Marketing Campaign Data Analysis Using PySpark

Below are the sample data for delivered AD campaigns which will be in 100s of gb per day. Various events captured are impressions, clicks and video ads.

File name - ad_campaigns_data.json


```
{
    "campaign_id": "ABCDFAE",
    "campaign_name": "Food category target campaign",
    "campaign_country": "USA",
    "os_type": "ios",
    "device_type": "apple",
    "place_id": "CASSBB-11",
    "user_id": "1264374214654454321",
    "event_type": "impression",
    "event_time": "2018-10-12T13:10:05.000Z"
  }
  {
    "campaign_id": "ABCDFAE",
    "campaign_name": "Food category target campaign",
    "campaign_country": "USA",
    "os_type": "android",
    "device_type": "MOTOROLA",
    "place_id": "CADGBD-13",
    "user_id": "1674374214654454321",
    "event_type": "impression",
    "event_time": "2018-10-12T13:09:04.000Z"
  }
  {
    "campaign_id": "ABCDFAE",
    "campaign_name": "Food category target campaign",
    "campaign_country": "USA",
    "os_type": "android",
    "device_type": "SAMSUNG",
    "place_id": "BADGBA-12",
    "user_id": "5747421465445443",
```

```
    "event_type": "video ad",
    "event_time": "2018-10-12T13:10:10.000Z"
  }
  {
    "campaign_id": "ABCDFAE",
    "campaign_name": "Food category target campaign",
    "campaign_country": "USA",
    "os_type": "android",
    "device_type": "SAMSUNG",
    "place_id": "CASSBB-11",
    "user_id": "1864374214654454132",
    "event_type": "click",
    "event_time": "2018-10-12T13:10:12.000Z"
  }
```


Below are the sample user profile data which will be in few 100 gbs

File name - user_profile_data.json


```
{
    "user_id": "1264374214654454321",
    "country": "USA",
    "gender": "male",
    "age_group": "18-25",
    "category": [
      "shopper",
      "student"
    ]
  }
  {
    "user_id": "1674374214654454321",
    "country": "USA",
    "gender": "female",
    "age_group": "25-50",
    "category": [
      "parent"
    ]
  }
  {
```

```
    "user_id": "5747421465445443",
    "country": "USA",
    "gender": "male",
    "age_group": "25-50",
    "category": [
      "shopper",
      "parent",
      "professional"
    ]
  }
  {
    "user_id": "1864374214654454132",
    "country": "USA",
    "gender": "male",
    "age_group": "50+",
    "category": [
      "professional"
    ]
  }
  {
    "user_id": "14537421465445443",
    "country": "USA",
    "gender": "female",
    "age_group": "18-25",
    "category": [
      "shopper",
      "student"
    ]
  }
  {
    "user_id": "25547421465445443",
    "country": "USA",
    "gender": "female",
    "age_group": "50+",
    "category": [
      "shopper",
      "professional"
    ]
  }
```

Below are the store file which will be in few 100 mbs

File name - store_data.json

```
{
    "store_name": "McDonald",
    "place_ids": [
      "CASSBB-11",
      "CADGBD-13",
      "FDBEGD-14"
    ]
  }
  {
    "store_name": "BurgerKing",
    "place_ids": [
      "CASSBB-11"
    ]
  }
  {
    "store_name": "Macys",
    "place_ids": [
      "BADGBA-13",
      "CASSBB-15",
      "FDBEGD-15"
    ]
  }
  {
    "store_name": "shoppers stop",
    "place_ids": [
      "BADGBA-12"
    ]
  }
```

Questions:

1. Load ad_campaigns_data.json, user_profile_data.json and store_data.json files in HDFS
2. Write PySpark code in Jupyter notebook to solve below mentioned analytical problems
3. Store processed json data of each problem statement into different HDFS output directory
4. Once output data is available into HDFS, create external Hive tables on top of it using Json Serde

Q1. Analyse data for each campaign_id, date, hour, os_type & value to get all the events with counts

Sample output:


```
{
  "campaign_id": "ABCDFAE",
  "date": "2018-10-12",
  "hour": "13",
  "type": "os_type",
  "value": "android",
  "event": {
    "impression": 2,
    "click": 1,
    "video ad": 1
  }
}
```

```
  {
    "campaign_id": "ABCDFAE",
    "date": "2018-10-12",
    "hour": "13",
    "type": "os_type",
    "value": "ios",
    "event": {
      "impression": 1
    }
  }
```

```
{
  "campaign_id": "SFCDFAD",
  "date": "2018-10-12",
  "hour": "11",
  "type": "os_type",
  "value": "android",
  "event": {
    "impression": 2
  }
}
{
  "campaign_id": "SFCDFAD",
  "date": "2018-10-12",
  "hour": "11",
  "type": "os_type",
  "value": "ios",
  "event": {
    "impression": 1
  }
}
```


Q2.Analyse data for each campaign_id, date, hour, store_name & value to get all the events with counts

Sample output:


```
{
    "campaign_id": "ABCDFAE",
    "date": "2018-10-12",
    "hour": "13",
    "type": "store_name",
    "value": "McDonald",
    "event": {
      "impression": 2,
      "click": 1,
      "video ad": 1
    }
  }
  {
```

```
  "campaign_id": "ABCDFAE",
  "date": "2018-10-12",
  "hour": "13",
  "type": "store_name",
  "value": "BurgerKing",
  "event": {
    "impression": 2,
    "click": 1,
    "video ad": 1
  }
}
{
  "campaign_id": "ABCDFAE",
  "date": "2018-10-12",
  "hour": "13",
  "type": "store_name",
  "value": "shoppers stop",
  "event": {
    "impression": 1
  }
}
{
  "campaign_id": "SFCDFAD",
  "date": "2018-10-12",
  "hour": "11",
  "type": "store_name",
  "value": "Macys",
  "event": {
    "impression": 1
  }
}
{
  "campaign_id": "SFCDFAD",
  "date": "2018-10-12",
  "hour": "11",
  "type": "store_name",
  "value": "shoppers stop",
  "event": {
    "impression": 1
  }
}
```

Q3.Analyse data for each campaign_id, date, hour, gender_type & value to get all the events with counts

Sample output:


```
{
    "campaign_id": "ABCDFAE",
    "date": "2018-10-12",
    "hour": "13",
    "type": "gender",
    "value": "male",
    "event": {
      "impression": 2,
      "click": 1,
      "video ad": 1
    }
  }
  {
    "campaign_id": "ABCDFAE",
    "date": "2018-10-12",
    "hour": "13",
    "type": "gender",
    "value": "female",
    "event": {
      "impression": 1
    }
  }
  {
    "campaign_id": "SFCDFAD",
    "date": "2018-10-12",
    "hour": "11",
    "type": "gender",
    "value": "male",
    "event": {
      "impression": 1
    }
  }
  {
```

```
    "campaign_id": "SFCDFAD",
    "date": "2018-10-12",
    "hour": "11",
    "type": "gender",
    "value": "female",
    "event": {
      "impression": 2
    }
  }
```

EXPECTATIONS

- The code **MUST** be working .i.e. no compile time error(s)
- MUST be demo-able .i.e. There MUST be an output for the questions asked.
- The output for the questions MUST be in the expected format ONLY
- The code should be modular