Interview Assignment - Data Engineering

Thank you for taking the time to do our technical assignment, we really appreciate it.

In this repository you can find the infrastructure setup and instructions necessary to solve the assignment described under The Assignment section.

The purpose of this assignment is to show your data engineering and coding skills to assess the problem solving skill set, creativity, and demonstrate your experience with the tools which you choose.

It is important to convince your future teammates with your competences in:

- Python: please code as if you were writing production code.
- Data structures and algorithms
- Distributed data processing frameworks
- Git
- (optional) SQL
- An insightful, readable, well-structured README.md; We don't expect extensive/fancy documentation or diagrams, but we would like to have an overview of what you did and why you did it.

We try and make sure that the docker compose file works as expected, but it could be that you run into some issues. If you do, please feel free to report them (or fix them) so we can improve the experience.

Existing state and setup

What can you find in this repo?

To make solving the assignment time efficient, we are providing you with a docker compose file that sets up all the infrastructure necessary to complete the assignment as well as some startup files for various ways of working with Spark. You can chose one between the notebook or the PySpark app to solve the assignment.

In the provided infrastructure, you will find a Hive catalog to store the table metadata, MinIO as a replacement for S3, Trino to run interactive SQL queries, Spark for the ETL and Jupyter notebooks as the IDE to build your solution.

Spark is set up to use the Hive catalog and it already has support for Iceberg tables, so you don't have to mess around with setting those up.

> Note: This entire setup needs reasonable amount of memory. If you are running a Mac and have Colima (or Docker Desktop or something similar) installed, make sure that the VM use by Docker has at least 8GB of memory.

- We generated historical_orders.json and recent_orders.json files for you to use in the assignment.
  - The historical_orders.json file contains ~54k orders from the last 2 years.
  - The recent_orders.json file contains 1422 orders from the last 24 hours.
- After you run `docker-compose up` in your terminal, you should be able to access the following:
  - Trino is accessible on `http://localhost:9090` (for SQL queries)
  - MinIO is accessible on `http://localhost:9001` (for storage, S3-like)
  - For Python:
    - Notebooks are accessible on `http://localhost:8888` and are stored in `./notebooks` (to code and/or test your solution)
    - You can find an empty project and some helper scripts in `./spark-jobs-python`. (to code and/or test your solution)
  - Other URLs are available in the docker compose file.

Data and buckets

Assignment requires some initial data for you to be able to work on it effectively. In the `data` folder, you can find the demo data that will be placed in the `demo-data` bucket in minio. In the same folder you can find `copy_to_minio.sh` script. That script will run on startup and create the following buckets:

- demo-data
- dal
- temp

Feel free to change the script and add any buckets you need for your solution.

> Note: DAL is a shortcut for Data Access Layer. We use it in various places.

Databases and tables

Hive comes preconfigured with iceberg connector and it exposes a single catalog called `dal`. If you want, you can change the containers and use some other table format. We leave that up to you. It is not necessary for the solution to be complete.

The Assignment

You have to develop a **data engineering solution** that will enable you to answer the following question:

> Which top 3 industries are showing biggest change in the last 24 hours compare to the past 30 days average?

To answer that you need to work with a fix set of orders (historical and recent), a batch data set of customers, and a value object of industries.

Orders (one-off)	[order_id, customer_id, order_line[product_id, volume, price], amount, timestamp]
Customers (batch)	[customer_id, company_name, specialised_industries]
Industries	Agriculture, Colours, Cleaning, Construction, Cosmetics, Food, Lubricants, Oil&Gas, Pharmaceuticals, Polymer

Your solution should:

- Define databases and tables for the data being ingested and processed.
- Provide a solution that can be run on triggered/scheduled basis to answer the question.
- Provide the same answer if run on the same time window multiple times (i.e. be idempotent).
- Provide an answer to the question above.
- Always provide some notes to show us your reasoning!

Notes

👉 Assume that Customers data set is coming in batch with any arbitrary frequency (e.g. every 12 hours).
👉 A customer can be specialised in multiple industries (e.g. Cosmetics & Pharmaceuticals).
🎯 Based on the aforementioned data sets, you should be able to calculate which top 3 industries fluctuated the most (whether positive or negative) in the last 24 hours of orders comparing to the historical data of the last 30 days.

We have provided you with an environment to do the assignment, but you are free to change any part of it. If you do change it, we would like to know what and why!

Considerations

1. When loading data from batch data sets, make sure you have de-duplication logic in place.
    i. To create a table in the `dal`, remember that you need to reference the full path, e.g. `dal.brenntag.customers` (brenntag is the database in this case)
2. Think about what and how to test.
3. To spin up and down the environmnet, please use `docker-compose up` and `docker-compose down + docker-compose down --volumes`

4. Please create tables for the data sets. Have a look at the notebook to see how to create a table and a database.
5. Please keep the code in the private repo in GitHub/GitLab/Bitbucket and provide access to the reviewers or alternatively send us a zip of your local Git repo.

Have fun and success!