

Real-Time Ads Data Processing Into Cassandra

1. This project was about processing real-time advertisement data using Spark Streaming to gain business insights and store the aggregated data into Cassandra.
2. I set up confluent kafka in local system and created topic named as ads_data. Wrote a python script to generate ads data in json format using 'random_mock_data_generator.py' file

random_mock_data_genrator.py:

```
# Create a list of random data
data = []
for i in range(1500):
    # Create an ISO string
    iso_string = "2023-08-23T12:01:05Z"

    # Convert the ISO string to a datetime object
    datetime_object = datetime.datetime.fromisoformat(iso_string)
    #print(datetime_object)

    # Create a timedelta object representing few seconds
    timedelta_object = datetime.timedelta(seconds=i)

    # Add the timedelta object to the datetime object
    datetime_object += timedelta_object

    # Convert the datetime object back to an ISO string
    new_iso_string = datetime_object.isoformat().replace('+00:00', 'Z')

    # Print the new ISO string
    #print(datetime_object)
    print(new_iso_string)

    ad_ids = [12345, 54321, 13245, 54231, 14235, 53241, 15234, 43251, 67891, 19876, 68791, 19786, 69781, 18796]

    data.append({
        "ad_id": str(random.choice(ad_ids)),
        "timestamp": new_iso_string,
        "clicks": random.randint(2, 52),
        "views": random.randint(15, 167),
        "cost": round(random.uniform(50.75, 112.66), 2)
    })

# Write the data to a JSON file
with open("mock_ads_data.json", "w") as f:
    json.dump(data, f)
```

mock_ads_data.json:

```
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:01:05Z",
  "clicks": 38,
  "views": 78,
  "cost": 75.37
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:01:06Z",
  "clicks": 8,
  "views": 145,
  "cost": 63.6
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:01:07Z",
  "clicks": 48,
  "views": 119,
  "cost": 71.79
},
{
  "ad_id": "18796",
  "timestamp": "2023-08-23T12:01:08Z",
  "clicks": 8,
  "views": 166,
  "cost": 64.56
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:01:09Z",
  "clicks": 12,
  "views": 123,
  "cost": 89.12
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:01:10Z",
  "clicks": 25,
  "views": 156,
  "cost": 95.43
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:01:11Z",
  "clicks": 35,
  "views": 145,
  "cost": 102.56
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:01:12Z",
  "clicks": 45,
  "views": 134,
  "cost": 110.67
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:01:13Z",
  "clicks": 55,
  "views": 123,
  "cost": 118.78
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:01:14Z",
  "clicks": 65,
  "views": 112,
  "cost": 126.89
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:01:15Z",
  "clicks": 75,
  "views": 101,
  "cost": 135.0
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:01:16Z",
  "clicks": 85,
  "views": 90,
  "cost": 143.11
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:01:17Z",
  "clicks": 95,
  "views": 79,
  "cost": 151.22
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:01:18Z",
  "clicks": 105,
  "views": 68,
  "cost": 159.33
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:01:19Z",
  "clicks": 115,
  "views": 57,
  "cost": 167.44
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:01:20Z",
  "clicks": 125,
  "views": 46,
  "cost": 175.55
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:01:21Z",
  "clicks": 135,
  "views": 35,
  "cost": 183.66
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:01:22Z",
  "clicks": 145,
  "views": 24,
  "cost": 191.77
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:01:23Z",
  "clicks": 155,
  "views": 13,
  "cost": 199.88
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:01:24Z",
  "clicks": 165,
  "views": 2,
  "cost": 207.99
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:01:25Z",
  "clicks": 175,
  "views": 1,
  "cost": 216.1
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:01:26Z",
  "clicks": 185,
  "views": 0,
  "cost": 224.21
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:01:27Z",
  "clicks": 195,
  "views": 0,
  "cost": 232.32
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:01:28Z",
  "clicks": 205,
  "views": 0,
  "cost": 240.43
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:01:29Z",
  "clicks": 215,
  "views": 0,
  "cost": 248.54
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:01:30Z",
  "clicks": 225,
  "views": 0,
  "cost": 256.65
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:01:31Z",
  "clicks": 235,
  "views": 0,
  "cost": 264.76
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:01:32Z",
  "clicks": 245,
  "views": 0,
  "cost": 272.87
