json-schema

April 10, 2021

[1]:
```
import os
import sys
import gzip
import json
from pathlib import Path
import csv

import pandas as pd
import s3fs
import pyarrow as pa
from pyarrow.json import read_json
import pyarrow.parquet as pq
import fastavro
import pygeohash
import snappy
import jsonschema
from jsonschema.exceptions import ValidationError


endpoint_url='https://storage.budsc.midwest-datascience.com'

current_dir = Path(os.getcwd()).absolute()
schema_dir = current_dir.joinpath('schemas')
schema_dir.mkdir(parents=True, exist_ok=True)
results_dir = current_dir.joinpath('results')
results_dir.mkdir(parents=True, exist_ok=True)

def read_jsonl_data():
    s3 = s3fs.S3FileSystem(
        anon=True,
        client_kwargs={
            'endpoint_url': endpoint_url
        }
    )
    src_data_path = 'data/processed/openflights/routes.jsonl.gz'
    with s3.open(src_data_path, 'rb') as f_gz:
        with gzip.open(f_gz, 'rb') as f:
            records = [json.loads(line) for line in f.readlines()]


    return records

records = read_jsonl_data()
```

[2]:
```
records[0]
```

[2]:
```
{'airline': {'airline_id': 410,
  'name': 'Aerocondor',
  'alias': 'ANA All Nippon Airways',
  'iata': '2B',
  'icao': 'ARD',
  'callsign': 'AEROCONDOR',
  'country': 'Portugal',
  'active': True},
 'src_airport': {'airport_id': 2965,
  'name': 'Sochi International Airport',
  'city': 'Sochi',
  'country': 'Russia',
  'iata': 'AER',
  'icao': 'URSS',
  'latitude': 43.449902,
  'longitude': 39.9566,
  'altitude': 89,
  'timezone': 3.0,
  'dst': 'N',
  'tz_id': 'Europe/Moscow',
  'type': 'airport',
  'source': 'OurAirports'},
 'dst_airport': {'airport_id': 2990,
  'name': 'Kazan International Airport',
  'city': 'Kazan',
  'country': 'Russia',
  'iata': 'KZN',
  'icao': 'UWKD',
  'latitude': 55.606201171875,
  'longitude': 49.278701782227,
  'altitude': 411,
  'timezone': 3.0,
  'dst': 'N',
  'tz_id': 'Europe/Moscow',
  'type': 'airport',
  'source': 'OurAirports'},
 'codeshare': False,
 'equipment': ['CR2']}
```

1 3.1

1.1 3.1.a JSON Schema

[3]:
```
import requests

def validate_jsonl_data(records):


    schema_path = schema_dir.joinpath("routes-schema.json")
    with open(schema_path) as f:
        validation_csv_path = results_dir.joinpath("validation-results.csv")
        schema = json.load(f)

    with open(validation_csv_path, 'w') as f:
        for i, record in enumerate(records):
            try:
                jsonschema.validate(instance=record,schema= schema)
            except ValidationError as e:
                print('e')
                msg='json routes is invalid'
                return False,msg
            msg='json routes is valid'
            return True,msg
            #pass


validate_jsonl_data(records)
```

[3]:
```
(True, 'json routes is valid')
```

[ ]:
```
import os
cwd = os.getcwd()
cwd
```