Meets Specifications

Dear Student,  
Excellent job submitting a great project   
And congratulations you did very well.

Kindly go through the suggestions carefully and please take them positively and constructively as an opportunity to learn and grow and also improve the overall quality of your work. Learning in data science never stops and as you will see, there is not always one perfect way to do things but intuition and experimentation are very often required depending on the situation.

For further learning in Cassandra you can refer to datastax self paced courses here - <https://academy.datastax.com/courses>

And as a general companion to the Nanodegree, I recommend the Orielly series of DBA courses there are over 4 parts for this series.

* DBA 1: Introduction to Database Administration  
  <http://archive.oreilly.com/oreillyschool/courses/dba1/index.html>
* DBA 2: Administering MySQL  
  <http://archive.oreilly.com/oreillyschool/courses/dba2/index.html>
* DBA 3: Creating a Data Warehouse  
  <http://archive.oreilly.com/oreillyschool/courses/dba3/index.html>
* DBA 4: Analyzing Data  
  <http://archive.oreilly.com/oreillyschool/courses/dba4/index.html>

*Also If this review helped you in some way, Please consider rating it positively.*  
I wish you all the best for your future endeavors. Keep learning and stay udacious!

**ETL Pipeline Processing**

**Student creates event\_data\_new.csv file.**

Nice job ! I see the event\_data\_new.csv file, which indicates you followed the ETL pipeline to create the csv file.

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

You use the appropriate datatypes. Be sure to keep this link handy for future Cassandra project when assigning the correct data types  
(<https://docs.datastax.com/en/cql/3.3/cql/cql_reference/cql_data_types_c.html>)

You might also check out this post regarding the difference between float and decimal in Cassandra  
(<https://www.oreilly.com/library/view/learning-apache-cassandra/9781787127296/dcd51887-3fad-4e4e-b1d1-491ef3d227b6.xhtml>)

**Data Modeling**

**Student creates the correct Apache Cassandra tables for each of the three queries. The CREATE TABLE statement should include the appropriate table.**

Great job! You followed the one table per query rule of Apache Cassandra. You are not replicating the same table for all three queries, which defies that rule. You have three distinct tables with unique tables names and uses appropriate CREATE table statements.

**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 work ! Your data is modeled correctly to generate the exact responses posed in the questions.  
The specific field names are used to retrieve data. This is also a very important point when we are accessing a very huge database. Doing SELECT \* can be very costly in terms of reading.

You may also check this link for some Simple Examples of Cassandra Data Model - <https://www.guru99.com/cassandra-data-model-rules.html>

**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.**

Nice work! Each of your table names indicate what the respective query is about.

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

Excellent work! This is probably one of the most important learnings that I want you to walk away with from this lesson. Apache Cassandra is a partition row store, which means the partition key determines which node a particular row is stored on. With the Primary key (which includes the Partition Key and any clustering columns), the partitions are distributed across the nodes of the cluster. It determines how data are chunked for write purposes. Any clustering column(s) would determine the order in which the data is sorted within the partition.

Here are some DataStax documentation you can refer to for the same:  
<https://docs.datastax.com/en/dse/5.1/cql/cql/cql_using/whereClustering.html>

This youtube video might be helpful when understanding how data is distributed across the nodes of the cluster in Cassandra.

(<https://www.youtube.com/watch?v=xQnIN9bW0og&t=>)

**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 work at understanding and then implementing the PRIMARY KEY with a COMPOSITE Partition for both the CREATE and INSERT statements. I am so glad you took the time to understand this and make this change. For e.g., in query 1, each row is uniquely identified with the combination of userId, sessionId and clustering column itemInSession. Nice work taking the time to look through the data and figuring this out. This is a very important lesson takeaway regarding how COMPOSITE PARTITION key works to uniquely identify each row.

Refer here for more details on difference between various aspects of keys - <http://sudotutorials.com/tutorials/cassandra/cassandra-primary-key-cluster-key-partition-key.html>

**Presentation**

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

Good work with the included description of the query the data is modeled after. Nice communication skills demonstrated here! Keep it up!

