Airbnb ETL Pipeline Documentation

Githublink: https://github.com/sauravjha3010/airbnb_etl.git

ETL Process Details

Data Extraction
The data is extracted from the 'AB_NYC_2019.csv' file using pandas. This CSV file contains information about Airbnb listings in New York City.

Data Transformation
The following transformations are applied to the data:
1. Date normalization: The 'last_review' column is split into separate 'date' and 'time' columns.
2. Missing value handling: The 'reviews_per_month' column is filled with 0 for missing values.
3. New metric calculation:
   - 'price_per_person': Calculated as price divided by minimum nights
   - 'is_expensive': Boolean flag set to True if the price is above the median price
4. Aggregation: Average price per neighborhood group is calculated

Data Loading
The transformed data is loaded into two PostgreSQL tables:
1. 'airbnb_data': Contains all original and transformed columns
2. 'avg_price_by_neighborhood': Contains average price for each neighborhood group

Metaflow Workflow

The Metaflow workflow is defined in the `AirbnbETLFlow` class and consists of the following steps:

1. start: Initializes the flow
2. load_data: Extracts data from the CSV file
3. transform_data: Applies all data transformations
4. load_to_db: Loads the transformed data into PostgreSQL
5. end: Finalizes the flow

To run the Metaflow workflow:

python -m metaflow run src/etl.py

To visualize the Metaflow graph:

python -m metaflow show AirbnbETLFlow

```
## Scalability Considerations

1. Batch Processing: The current implementation uses batch
processing for database operations. For larger datasets, consider
implementing incremental loading.
2. Parallel Processing: Metaflow supports parallel execution of
independent steps. This could be utilized for more complex
transformations.
3. Database Indexing: As the dataset grows, proper indexing of the
PostgreSQL tables will become crucial for query performance.
4. Data Partitioning: For very large datasets, consider partitioning
the data based on date or neighborhood.

## Challenges and Solutions

1. Data Quality: The dataset contained missing values, which were
handled by filling with appropriate default values.
2. Performance: To improve loading performance, data is inserted
into the database in chunks.
3. Workflow Management: Metaflow was used to create a reproducible
and failure-resistant workflow.

## Potential Improvements

1. Data Validation: Implement more robust data validation checks.
2. Error Handling: Enhance error handling and implement automatic
retries for transient failures.
3. Monitoring: Add monitoring and alerting for the ETL process.
4. Data Versioning: Implement a data versioning strategy for
tracking changes over time.
```

Demonstration

DE.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
import pandas as pd
from sqlalchemy import create_engine
import logging

# Set up logging
logging.basicConfig(level=logging.INFO)

# Load data
df = pd.read_csv('/content/AB_NYC_2019.csv')
logging.info(f"Data loaded successfully. Shape: {df.shape}")

# Display first few rows
print(df.head())
```

```
     id                                              name  host_id  \
0  2539                Clean & quiet apt home by the park     2787
1  2595                             Skylit Midtown Castle     2845
2  3647               THE VILLAGE OF HARLEM....NEW YORK !     4632
3  3831                   Cozy Entire Floor of Brownstone     4869
4  5022  Entire Apt: Spacious Studio/Loft by central park     7192

     host_name neighbourhood_group neighbourhood  latitude  longitude  \
0         John            Brooklyn    Kensington  40.64749  -73.97237
1     Jennifer           Manhattan       Midtown  40.75362  -73.98377
2    Elisabeth           Manhattan        Harlem  40.80902  -73.94190
3  LisaRoxanne            Brooklyn  Clinton Hill  40.68514  -73.95976
4        Laura           Manhattan   East Harlem  40.79851  -73.94399

        room_type  price  minimum_nights  number_of_reviews last_review  \
0     Private room    149               1                  9  2018-10-19
1  Entire home/apt    225               1                 45  2019-05-21
2     Private room    150               3                  0         NaN
3  Entire home/apt     89               1                270  2019-07-05
4  Entire home/apt     80              10                  9  2018-11-19

   reviews_per_month  calculated_host_listings_count  availability_365
0               0.21                               6               365
1               0.38                               2               355
2                NaN                               1               365
3               4.64                               1               194
4               0.10                               1                 0
```

Cell 2: Data Transformation

```python
# Normalize data
df['date'] = pd.to_datetime(df['last_review']).dt.date
df['time'] = pd.to_datetime(df['last_review']).dt.time

# Calculate average price per neighborhood
avg_price = df.groupby('neighbourhood_group')['price'].mean().reset_index()

# Handle missing values
df['reviews_per_month'].fillna(0, inplace=True)

# Additional transformations
df['price_per_person'] = df['price'] / df['minimum_nights']
df['is_expensive'] = df['price'] > df['price'].median()

logging.info("Data transformation completed successfully")

# Display transformed data
print(df[['id', 'name', 'date', 'time', 'price_per_person', 'is_expensive']].head())
print("\nAverage price by neighborhood:")
print(avg_price)
```

