ETL Process

Close Approach JSON Data

NASA Web API call

Historical

https://ssd-api.jpl.nasa.gov/cad.api?date-min=1900-01-01&date-max=2021-12-31&dist-max=0.2'

Future

https://ssd-api.jpl.nasa.gov/cad.api?date-min=2022-01-01&date-max=2100-01-01&dist-max=0.2

JSON Data

{

- "signature": {
  - "source": "NASA/JPL SBDB Close Approach Data API",
  - "version": "1.4"

  },
- "count": "34780",
- "fields": [
  - "des",
  - "orbit_id",
  - "jd",
  - "cd",
  - "dist",
  - "dist_min",
  - "dist_max",
  - "v_rel",
  - "v_inf",
  - "t_sigma_f",
  - "h"

  ],
- "data": [
  - [
    - "2012 BV13",
    - "11",
    - "2451911.004746058",
    - "2001-Jan-01 12:07",
    - "0.170022291915518",
    - "0.168216002365715",
    - "0.171829680144807",
    - "6.87897309388932",
    - "6.87669456357091",

Transformation

```
spark.sparkContext.addFile(url_endpoint)

# read cad json file into spark session
cad_json_file = SparkFiles.get(json_filename)
json_df = spark.read.json(cad_json_file, multiLine=True)

# create temporary dataframe from data column in dataframe
array_data_df = json_df.select(F.explode("data").alias('data'))

# create tabular formatted dataframe
tabular_df = array_data_df.select(array_data_df['data'].getItem(0).alias('des'),
               array_data_df['data'].getItem(1).alias('orbit_id'),
               array_data_df['data'].getItem(2).alias('jd'),
               array_data_df['data'].getItem(3).alias('cd'),
               array_data_df['data'].getItem(4).alias('dist'),
               array_data_df['data'].getItem(5).alias('dist_min'),
               array_data_df['data'].getItem(6).alias('dist_max'),
               array_data_df['data'].getItem(7).alias('v_rel'),
               array_data_df['data'].getItem(8).alias('v_inf'),
               array_data_df['data'].getItem(9).alias('t_sigma_f'),
               array_data_df['data'].getItem(10).alias('h')
               )

# create final dataframe for loading postgres table
cad_final_df = (tabular_df
  .transform(lambda df: df.withColumn("cd", F.to_timestamp(tabular_df["cd"], 'yyyy-MMM-dd HH:mm')))
  .transform(lambda df: df.withColumn("dist", tabular_df["dist"].cast(T.DecimalType(precision=24, scale=16))))
```

Load

```
def load_cad_data_aws_rds(df, mode, table_name):
  """
  Load data in dataframe arg df into aws rds neo database

  args:
    df: dataframe containing source data to load into database
    mode: write mode ie. append, overwrite
    table_name: name of table in database to load data into
  """

  password = getpass('Enter database password')

  # Configure settings for RDS
  jdbc_url="jdbc:postgresql://neo-db.ctohlxwhjvlb.us-east-1.rds.amazonaws.com:5432/neo"
  config = {"user":"postgres",
            "password": password,
            "driver":"org.postgresql.Driver"}

  mode = 'overwrite'
  df.write.jdbc(url=jdbc_url, table=table_name, mode=mode, properties=config)
```