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:01:33Z",
  "clicks": 255,
  "views": 0,
  "cost": 280.98
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:01:34Z",
  "clicks": 265,
  "views": 0,
  "cost": 289.09
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:01:35Z",
  "clicks": 275,
  "views": 0,
  "cost": 297.2
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:01:36Z",
  "clicks": 285,
  "views": 0,
  "cost": 305.31
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:01:37Z",
  "clicks": 295,
  "views": 0,
  "cost": 313.42
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:01:38Z",
  "clicks": 305,
  "views": 0,
  "cost": 321.53
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:01:39Z",
  "clicks": 315,
  "views": 0,
  "cost": 329.64
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:01:40Z",
  "clicks": 325,
  "views": 0,
  "cost": 337.75
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:01:41Z",
  "clicks": 335,
  "views": 0,
  "cost": 345.86
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:01:42Z",
  "clicks": 345,
  "views": 0,
  "cost": 353.97
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:01:43Z",
  "clicks": 355,
  "views": 0,
  "cost": 362.08
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:01:44Z",
  "clicks": 365,
  "views": 0,
  "cost": 370.19
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:01:45Z",
  "clicks": 375,
  "views": 0,
  "cost": 378.3
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:01:46Z",
  "clicks": 385,
  "views": 0,
  "cost": 386.41
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:01:47Z",
  "clicks": 395,
  "views": 0,
  "cost": 394.52
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:01:48Z",
  "clicks": 405,
  "views": 0,
  "cost": 402.63
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:01:49Z",
  "clicks": 415,
  "views": 0,
  "cost": 410.74
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:01:50Z",
  "clicks": 425,
  "views": 0,
  "cost": 418.85
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:01:51Z",
  "clicks": 435,
  "views": 0,
  "cost": 426.96
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:01:52Z",
  "clicks": 445,
  "views": 0,
  "cost": 435.07
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:01:53Z",
  "clicks": 455,
  "views": 0,
  "cost": 443.18
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:01:54Z",
  "clicks": 465,
  "views": 0,
  "cost": 451.29
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:01:55Z",
  "clicks": 475,
  "views": 0,
  "cost": 459.4
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:01:56Z",
  "clicks": 485,
  "views": 0,
  "cost": 467.51
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:01:57Z",
  "clicks": 495,
  "views": 0,
  "cost": 475.62
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:01:58Z",
  "clicks": 505,
  "views": 0,
  "cost": 483.73
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:01:59Z",
  "clicks": 515,
  "views": 0,
  "cost": 491.84
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:02:00Z",
  "clicks": 525,
  "views": 0,
  "cost": 500.0
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:02:01Z",
  "clicks": 535,
  "views": 0,
  "cost": 508.16
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:02:02Z",
  "clicks": 545,
  "views": 0,
  "cost": 516.32
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:02:03Z",
  "clicks": 555,
  "views": 0,
  "cost": 524.48
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:02:04Z",
  "clicks": 565,
  "views": 0,
  "cost": 532.64
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:02:05Z",
  "clicks": 575,
  "views": 0,
  "cost": 540.8
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:02:06Z",
  "clicks": 585,
  "views": 0,
  "cost": 548.96
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:02:07Z",
  "clicks": 595,
  "views": 0,
  "cost": 557.12
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:02:08Z",
  "clicks": 605,
  "views": 0,
  "cost": 565.28
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:02:09Z",
  "clicks": 615,
  "views": 0,
  "cost": 573.44
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:02:10Z",
  "clicks": 625,
  "views": 0,
  "cost": 581.6
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:02:11Z",
  "clicks": 635,
  "views": 0,
  "cost": 589.76
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:02:12Z",
  "clicks": 645,
  "views": 0,
  "cost": 597.92
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:02:13Z",
  "clicks": 655,
  "views": 0,
  "cost": 606.08
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:02:14Z",
  "clicks": 665,
  "views": 0,
  "cost": 614.24
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:02:15Z",