You might want to look into using PrettyTable[<http://zetcode.com/python/prettytable/>]. For example:

# use pretty table to display data in tabular form and include headings

t = PrettyTable(['Artist', 'Song', 'Length'])

for row in rows:

t.add\_row([row.artist, row.song, row.length])

print(t)

**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.**

Nice work !  
Your notebook is well formatted. Another further improvement suggestion is that you can also add additional images and project overview to give more context about the project.

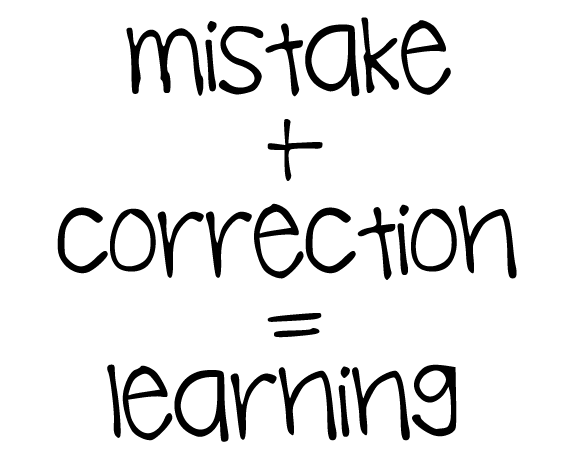
### Requires Changes

#### 5 SPECIFICATIONS REQUIRE CHANGES

Dear student,

Well done! This is a great attempt and you have done a great job so far. I believe you've learned a lot through this challenging project.

However, according to the rubric, there are some issues which you need to address to get a pass.

[](https://udacity-reviews-uploads.s3.us-west-2.amazonaws.com/_attachments/62193/1533399702/learn.png)

I'm sure you will pass in the next submission.

Looking forward to your next submission  Stay udacious and keep going!

### ETL Pipeline Processing

**Student creates event\_data\_new.csv file.**

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

Nice work! You are using the correct datatype, such as int for the itemInsession and the ids. You are also using the correct string datatype for the relevant columns as all of your INSERT statements are processing the data correctly.

### Data Modeling

**Student creates the correct Apache Cassandra tables for each of the three queries. The CREATE TABLE statement should include the appropriate table.**

Great job! You followed the one table per query rule of Apache Cassandra. You are not replicating the same table for all three queries, which defies that rule.

**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.**

The rubric requires that "Student demonstrates good understanding of data modeling by generating correct SELECT statements to generate the result being asked for in the question."

Here, instead of select \*, you should just select the information that is asked for.

**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.**

Nice! the table names reflect the query and the result it generates.

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

The sequence of the columns in the CREATE and INSERT statements should follow the order of the COMPOSITE PRIMARY KEY and CLUSTERING columns. The data should be inserted and retrieved in the same order as how the COMPOSITE PRIMARY KEY is set up.

Note that this is important because Apache Cassandra is a partition row store, which means the partition key determines which any particular row is stored on which node. In case of composite partition key, partitions are distributed across the nodes of the cluster and how they are chunked for write purposes. Any clustering column(s) would determine the order in which the data is sorted within the partition.

## Most of the queries have such issues in your implementation. Please double check each statement and make sure they are in the correct order (be consistent with the primary key).

Here are some DataStax documentation as your reference:  
<https://docs.datastax.com/en/dse/5.1/cql/cql/cql_using/whereClustering.html>

### 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.**

For query 2, since we are retrieving data by userid and sessionid, you should use both of them as your partition key.  
Here you could use PRIMARY KEY ((userId, sessionId), itemInSession) to uniquely identify each row.

**Explanation to the partition key** Using more than one column for the partition key (such as (userId, sessionId)) breaks the data into chunks, or buckets. Then, when we query with the partition key, it identifies a bunch of rows all of which have the same partition key.  
Here why do we use both? Because both of them are given, then use them as the partition key will make the identification process more direct, i.e. more efficient.

Here has a good discussion on these concepts. <https://stackoverflow.com/questions/24949676/difference-between-partition-key-composite-key-and-clustering-key-in-cassandra>

### Presentation

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

More explanations are needed to organize your thoughts on each question.  
Here you need to explain a little bit, how based on the query you have built the table. How the primary keys are meant to work?

**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.**

As required in the rubric. Any in-line comments that were clearly part of the project instructions should be removed so the notebook provides a professional look. Please make a clean-up such as removing those TODOs.