Data Engineer Applicant Assessment

Programming Exercise

Objective

This exercise is to help assess your level of understanding in Python, AWS services, Github and problem solving. The following exercise is a small example of a potential custom project the ACS Data Engineering team may engage in. We are interested in how effective you are in answering the question, the steps you took to answer the question and any additional insight to help the customer gain more value from this process.

Setup

Attached is a simple tab separated file which contains what we call "hit level data". A hit level record is a single "hit" from a visitor on the client's site. Based on the client's implementation, several variables can be set and sent to Adobe Analytics for deeper analysis.

Your exercise is to write a Python application that is capable of reading this hit level data file and answer the client's question. Ideally the expectation is for the candidate to use their own free tier AWS account to build and deploy this python code and deployment scripts, additional documentation and results within a public Github repository that needs to be shared as part of the exercise. Review Appendix A for more details on how the hit level data file is formatted

Development Exercise

The Client's Question (Business problem)

How much revenue is the client getting from external Search Engines, such as Google, Yahoo and MSN, and which keywords are performing the best based on revenue?

Development Requirements

1. Create a Python application that needs to be deployed and executed within AWS
2. The Python application needs to contain at least one class
3. The Python application needs to accept a single argument, which is the file that needs to be processed.

Bonus Points: Unit test cases, serverless deployment scripts, Business problem presentation

Deliverable Requirements

The final output needs to be a tab delimited file with the following data points:

- Search Engine Domain (i.e. google.com)
- Search Keyword (i.e. "Laffy Taffy")
- Revenue (i.e. 12.95)

The final output also has the following requirements:

- A header row needs to be included. Use the above bulleted items for each column header, minus the example.
- Sorted by revenue, descending, so the client can easily review which keyword is performing the best.
- The output file should have the following naming convention:

[Date]_SearchKeywordPerformance.tab
[Date] corresponds to the date the application executed for
The format should be YYYY-mm-dd (i.e. 2009-10-08)

Timeline

If you have any questions, you are encouraged to reach out to us for any clarity.

Verification

There will be a 60-minutes code review video call where, you will first walkthrough the business use case and how did you approach to solve this problem followed by executing the code to review results and finally review the code.

Other Considerations

Because our team deals with extremely large files, over 10 GB per file uncompressed. Give us your thoughts on how well this application will scale and if not, what improvements could be made to ensure it can scale to this file size.

Please consider this as a project you would develop for a client while you were working on the team. So, take into consideration good coding practices

Appendix A: Explanation of the Hit Level Data File

In order to achieve your objective, you will need to parse this file review the following columns:

Column Name	Column Description	Data Type
hit_time_gmt	The timestamp of the hit based in Unix time.	Int (11)
date_time	The time of the hit in readable format, based on the report suite's time zone.	datetime()
user_agent	User agent string sent in the HTTP header of the image request.	text
ip	IP Address based on the HTTP header of the image request.	Varchar (20)
geo_city	Name of the city the hit came from, based on IP	Varchar (32)
geo_country	Abbreviation of the country the hit came from, based on IP	Varchar (4)
geo_region	Name of the state or region the hit came from, based on IP	Varchar (32)
pagename	Used to populate the Pages dimension. If the pagename variable is empty, Analytics uses page_url instead.	Varchar (100)
page_url	The URL of the hit. Not used in link tracking image requests.	Varchar (255)
product_list	Product list as passed in through the products variable. Products are delimited by commas while individual product properties are delimited by semicolons. For details on how the product_list is formatted, please review Appendix B	text
referrer	The referring URL to the page the visitor is currently reviewing. Example: http://search.yahoo.com/search?p=marketing&sm=Yahoo%21+Search	Varchar (255)
event_list	A comma separated list of events that occur during the visit. Below is an explanation of the events. Example format: "2,200,201,100"	text

- 1: Purchase
- 2: Product view
- 10: Shopping Cart Open
- 11: Shopping Cart Checkout
- 12: Shopping Cart Add
- 13: Shopping Cart Remove
- 14: Shopping Cart View

Appendix B: Description - Product List variable

The products list is a comma (,) delimited list of products with a semi-colon (;) delimited list of attributes for each product

Example Format

[Category];{Product Name];[Number of Items];[Total Revenue];[Custom Event]|[Custom Event];[Merchandizing eVar],.

Real Example

"Computers;HP Pavillion;1;1000;200|201,Office Supplies;Red Folders;4;4.00;205|206|207"

Product Attribute Description

- Category: The category for the product (Shoes, Clothes)
- Product Name: Either the product ID or the product name
- Number of Items: The number of products
- Total Revenue: The price of the product. Revenue is only actualized when the purchase event is set in the events_list
- Custom Event: Events only applied to a specific product
- Merchandising eVar: eVars only applied to a specific product