Teamviewer Data Engineer Challenge

Basic Data Ingestion Task Answer:

Freshdesk is a customer support and help desk software solution developed by Freshworks. It's a Ticketing System that is designed to help businesses manage their customer inquiries, issues, and support requests more efficiently.

As per question to get all of the information about tickets from Freshdesk API we need to call the following API endpoint with valid API token.

https://domain.freshdesk.com/api/v2/tickets

The above API endpoints give the list of all tickets that are created. To use this API endpoint very efficiently we have query parameters that we can use to get relevant information very efficiently.

We can provide the filter parameters that can be attached to API endpoint calls as a query parameter and based on that we can receive the relevant information. Some filters that I would use while calling this API endpoint as follows.

Assumption : We already have one time dump from this API endpoint.

- *https://domain.freshdesk.com/api/v2/tickets?company_id=[id]*

  - Using this filter as a query parameter we can only ask to provide the data for specific company id. This query parameter helps when one company has different child companies and we need to retrieve the ticket data for one specific company but not for all.

- *https://domain.freshdesk.com/api/v2/tickets?updated_since={date}*

  - This filter allows us to load data incrementally. As per above assumption, when we try to load the data a second time we do not want to load data again that we already have with us, so we only load data from a specific date(ideally t-1). So for ex. If we received the first dump on 1nd Oct 2023 at 23:59:59 then we can set the parameter like below,

    https://domain.freshdesk.com/api/v2/tickets?updated_since=2023-10-02T00:00:00Z

    Here we are asking to send the data from 2023-10-02T00:00:00Z so we received the incremental data for one day.

Q 3.1] Which technologies would you use?

- **Python** as a programming language component to utilise Freshdesk's API to compile a list of tickets, download necessary ticket information, apply specific filters to isolate data pertinent to your company, and limit it to the previous day
- **Airflow** to schedule the execution of both Python and DBT components at predefined intervals and in a specific order using DAG(Directed Acyclic Graph).
- **Snowflake** or **Redshift** as a data warehouse to store the data coming from API endpoints as well as transformed data, using tables and views.
- Generating **oAuth token** or **API token** as an Authentication and authorization method to validate the API call.
- To protect the sensitive credentials and API keys using the combination of AWS solutions, including **Secrets Manager** (SM) and **Key Management Service** (KMS).
- **Data Build Tool** (DBT) components to read data from existing raw tables, to perform necessary data transformation as business logic and stored the transformed data in production ready target tables, which can be accessible to various departments in the company and tools like Tableau.
- **Tableau** component to create various dashboards for data visualisation, including the tiles that give you insights about monitoring and performance evaluation of each customer service agent's based on their ticket-related activities..

  Below is the overall architecture for tools and methodologies that can interact with each other to provide the Extraction, Load and transform of the raw data to clean to data and visualisation of the clean data.

  LINK

Q 3.2] How would the process be deployed and updated?

- We need to create the DAG(Directed Acyclic Graph) which has python code to retrieve the data from the API.
- We need to schedule this via CRON expression and then airflow will run this DAG as per scheduled.
- Once we merge the code, Airflow which runs on the EC2 instance(for example) will pick the changes and run the DAG as per scheduled time.

Q 3.3] Where would you store the sensitive information such as access credentials?

- We can use the AWS Key Management System (KMS) and Secrets Manager (SM) to protect our credentials.

Q 3.4] How would you structure the data?

- When we fetch the data from *https://domain.freshdesk.com/api/v2/tickets* API endpoint we will get all the data. Of Course we can add some query parameters to it. When we do receive the data we receive the following data points in terms of JSON response.


```
[
  {
    "cc_emails" : ["user@cc.com", "user2@cc.com"],
    "fwd_emails" : [ ],
    "reply_cc_emails" : ["user@cc.com", "user2@cc.com"],
    "fr_escalated" : false,
    "spam" : false,
    "email_config_id" : null,
    "group_id" : 2,
    "priority" : 1,
    "requester_id" : 5,
    "responder_id" : 1,
    "source" : 2,
    "status" : 2,
    "subject" : "Please help",
    "to_emails" : null,
    "product_id" : null,
    "id" : 18,
    "type" : Lead,
    "created_at" : "2015-08-17T12:02:50Z",
    "updated_at" : "2015-08-17T12:02:51Z",
    "due_by" : "2015-08-20T11:30:00Z",
    "fr_due_by" : "2015-08-18T11:30:00Z",
    "is_escalated" : false,
    "custom_fields" : {
      "category" : "Default"
    }
  },
]
```


- To measure the efficiency of support agents we do not always require all the columns that we are getting via API call. In order to get the information about agents efficiency I took the following columns and created the schema for it.

Column Name	Data Type
responder_id	varchar
requester_id	varchar
Id (PK)	varchar
product_id	varchar
source	int
status	int
priority	int
created_at	date
updated_at	date
is_escalted	boolean

- The overall data structure/data modelling to evaluate efficiency of support agents is as follows.

  LINK

Q 3.5] How would you report on it?

- I would connect the production ready tables to Tableau visualisation tool to create dashboards.
- By creating views and calculating specific metrics in Tableau we can also define aggregate measures which gives additional insights about current use cases.
- This can give the answers to some questions like,

  1. Which agent solves the tickets in a particular date range?
  2. Which agent takes the average longest duration to solve the tickets?
  3. Average duration for each agent to solve the tickets.
  4. From which source the highest ticket was created?