  "clicks": 675,
  "views": 0,
  "cost": 622.4
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:02:16Z",
  "clicks": 685,
  "views": 0,
  "cost": 630.56
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:02:17Z",
  "clicks": 695,
  "views": 0,
  "cost": 638.72
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:02:18Z",
  "clicks": 705,
  "views": 0,
  "cost": 646.88
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:02:19Z",
  "clicks": 715,
  "views": 0,
  "cost": 655.04
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:02:20Z",
  "clicks": 725,
  "views": 0,
  "cost": 663.2
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:02:21Z",
  "clicks": 735,
  "views": 0,
  "cost": 671.36
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:02:22Z",
  "clicks": 745,
  "views": 0,
  "cost": 679.52
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:02:23Z",
  "clicks": 755,
  "views": 0,
  "cost": 687.68
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:02:24Z",
  "clicks": 765,
  "views": 0,
  "cost": 695.84
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:02:25Z",
  "clicks": 775,
  "views": 0,
  "cost": 704.0
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:02:26Z",
  "clicks": 785,
  "views": 0,
  "cost": 712.16
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:02:27Z",
  "clicks": 795,
  "views": 0,
  "cost": 720.32
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:02:28Z",
  "clicks": 805,
  "views": 0,
  "cost": 728.48
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:02:29Z",
  "clicks": 815,
  "views": 0,
  "cost": 736.64
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:02:30Z",
  "clicks": 825,
  "views": 0,
  "cost": 744.8
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:02:31Z",
  "clicks": 835,
  "views": 0,
  "cost": 752.96
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:02:32Z",
  "clicks": 845,
  "views": 0,
  "cost": 761.12
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:02:33Z",
  "clicks": 855,
  "views": 0,
  "cost": 769.28
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:02:34Z",
  "clicks": 865,
  "views": 0,
  "cost": 777.44
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:02:35Z",
  "clicks": 875,
  "views": 0,
  "cost": 785.6
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:02:36Z",
  "clicks": 885,
  "views": 0,
  "cost": 793.76
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:02:37Z",
  "clicks": 895,
  "views": 0,
  "cost": 801.92
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:02:38Z",
  "clicks": 905,
  "views": 0,
  "cost": 810.08
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:02:39Z",
  "clicks": 915,
  "views": 0,
  "cost": 818.24
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:02:40Z",
  "clicks": 925,
  "views": 0,
  "cost": 826.4
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:02:41Z",
  "clicks": 935,
  "views": 0,
  "cost": 834.56
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:02:42Z",
  "clicks": 945,
  "views": 0,
  "cost": 842.72
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:02:43Z",
  "clicks": 955,
  "views": 0,
  "cost": 850.88
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:02:44Z",
  "clicks": 965,
  "views": 0,
  "cost": 859.04
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:02:45Z",
  "clicks": 975,
  "views": 0,
  "cost": 867.2
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:02:46Z",
  "clicks": 985,
  "views": 0,
  "cost": 875.36
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:02:47Z",
  "clicks": 995,
  "views": 0,
  "cost": 883.52
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:02:48Z",
  "clicks": 1005,
  "views": 0,
  "cost": 891.68
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:02:49Z",
  "clicks": 1015,
  "views": 0,
  "cost": 899.84
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:02:50Z",
  "clicks": 1025,
  "views": 0,
  "cost": 908.0
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:02:51Z",
  "clicks": 1035,
  "views": 0,
  "cost": 916.16
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:02:52Z",
  "clicks": 1045,
  "views": 0,
  "cost": 924.32
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:02:53Z",
  "clicks": 1055,
  "views": 0,
  "cost": 932.48
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:02:54Z",
  "clicks": 1065,
  "views": 0,
  "cost": 940.64
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:02:55Z",
  "clicks": 1075,
  "views": 0,
  "cost": 948.8
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:02:56Z",
  "clicks": 1085,
  "views": 0,
  "cost": 956.96
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:02:57Z",
  "clicks": 1095,
  "views": 0,
  "cost": 965.12
