support-reporting-level1

November 11, 2024

0.1 load data

- `root`: location of the csv files.
- `fromDt` and `toDt` are start and end dates for range

[1]:
```
import sys
sys.path.append('../')

import dataframeLoader as dfl
import pandas as pd

from importlib import reload
reload(dfl)

# Provide csv data location and appliance and timerange information.
root = '../../.dataDir'
fromDt = '2024-09-26'
toDt = '2024-10-05'

# Provide list of prometheus metrics to load.
# metricsArr = ['cpu_used', 'download_workers_count', 'memory_used',␣
  ↪'task_queue_length', 'infra_access_latency', 'pod_cpu_usage',␣
  ↪'pod_memory_usage']
metricsArr = ['cpu_used'
              ,'task_queue_length'
              , 'memory_used'
              ]


daterange=[fromDt, toDt]
df = dfl.loadApplianceTimeSeriesData(root, metricsArr, daterange)
```

```
loading Unstrctured Data from file: SCANPROC-*.csv

loading Strctured Data from file: STRUCTURED-*.csv

processing securiti_appliance_cpu_used-max*.csv

processing securiti_appliance_cpu_used-avg*.csv

processing securiti_appliance_task_queue_length-max*.csv
processing securiti_appliance_task_queue_length-avg*.csv

processing securiti_appliance_memory_used-max*.csv

processing securiti_appliance_memory_used-avg*.csv

loading Unstrctured Data from file: UNSTRUCTURED-*.csv
```

0.2 Generate plotly report

- `appliance_id`: unique identifier of the appliance.

[2]:
```
reload(dfl)
appliance_id='58e98e10-1b19-4c84-93c0-db2ad5903b80'
dfp = df[(df['appliance_id'] == appliance_id)]
# Get Full list of metrics in dataframe
# metrics_category_order = list(dfp.metrics.unique())
# Provide metrics to show from the data frame. Order is preserved.
metrics_category_order = {# "Indicator": "Chart Description"
            "uniqPodCount": "Scheduled Download workers by datasource"
            ,"cpu_used_avg": "Average CPU by Appliance Node/VM"
            , "memory_used_avg": "Average Memory by Appliance Node/VM"
            , "fileDownloadTimeInHrs":  "Time spent by connectors in␣
↪downloading files for scanning"
            , "IdleTimeInHrs": "Cumulative time spent waiting by (all) download␣
↪workers by datasource"
            , "scanTimeInHrs":  "Cumulative time spent scanning by (all)␣
↪download workers by datasource"
            , "dataScannedinGB" :  "Data scanned in Gigabits per hour"
            ,"numberOfColsScanned":  "Number of structured data columns scanned␣
↪per hour"
            , "numberOfChunksScanned":  "Number of structured data row chunks␣
↪(of 64 rows) scanned per hour"
            , "numFilesScanned":  "Number of files/tables scanned per hour"
            , "avgFileSizeInMB":  "Average size of file or table scanned"
            , "task_queue_length_avg":  "Average temporary task queue length␣
↪(indicator of file tasks in queue for download or scanning)"
             }

title = 'Appliance plot for appliance_id '+appliance_id+' between '+fromDt+'␣
↪and '+toDt
fig  = dfl.plotMetricsFacetForApplianceId(dfp, title, metrics_category_order,␣
↪'node_ip', 'GraphColor')
fig.show()
```