Week 5 - main homework

Description

Thanks to the pipeline implemented last week our data is classified. We're able to distinguish valid and invalid records. Now it's time to build our business use cases on top of it!

The first and the most important use case required by the business are sessions. They want to know the activity of every user, so be able to know his visit duration, the pages he visited most often during the last few days and so on. We'll implement a POC code to show them that it's possible.

Apart from that, after the coding exercises, I also prepared some open questions that you should be able to answer with the help of covered lessons and the code you wrote for part 1.

Happy coding 🍀

Part 1 - coding exercise

The requirement for this exercise is to take the file with some user events and generate one session for every visit_id. By a "session" I mean one JSON structure composed of the following fields:

Field name	Field type	Explanation
session_id	text	The identifier for the session. Can be the same as visit_id field.
start_time	text	The action time of the first event in the session.
duration_seconds	int	The session duration in seconds. It's calculated as "last log event time + random(1, 3) seconds"
user_id	long	The unique identifier of the user related to the session. Equal to user_id field
pages	list	
- page	structure	

- name	string	The name of the visited page. Equal to page.current in the input logs.
- event_time	date time	The moment when the user opened this page. Equal to event_time in the input_logs.
source	structure	
-site	string	The visited website address. Equal to source.site.
-api_version	string	The API version that sent the event. Equal to source.api_version

Below you can find an example of the event to generate:

```
{"session_id": "abcdef", "start_time": "2020-05-04T10:30:12+00:00",
"duration_seconds": 10, "user_id": 102, "pages": [{"name":
"home.html", "event_time": "2020-05-04T10:30:12+00:00"}, {"name":
"contact.html", "event_time": "2020-05-04T10:30:16+00:00"}, {"name":
"home.html", "event_time": "2020-05-04T10:30:22+00:00"}], "source":
{"site": "partner_a.com", "api_version": "v2"}}
```

The implementation flow:

1. Read the data from the file in the JSON format.
    a. in the reader, please define the schema in `.schema(...)` method. You can use the information from the last week's exercise.
2. For PySpark,
    a. convert the `DataFrame` to `RDD` of type tuple. An example of tuple is a pair of values like (1, "a"), (2, "b") where the first element will be later used to group similar items. To make the conversion, define a function that will take the Row of DataFrame and put the grouping key in the first position and remaining data to use in the second position of the tuple.
    b. use `groupByKey(...)` you to group similar records with the first items from the tuple
    c. use `mapValues(...)` to transform the list of second items of the tuple into the session
    d. extract only values - we don't need to write the keys in the JSON; it can be done with `values()` function
    e. convert the `RDD` into `DataFrame` with `.toDF()` method, in order to use JSON sink.
3. For Scala/Java Spark, you can use groupByKey(...) followed by a mapping function inside `mapGroups(...)`.

The file to process is on Github.

Expected outcome

- create a new branch called week5_sessionization_batch_app and push the corresponding homework code there
- create new Pull Request and send me the link

Part 2 - open questions

1. Let's suppose that the data is partitioned by visit_id in our Apache Kafka topic. My questions are:
   a) Do you need to use groupBy in your data processing logic to implement the sessionization pipeline?
   b) If you base your processing logic per partition basis (no group by in the code), what change in your topic topology (replication factor, partitions, something else) can break your processing logic?
   c) What are the dangers of partitioning records by visit_id?

Expected outcome

- create a new branch called week5_questions
- create a file called answers.md (it uses a Markdown syntax, you can find a cheat sheet here: https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet)
- copy questions from the list
- put your answer under every question
- create new Pull Request and send me the link