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:02:58Z",
  "clicks": 1105,
  "views": 0,
  "cost": 973.28
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:02:59Z",
  "clicks": 1115,
  "views": 0,
  "cost": 981.44
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:03:00Z",
  "clicks": 1125,
  "views": 0,
  "cost": 989.6
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:03:01Z",
  "clicks": 1135,
  "views": 0,
  "cost": 997.76
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:03:02Z",
  "clicks": 1145,
  "views": 0,
  "cost": 1005.92
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:03:03Z",
  "clicks": 1155,
  "views": 0,
  "cost": 1014.08
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:03:04Z",
  "clicks": 1165,
  "views": 0,
  "cost": 1022.24
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:03:05Z",
  "clicks": 1175,
  "views": 0,
  "cost": 1030.4
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:03:06Z",
  "clicks": 1185,
  "views": 0,
  "cost": 1038.56
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:03:07Z",
  "clicks": 1195,
  "views": 0,
  "cost": 1046.72
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:03:08Z",
  "clicks": 1205,
  "views": 0,
  "cost": 1054.88
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:03:09Z",
  "clicks": 1215,
  "views": 0,
  "cost": 1063.04
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:03:10Z",
  "clicks": 1225,
  "views": 0,
  "cost": 1071.2
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:03:11Z",
  "clicks": 1235,
  "views": 0,
  "cost": 1079.36
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:03:12Z",
  "clicks": 1245,
  "views": 0,
  "cost": 1087.52
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:03:13Z",
  "clicks": 1255,
  "views": 0,
  "cost": 1095.68
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:03:14Z",
  "clicks": 1265,
  "views": 0,
  "cost": 1103.84
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:03:15Z",
  "clicks": 1275,
  "views": 0,
  "cost": 1112.0
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:03:16Z",
  "clicks": 1285,
  "views": 0,
  "cost": 1120.16
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:03:17Z",
  "clicks": 1295,
  "views": 0,
  "cost": 1128.32
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:03:18Z",
  "clicks": 1305,
  "views": 0,
  "cost": 1136.48
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:03:19Z",
  "clicks": 1315,
  "views": 0,
  "cost": 1144.64
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:03:20Z",
  "clicks": 1325,
  "views": 0,
  "cost": 1152.8
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:03:21Z",
  "clicks": 1335,
  "views": 0,
  "cost": 1160.96
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:03:22Z",
  "clicks": 1345,
  "views": 0,
  "cost": 1169.12
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:03:23Z",
  "clicks": 1355,
  "views": 0,
  "cost": 1177.28
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:03:24Z",
  "clicks": 1365,
  "views": 0,
  "cost": 1185.44
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:03:25Z",
  "clicks": 1375,
  "views": 0,
  "cost": 1193.6
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:03:26Z",
  "clicks": 1385,
  "views": 0,
  "cost": 1201.76
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:03:27Z",
  "clicks": 1395,
  "views": 0,
  "cost": 1209.92
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:03:28Z",
  "clicks": 1405,
  "views": 0,
  "cost": 1218.08
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:03:29Z",
  "clicks": 1415,
  "views": 0,
  "cost": 1226.24
},
{
  "ad_id": "13245",
  "timestamp": "2023-08-23T12:03:30Z",
  "clicks": 1425,
  "views": 0,
  "cost": 1234.4
},
{
  "ad_id": "14235",
  "timestamp": "2023-08-23T12:03:31Z",
  "clicks": 1435,
  "views": 0,
  "cost": 1242.56
},
{
  "ad_id": "15234",
  "timestamp": "2023-08-23T12:03:32Z",
  "clicks": 1445,
  "views": 0,
  "cost": 1250.72
},
{
  "ad_id": "43251",
  "timestamp": "2023-08-23T12:03:33Z",
  "clicks": 1455,
  "views": 0,
  "cost": 1258.88
},
{
  "ad_id": "67891",
  "timestamp": "2023-08-23T12:03:34Z",
  "clicks": 1465,
  "views": 0,
  "cost": 1267.04
},
{
  "ad_id": "19876",
  "timestamp": "2023-08-23T12:03:35Z",
  "clicks": 1475,
  "views": 0,
  "cost": 1275.2
},
{
  "ad_id": "68791",
  "timestamp": "2023-08-23T12:03:36Z",
  "clicks": 1485,
  "views": 0,
  "cost": 1283.36
},
{
  "ad_id": "19786",
  "timestamp": "2023-08-23T12:03:37Z",
  "clicks": 1495,
  "views": 0,
  "cost": 1291.52
},
{
  "ad_id": "54321",
  "timestamp": "2023-08-23T12:03:38Z",
  "clicks": 1505,
  "views": 0,
  "cost": 1300.0
},
{
  "ad_id": "12345",
  "timestamp": "2023-08-23T12:03:39Z",
  "clicks": 1515,
  "views": 0,
  "cost": 1308
```