```
     id                                              name        date  \
0  2539                Clean & quiet apt home by the park  2018-10-19
1  2595                             Skylit Midtown Castle  2019-05-21
2  3647               THE VILLAGE OF HARLEM....NEW YORK !         NaT
3  3831                   Cozy Entire Floor of Brownstone  2019-07-05
4  5022  Entire Apt: Spacious Studio/Loft by central park  2018-11-19

       time  price_per_person  is_expensive
0  00:00:00             149.0          True
1  00:00:00             225.0          True
2       NaT              50.0          True
3  00:00:00              89.0         False
4  00:00:00               0.0         False

Average price by neighborhood:
  neighbourhood_group       price
0               Bronx   87.496792
1            Brooklyn  124.383207
2           Manhattan  196.875814
3              Queens   99.517649
4       Staten Island  114.812332
```

Cell 3: Database Setup and Data Loading

```python
# Set up PostgreSQL
!apt-get install postgresql postgresql-contrib
!service postgresql start
!sudo -u postgres psql -c "CREATE USER colab WITH PASSWORD 'colab';"
!sudo -u postgres psql -c "CREATE DATABASE airbnb_db WITH OWNER colab;"
!sudo -u postgres psql -c "GRANT ALL PRIVILEGES ON DATABASE airbnb_db TO colab;"

# Load data to database
engine = create_engine('postgresql://colab:colab@localhost/airbnb_db')

# Load original data
df.to_sql('airbnb_data', engine, if_exists='replace', index=False, chunksize=1000)

# Load transformed data
avg_price.to_sql('avg_price_by_neighborhood', engine, if_exists='replace', index=False)

logging.info("Data loaded to database successfully")
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
postgresql is already the newest version (14+238).
postgresql-contrib is already the newest version (14+238).
0 upgraded, 0 newly installed, 0 to remove and 45 not upgraded.
 * Starting PostgreSQL 14 database server
   ...done.
ERROR:  role "colab" already exists
ERROR:  database "airbnb_db" already exists
GRANT
```

Cell 4: Database Queries

```
[5] import pandas as pd
    from sqlalchemy import create_engine, text

    engine = create_engine('postgresql://colab:colab@localhost/airbnb_db')

    # Query the airbnb_data table
    query = text("SELECT * FROM airbnb_data LIMIT 5")
    result = pd.read_sql(query, engine)
    print("Sample data from airbnb_data table:")
    print(result)

    # Query the avg_price_by_neighborhood table
    query = text("SELECT * FROM avg_price_by_neighborhood")
    result = pd.read_sql(query, engine)
    print("\nAverage price by neighborhood:")
    print(result)
```

```
[5]
Sample data from airbnb_data table:
     id                                              name  host_id  \
0  2539                Clean & quiet apt home by the park     2787
1  2595                             Skylit Midtown Castle     2845
2  3647               THE VILLAGE OF HARLEM....NEW YORK !     4632
3  3831                   Cozy Entire Floor of Brownstone     4869
4  5022  Entire Apt: Spacious Studio/Loft by central park     7192

     host_name neighbourhood_group neighbourhood  latitude  longitude  \
0         John            Brooklyn    Kensington  40.64749  -73.97237
1     Jennifer           Manhattan       Midtown  40.75362  -73.98377
2    Elisabeth           Manhattan        Harlem  40.80902  -73.94190
3  LisaRoxanne            Brooklyn  Clinton Hill  40.68514  -73.95976
4        Laura           Manhattan   East Harlem  40.79851  -73.94399
```

```
[5]
        room_type  price  minimum_nights  number_of_reviews last_review  \
0     Private room    149               1                  9  2018-10-19
1  Entire home/apt    225               1                 45  2019-05-21
2     Private room    150               3                  0        None
3  Entire home/apt     89               1                270  2019-07-05
4  Entire home/apt     80              10                  9  2018-11-19

   reviews_per_month  calculated_host_listings_count  availability_365  \
0               0.21                               6               365
1               0.38                               2               355
2               0.00                               1               365
3               4.64                               1               194
4               0.10                               1                 0

         date      time  price_per_person  is_expensive
0  2018-10-19  00:00:00             149.0          True
1  2019-05-21  00:00:00             225.0          True
2        None      None              50.0          True
3  2019-07-05  00:00:00              89.0         False
4  2018-11-19  00:00:00               0.0         False

Average price by neighborhood:
  neighbourhood_group       price
0               Bronx   87.496792
1            Brooklyn  124.383207
2           Manhattan  196.875814
3              Queens   99.517649
4       Staten Island  114.812332
```

Cell 5: Error Handling Demonstration

```
# Attempt to load non-existent file
try:
    df_error = pd.read_csv('non_existent_file.csv')
except FileNotFoundError as e:
    logging.error(f"Error loading data: {str(e)}")
    print("File not found error handled successfully")
```

```
ERROR:root:Error loading data: [Errno 2] No such file or directory: 'non_existent_file.csv'
File not found error handled successfully
```