
```

In [10]:
```
sc.stop()
```