3. Created a producer script which was publishing those json ads data into kafka topic.

```
from confluent_kafka import Producer
import json
import time

def delivery_report(err, msg):
    if err is not None:
        print(f"Message delivery failed: {err}")
    else:
        print(f"Message delivered to {msg.topic()}")

with open('mock_ads_data.json') as f:
    data = json.load(f)

p = Producer({'bootstrap.servers': 'localhost:9092'})

for record in data:
    p.poll(0)
    record_str = json.dumps(record)
    p.produce('ads_data', record_str, callback=delivery_report)
    print("Message Published -> ", record_str)
    time.sleep(0.5) # wait for half second before sending the next record

p.flush()
```

4. Next I set up a spark streaming application where I used the kafka connector to read data from ads_data topic. Then parsed the json data into proper structure using my defined schema.

```
# Create a SparkSession
spark = SparkSession.builder \
    .appName("Ads Stream") \
    .master("local[3]") \
    .config("spark.sql.shuffle.partitions", "2") \
    .config("spark.streaming.stopGracefullyOnShutdown", "true") \
    .config("spark.jars.packages", "org.apache.spark:spark-sql-kafka-0-10_2.12:3.3.0") \
    .config("spark.cassandra.connection.host", "localhost") \
    .config("spark.cassandra.connection.port", "9042") \
    .getOrCreate()

spark.sparkContext.setLogLevel("ERROR")
load_dotenv()

# Define the schema of the JSON data
schema = StructType([
    StructField("ad_id", StringType()),
    StructField("timestamp", TimestampType()),
    StructField("clicks", IntegerType()),
    StructField("views", IntegerType()),
    StructField("cost", FloatType()),
])

# Read the stream from Kafka
df = spark.readStream.format("kafka") \
    .option("kafka.bootstrap.servers", "localhost:9092") \
    .option("subscribe", "ads_data") \
    .option("startingOffsets", "earliest") \
    .load()

# Parse the JSON data and select the fields
df = df.select(from_json(col("value").cast("string"), schema).alias("data").select("data.*")
```

5. Further transformed data by performed windowed based aggregation and included cassandra connection script into spark stream app so that I can write aggregated data into cassandra table.

```
# Perform the aggregation in windows of 1 minute and sliding interval of 30 secs
df = df.groupBy("ad_id", window(df.timestamp, "1 minute", "30 seconds")).agg(sum("clicks").alias("total_clicks"),
                                                                              sum("views").alias("total_views"),
                                                                              sum("cost").alias("total_cost"),
                                                                              sum("cost")/sum("views").alias("avg_cost_per_view"))

##### connect cassandra db #####
|
# This secure connect bundle is autogenerated when you download your SCB,
# if yours is different update the file name below
cloud_config= {
| 'secure_connect_bundle': 'secure-connect-ads-data-db.zip'
| }

# This token JSON file is autogenerated when you download your token,
# if yours is different update the file name below
with open("ads_data_db-token.json") as f:
| | secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

row = session.execute("select release_version from system.local").one()
if row:
| print(row[0])
else:
| print("An error occurred.")

keyspace="ads_data"
##### end #####
```

6. But writing data with the logic was like if entry already existed with 'ad_id' then updated that record or else inserting new entry in table.

```
#### create table #####
# Table does not exist, create it
create_table_query = f"""
CREATE TABLE IF NOT EXISTS ads_data.agg_ads_data (
|   ad_id TEXT,
|   total_clicks INT,
|   total_views INT,
|   total_cost FLOAT,
|   avg_cost_per_view FLOAT,
|   PRIMARY KEY (ad_id)
| )
| """
session.execute(create_table_query)

# Convert Spark DataFrame to Pandas DataFrame
# pandas_df = df.toPandas()

# Write the output to the console in complete mode
write_query = df.writeStream \
.outputMode("update") \
.format("org.apache.spark.sql.cassandra") \
.option("keyspace", "ads_data") \
.option("table", "agg_ads_data") \
.option("truncate", "false") \
.trigger(processingTime="3 second") \
.start() \
.awaitTermination()
```