

Data Modeling with Cassandra

REVIEW

HISTORY

Meets Specifications

Dear student,

Congratulations on completing the second project of the Nanodegree. You have put a lot of effort and I am sure you have learned a lot throughout the journey. Great job 🍑 The submission shows that you have understood the concepts of dimensional modeling in Cassandra pretty well. Some of the key points you should feel really confident now as you did well in the submission:

- Selection of datatypes. eg `float` for `length`, `int` for `user_id`.
- Creating separate tables for each query
- Choices of COMPOSITE KEYS / CLUSTERING columns.
- Maintaining order of COMPOSITE KEYS to CREATE and INSERT statements.

We hope the lessons gave a very good base but there is a lot to explore. You can take a look at this free course on [Apache Cassandra](#) from DataStax itself if you want to hone your skills on Cassandra Data Modeling.

All the very best for the upcoming projects and all your endeavors. 🍑

ETL Pipeline Processing

Student creates `event_data_new.csv` file.

- **Perfect!!** The ETL process creates the `event_data_new.csv` from the log files in the staging area.
- You have followed the instructions properly and there are 6821 records as expected.

Student uses the appropriate datatype within the `CREATE` statement.

- Good work with the datatypes selection for the fields. The very basic importance of assigning correct datatype is when we need to do data wrangling/aggregation. What does that mean? For example, if you are asked to sum up the total length of songs a user listened to in a particular session, having the length's datatype as the float will help to get the end result perfectly.
- While data modeling you may like to keep this [official documentation](#) on datatypes handy for future Cassandra project when assigning the correct data types.

Data Modeling

Student creates the correct Apache Cassandra tables for each of the three queries.

The `CREATE TABLE` statement should include the appropriate table.

Perfect

- **You have adhered to one table per query rule of Apache Cassandra.** No two queries are answered from the same table. This is important as this decides the efficiency of query retrieval.
- It is understandable that keeping similar data will eat a lot of space. *But if you compare the cost of storage to CPU, the former will be always preferred.* This [documentation](#) clearly describes the fundamental modeling concept of Cassandra, why data duplication in separate tables are allowed.




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.

Awesome

- The specific field names are used to retrieve data. **This is also a very important point when we are accessing a very huge database and doing `SELECT *` can be very costly in terms of reading.** This is a very critical point here. As I mentioned above which is a basic principle of Cassandra data modeling, *Prefer storage cost over CPU cost or read/write cost.*, so we do not really mind keeping duplicate data, but while reading we must be alert not to increase IO operations.
- To shout out an example you have perfectly done: In Query 3, the SELECT statement should not require anything more than the user's first name and last name which is intended.

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.

- **Great work here!!** The table names reflect the query. **Cassandra is a query first database where data models/tables are created based on the query it is going to answer** (hence we create different tables for different queries). To help the stakeholders to quickly understand what is going to be the use of any specific table, **a name which reflects the query will be more helpful.**
 - artist_title_length 
 - artist_song_user 
 - user_song 

The sequence in which columns appear should reflect how the data is partitioned and the order of the data within the partitions.

Perfect

- You have done a splendid job here by maintaining the sequence of CREATE and INSERT to the order of COMPOSITE PRIMARY KEY and CLUSTERING COLUMNS. This is one of the key takeaways from this project and you did it perfectly. **The data should be inserted and retrieved in the same order as to how the COMPOSITE PRIMARY KEY is set up.** This is important because Apache Cassandra is a partition row store, which means the partition key determines where any particular row is stored and on which node.
- Here is a [DataStax documentation](#) for you further understanding. and an amazing [article](#) on Cassandra Data Modelling.

PRIMARY KEYS

The combination of the PARTITION KEY alone or with the addition of CLUSTERING COLUMNS should be used appropriately to uniquely identify each row.

Awesome

- The COMPOSITE KEY combination for the first table is in place. **The `sessionId` and `itemInSession` perfectly filters the data as per the query we are answering from the table.**
- You have explicitly defined `itemInSession` as CLUSTERING COLUMN for the second table to fulfill the sorting requirement.
- Good use of `userId` in the third table to uniquely identify each record per song per user.

- Optionally you can read [this blog](#) which discusses Clustering columns, partition keys with examples.

Presentation

The notebooks should include a description of the query the data is modeled after.

- This rubric is to evaluate the project presentation in a professional and polished manner.
Amazing work with the notebook presentation. The Queries are divided into subtasks. Each task has proper headings and description about schema design. 👍

I would suggest you using Markdown cells, rather than code comments to make it more appealing. You can do that by:

Going to Cell tab -> Cell Type-> Markdown cells

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.

- The TODOs and inline comments are removed. Consider you are presenting this project as this notebook to an employer. We do not need to have all the questions and instructions to be in there. It is more professional when separate them as tasks and mention what we have done and why we have done it. You have done it perfectly. Great job 👍