**PROJECT SPECIFICATION**

**Data Modeling with Postgres**

Table Creation

| CRITERIA | MEETS SPECIFICATIONS |
| --- | --- |
| Table creation script runs without errors. | The script, create\_tables.py, runs in the terminal without errors. The script successfully connects to the Sparkify database, drops any tables if they exist, and creates the tables. |
| Fact and dimensional tables for a star schema are properly defined. | CREATE statements in sql\_queries.py specify all columns for each of the five tables with the right data types and conditions. |

ETL

| CRITERIA | MEETS SPECIFICATIONS |
| --- | --- |
| ETL script runs without errors. | The script, etl.py, runs in the terminal without errors. The script connects to the Sparkify database, extracts and processes the log\_data and song\_data, and loads data into the five tables.  Since this is a subset of the much larger dataset, the solution dataset will only have 1 row with values for value containing ID for both songid and artistid in the fact table. Those are the only 2 values that the query in the sql\_queries.py will return that are not-NONE. The rest of the rows will have NONE values for those two variables. |
| ETL script properly processes transformations in Python. | INSERT statements are correctly written for each table, and handle existing records where appropriate. songs and artists tables are used to retrieve the correct information for the songplays INSERT. |

Code Quality

| CRITERIA | MEETS SPECIFICATIONS |
| --- | --- |
| The project shows proper use of documentation. | 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. |
| The project code is clean and modular. | Scripts have an intuitive, easy-to-follow structure with code separated into logical functions. Naming for variables and functions follows the PEP8 style guidelines. |