**Producing Pivot Tables from Array-Based vs Key-Value-Based Kafka Data Records Showed Comparable Performance**

Background

Data records are ingested from Kafka and transformed by PySpark to produce a pivot-cube table with serial number and timestamp as row indices and sensor readings as columns. Two formats of the ingested data are proposed:

1. Array-based format  
   {  
    "dataSourceId": "0.1.0",   
    "sourceFileId": "testdoc.csv",   
    "schemaVersion": "0.1.1",   
    "headers": ["colZero", "colOne", "colTwo", "colThree", "colFour", "plant", "Station"],   
    "data": [  
    ["0", "1", "2", "3", "4", "aPlantName", "aStationName"],   
    ["5", "6", "7", "8", "9", "aPlantName", "aStationName"],   
    ["10", "22", "12", "13", "14", "aPlantName", "aStationName"]  
    ]  
   }
2. Key-value based format  
   {  
    "dataSourceId": "0.1.0",   
    "sourceFileId": "testdoc.csv",   
    "schemaVersion": "0.1.1",   
    "rows": [  
    {"colZero": "0", "colOne": "1", "colTwo": "2", "colThree": "3", "colFour": "4", "plant": "aPlantName", "Station": "aStationName"},   
    {"colZero": "5", "colOne": "6", "colTwo": "7", "colThree": "8", "colFour": "9", "plant": "aPlantName", "Station": "aStationName"},   
    {"colZero": "10", "colOne": "22", "colTwo": "12", "colThree": "13", "colFour": "14", "plant": "aPlantName", "Station": "aStationName"}  
    ]  
   }

Data

A total of 20 sample CSV files from Borg Warner L14 data pool, which sums up to 297 single-row data records.

Transformation Logic

1. Array-based format (kafka-array-to-cube.ipynb)
2. Key-value based format (kafka-key-value-to-cube.ipynb)

Results

Total processing time needed to move data through all the following stages are recorded:

1. Ingest data from Kafka as one record per csv file
2. Apply technical schema and select appropriate fields to transform into readings dataframe
3. Clean up column headers (array-based) or keys (key-value based) by converting all illegal characters into ' \_'
4. Apply business logic mapping to incoming data columns
5. Create pivot table from the data stream using sensor readings as columns
6. Collapse records with the same serial number and timestamp into a single record
7. The pivot results are written continuously into Delta table using ' upsert ' operations

Table 1. Comparison of processing time for array and key-value based formats

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Total time (s)** | | | **Average** | **std** |
| **Array** | 455 | 344 | 348 | 382.3333333 | 62.96295207 |
| **key-value** | 368 | 358 | 353 | 359.6666667 | 7.637626158 |

Discussions and Conclusions

The processing times for both formats are largely comparable, slightly lower for key-value pair.

May move forward with key-value based format.