UDACITY

‹ Return to Classroom

Data Lake

REVIEW

CODE REVIEW

HISTORY

Meets Specifications

Congratulations on completing the **Data Lake** Project. You have demonstrated good understanding of Spark and data lakes to build an ETL pipeline for a data lake hosted on S3. The ETL script ran without any errors and the output tables were correctly partitioned. Good job with detailed README. In the review, you'll find some feedback to help you continue to improve on your project.

A few articles of interest:

- TOP FIVE DIFFERENCES BETWEEN DATA LAKES AND DATA WAREHOUSES
  https://www.blue-granite.com/blog/bid/402596/top-five-differences-between-data-lakes-and-data-warehouses
- What Is A Data Lake? A Super-Simple Explanation For Anyone
  https://www.forbes.com/sites/bernardmarr/2018/08/27/what-is-a-data-lake-a-super-simple-explanation-for-anyone/#4f61b70d76e0
- When Should We Load Relational Data to a Data Lake?
  https://www.sqlchick.com/entries/2018/11/13/when-should-we-load-relational-data-to-a-data-lake

ETL

The script, etl.py, runs in the terminal without errors. The script reads song_data and load_data from S3, transforms them to create five different tables, and writes them to partitioned parquet files in table directories on S3.

✓ The ETL script ran without any other errors, creating the five tables.

Each of the five tables are written to parquet files in a separate analytics directory on S3. Each table has its own folder within the directory. Songs table files are partitioned by year and then artist. Time table files are partitioned by year and month. Songplays table files are partitioned by year and month.

The tables are correctly partitioned.

✓ Songs table partitioned by year and artist
✓ Time table partitioned by year and month
✓ Songsplays table partitioned by year and month

Partitioning helps queries to run much faster. See this article on detailed info on partition,
https://mungingdata.com/apache-spark/partitionby/

Each table includes the right columns and data types. Duplicates are addressed where appropriate.

✓ Duplicates are handled appropriately.

Code Quality

The README file includes a summary of the project, how to run the Python scripts, and an explanation of the files in the repository. Comments are used effectively and each function has a docstring.

✓ README
Good job with the well-structured and detailed README. A nice README is a great way to showcase your project to potential employers. Suggestions:

- Add a screenshot or an image (ER Diagram) showing how the fact and dimension tables are connected.
  You can make use of online tools like https://www.lucidchart.com/
- Refer good READMEs on web:
  https://github.com/matiassingers/awesome-readme
  https://bulldogjob.com/news/449-how-to-write-a-good-readme-for-your-github-project
  https://medium.com/@meakaakka/a-beginners-guide-to-writing-a-kickass-readme-7ac01da88ab3

✓ Docstrings
Nice job adding docstrings to all methods in etl.py

Scripts have an intuitive, easy-to-follow structure with code separated into logical functions. Naming for variables and functions follows the PEP8 style guidelines.

- ✓ Well commented - more readable
- ✓ Sensible variable names
- ✓ Nicely organized into logical functions

DOWNLOAD PROJECT

RETURN TO PATH