Code Logic: Online Advertising

1. **Executing the SQL queries**

   Created a database named **upgrad** and three tables namely ads, served_ads and users.
   CampaignID, target gender, and other information about the list of ads is included in the **ads table**, along with certain derived properties like **Status (Active or Inactive), CPM, and Current Slot Budget.**

   The **served_ads** table contains information on the ads that have been shown to the user, including **userID, campaign Id, CPM, CPC** and other parameters.

   The **user** table will be used to hold **user information**.
   The data to upload in to this table will be downloaded from https://de-capstone-project1.s3.amazonaws.com/users_500k.csv

   The query that is used to copy the data into the user table after it has been downloaded from the above-mentioned location is as follows.

   LOAD data local infile '~/users_500k.csv' \ INTO TABLE users \ fields terminated by '|' \ ignore 1 rows;

2. **Ad_Manager**

   The main function of the ad_manager is to process all types of events, including New, Update, and Stop campaign events, by reading ad campaign events from the Kafka central repository hosted by Upgrade.
   We need a mysql connector in the script to connect to the database and write the data to the advertisements table.
   Additionally, we require a pykafka consumer to read data from the de-capstone1 topic and the kafka queue that is hosted at IP 18.211.252.152:9092.
   Additional parameters like Status, Slot Budget for each campaign, and CPM are also derived by the script.

© Copyright 2020. upGrad Education Pvt. Ltd. All rights reserved

Command to execute ad_manager script :

python3 ad_manager.py 18.211.252.152:9092 de-capstone1 ec2-54-91-228-196.compute-1.amazonaws.com root 123 upgrad

3. **Ad_server**

Since it is in charge of delivering advertisements to client devices, this is the project's key component. When the user is online and active, Ad_server must send the request and user information. After receiving the request, the ad_server must retrieve the list of advertisements before conducting the Second-Price Auction. The winning ad is then posted or served to the user, and its details are recorded in the served_ads table.

Command to execute ad_server script :

python3 ad_server.py ec2-54-91-228-196.compute-1.amazonaws.com root 123 upgrad

4. **User Simulator**

The user simulator must interact with the ad_server's and the feedback handler's APIs as if they were real users. The API that receives advertising must be accessed first, followed by the feedback API (the feedback API will be created later). Additionally, it accesses the user table to retrieve the data.

Command to execute user_simulator script :

python3 user_simulator.py ec2-54-91-228-196.compute-1.amazonaws.com root 123 upgrad http 0.0.0.0 5000 0.0.0.0 8000

© Copyright 2020. upGrad Education Pvt. Ltd. All rights reserved