ANTENNA Data Engineer Case Study

Objective

This document describes ANTENNA's technical case study for the Data Engineer position, a standard step in our recruiting process.

This is a software design problem that involves similar/same data and technology that we use at ANTENNA, designed to give you a sneak peek of what kind of data we process, and what kind of systems we build. This also allows us to see how you approach similar problems that we have, and how you make design and implementation choices.

Overview

At ANTENNA, transaction data is used to extract important insights on user level subscription behavior. An important step in such a process is preprocessing raw transaction data and turning them into canonicalized subscription events.

In this project, you are given a small data set that contains anonymised transactions with a number of merchants selling digital products, some of which are SVOD (subscription video on demand) services such as Premium plans on Netflix. We ask you to design and build a python pipeline that identifies key subscription events such as sign-up and cancellation. Such results could then be passed downstream to generate aggregated insights regarding subscriptions to each individual service, for example, how many people signed up for Hulu in September 2020.

Data

As part of this project you will receive a data table which looks like below.

item_id	buyer_id	order_date	merchant_id	merchant_name	status	last_updated	description
8,905,026,709	-1,199,890,392	2016-05-03	6	Netflix	N	2020-06-17 10:19	Thanks for coming back to Netflix!
8,978,265,304	-2,140,382,112	2016-06-27	6	Netflix	N	2020-06-29 09:07	Thanks for coming back to Netflix!
8,978,265,185	-2,140,382,112	2016-06-24	1,493	Google Play	N	2020-06-29 09:07	Monthly Subscription (HBO NOW) Monthly Subscription - Auto Renews on Jul 24, 2016
8,978,265,243	-2,140,382,112	2016-07-24	1,493	Google Play	N	2020-06-29 09:07	Monthly Subscription (HBO NOW) Monthly Subscription - Auto Renews on Aug 24, 2016

The description field, which is a string, contains signals about certain subscription actions. For example, the first row shows buyer -2140382112 started a HBO Now subscription on Google Play. Here “HBO Now” is the name of the service, and “Google Play”, which is in the merchant column, is the name of the merchant. The description tells us a HBO Now subscription on Google Play is started or renewed. The 2 key concepts, merchant and service, deserve some more explanation.

- **Merchant** = the party that distributes the service to the customer. A merchant is a distributor and can be a service. In the previous example, *Google Play* is the merchant, which distributes the service *HBO Now*. Merchant is given in the data already.
- **Service** = the actual provider of the content you watch. These are the channels of the new, streaming landscape. HBO Now, Showtime, et al. Services are distributed both via large distributors like Google Play and Amazon Channels but also directly (i.e. Netflix also distributes Netflix directly, in which case Netflix is both the merchant and the service). Services are not explicitly called out in the data and need to be identified from descriptions.

Matching Rules

The description needs to be processed so we can extract the service name. To limit the scope of this project, you are provided with a table with matching rules, describing how to identify various services from the description string. There are 3 types of matching patterns

1. S (start): if the pattern in *text_match* column is a prefix of the description string, a match is found for the corresponding service (both ID and name are included in the table)
2. A (anywhere): similar to above, except that the pattern doesn’t have to be in the beginning of the description
3. R (regular expression): use *text_match* as a regular regular expression

merchant_id	service_id	service_name	text_match	text_exclude	matching
1,493	1	Hulu	hulu		S
5	1	Hulu	Hulu		S
1,178	1	Hulu		Live TV	S
5	1	Hulu	Renewable Subscription iOS		S

By iterating through all the rules in this table, you can extract the service from the description. If nothing is found then you can simply ignore that transaction.

You still need to figure out what kind of action (signup versus cancellation) the remaining transactions are about. If there are trial signup and cancellation, please make your own judgment as to how to treat them.

Data Updates

You probably have already noticed that there are two columns pertaining to how a data entry is updated: *status* and *last_updated*. A transaction can be modified in the future when more information becomes available, or the process that generates those transactions changed. Column *last_updated* tells us when the row is entered into the data set, while column *status* tells us if a transaction with the same *item_id* already exists, how should that previous row be handled. There 3 types of statuses:

1. N (new): this transaction is new
2. U (update): this transaction is updated; discard the old one
3. D (delete): remove this transaction

When you process this data, keep in mind that you are simulating a recurring job that processes incremental data on a regular cadence.

Results

After this process, the data should include a few additional columns, which indicate the ID and name of the SVOD service, and the type of the action. For example, as shown below, buyer -2147454895 canceled their Netflix subscription on 10/30/2017, while buyer -2147449680 signed up for a Starz subscription on Amazon Channels on 5/22/2017.

item_id	buyer_id	order_date	merchant_id	merchant_name	service_id	service_name	signal_type	description
8,421,384,708	-2,147,454,895	2017-10-30	6	Netflix	3	Netflix	Cancel	we've cancelled your 1 screen at a time plan
6,163,461,876	-2,147,449,680	2017-05-22	3,258	Amazon Channels	8	Starz	Signup	Subscription to STARZ on Amazon Channels
7,837,913,609	-2,147,423,658	2016-12-29	1,873	The New York Times	42	New York Times	Signup	NYTimes Digital Subscription Type: Basic Digital Access
1,588,259,352	-2,147,423,102	2016-02-16	6	Netflix	3	Netflix	Cancel	we've cancelled your Netflix Account. This change will be effective on Monday, 7 March 2016.

To simplify the project, you only need to consider the following services, although the data may include many others.

- Netflix
- Hulu
- CBS All Access
- Starz
- Showtime

Deliverables

There are 2 sets of deliverables for this project

1, Code and Data

Assuming the input data is CSV, deliver the code that appends necessary columns to the input, only for the rows that are pertinent to the 5 services above. Other rows should be discarded.
Requirements

1. Use Python as the programming language
2. If you have a Github account, you can put your source code there. You can also email us your code.
3. Output data is accessible from a common cloud storage (i.e. AWS S3 or Google Storage)

2, Architecture Design

With the learning from processing this very small data set, outline a design for a scalable architecture in the cloud. Please discuss

1. How would you store this data, assuming the scale is on the order of 1 billion rows? Would you choose a relational database, MongoDB, Hive, flat files, or something else? Explain your reasoning.
2. What technology would you use to build a pipeline that executes such a process with high throughput, assuming we'd like to run such a process daily, with latency lower than 1 hour.

Presentation

Once you are ready, please let us know and we'll schedule a Zoom call for you to meet some of us on the ANTENNA team and present your work, walking through your code and design. Having a few slides will greatly help facilitate this discussion.