Project: Data Modeling with Cassandra

A startup called Sparkify wants to analyze the data they've been collecting on songs and user activity on their new music streaming app. The analysis team is particularly interested in understanding what songs users are listening to. Currently, there is no easy way to query the data to generate the results, since the data reside in a directory of CSV files on user activity on the app.
They'd like a data engineer to create an Apache Cassandra database which can create queries on song play data to answer the questions, and wish to bring you on the project. Your role is to create a database for this analysis. You'll be able to test your database by running queries given to you by the analytics team from Sparkify to create the results.

Project Overview

In this project, you'll apply what you've learned on data modeling with Apache Cassandra and complete an ETL pipeline using Python. To complete the project, you will need to model your data by creating tables in Apache Cassandra to run queries. You are provided with part of the ETL pipeline that transfers data from a set of CSV files within a directory to create a streamlined CSV file to model and insert data into Apache Cassandra tables.
We have provided you with a project template that takes care of all the imports and provides a structure for ETL pipeline you'd need to process this data.

Project Details

Datasets

For this project, you'll be working with one dataset: `event_data`. The directory of CSV files partitioned by date. Here are examples of filepaths to two files in the dataset:

```
event_data/2018-11-08-events.csv
event_data/2018-11-09-events.csv
```

Project Template

To get started with the project, go to the workspace on the next page, where you'll find the project template (a Jupyter notebook file). You can work on your project and submit your work through this workspace.
The project template includes one Jupyter Notebook file, in which:

- you will process the `event_datafile_new.csv` dataset to create a denormalized dataset
- you will model the data tables keeping in mind the queries you need to run
- you have been provided queries that you will need to model your data tables for
- you will load the data into tables you create in Apache Cassandra and run your queries

Project Steps

Below are steps you can follow to complete each component of this project.

Modeling your NoSQL database or Apache Cassandra database

- Design tables to answer the queries outlined in the project template
- Write Apache Cassandra `CREATE KEYSPACE` and `SET KEYSPACE` statements
- Develop your `CREATE` statement for each of the tables to address each question
- Load the data with `INSERT` statement for each of the tables
- Include `IF NOT EXISTS` clauses in your `CREATE` statements to create tables only if the tables do not already exist. We recommend you also include `DROP TABLE` statement for each table, this way you can run drop and create tables whenever you want to reset your database and test your ETL pipeline
- Test by running the proper select statements with the correct `WHERE` clause

Build ETL Pipeline

- Implement the logic in section Part I of the notebook template to iterate through each event file in `event_data` to process and create a new CSV file in Python
- Make necessary edits to Part II of the notebook template to include Apache Cassandra `CREATE` and `INSERT` statements to load processed records into relevant tables in your data model
- Test by running `SELECT` statements after running the queries on your database

Project requirements

ETL Pipeline Processing

Criteria	Submission Requirements
Student completes the ETL pipeline procedures.	Student creates `event_data_new.csv` file.
Student uses the correct datatype for each Cassandra `CREATE` statement.	Student uses the appropriate datatype within the `CREATE` statement.

Data Modeling

Criteria	Submission Requirements
Student creates correct data models for the queries they need to run.	Student creates the correct Apache Cassandra tables for each of the three queries. The `CREATE TABLE` statement should include the appropriate table.
Student can set up the data model correctly to generate the exact responses posed in the questions.	Student demonstrates good understanding of data modeling by generating correct SELECT statements to generate the result being asked for in the question. The SELECT statement should NOT use `ALLOW FILTERING` to generate the results.
Student models the data by using appropriate table names.	Student should use table names that reflect the query and the result it will generate. Table names should include alphanumeric characters and underscores, and table names must start with a letter.

Student has given careful thought to how the data is modeled in the table and the sequence and order in which data is partitioned, inserted and retrieved from the table.	The sequence in which columns appear should reflect how the data is partitioned and the order of the data within the partitions.

PRIMARY KEYS

Criteria	Submission Requirements
The PRIMARY key for each table should uniquely identify each row in each of the tables.	The combination of the PARTITION KEY alone or with the addition of CLUSTERING COLUMNS should be used appropriately to uniquely identify each row.

Presentation

Criteria	Submission Requirements
Student provides responses to the questions.	The notebooks should include a description of the query the data is modeled after.
Students notebook code should be clean and modular.	Code should be organized well into the different queries. Any in-line comments that were clearly part of the project instructions should be removed so the notebook provides a professional look.

Suggestions to Make Your Project Stand Out

- You can add description of your PRIMARY KEY and how you arrived at the decision to use each for the query
- Use Panda dataframes to add columns to your query output