

10 Academy Cohort B - Weekly Challenge: Week 2

## Data Engineering: Data warehouse tech stack with MySQL/PostgreSQL, DBT, Airflow

# Overview

Business Need

You and your colleagues have joined to create an AI startup that deploys sensors to businesses, collects data from all activities in a business - people’s interaction, traffic flows, smart appliances installed in a company. Your startup helps organisations obtain critical intelligence based on public and private data they collect and organise.

A city traffic department wants to collect traffic data using swarm UAVs (drones) from a number of locations in the city and use the data collected for improving traffic flow in the city and for a number of other undisclosed projects. Your startup is responsible for creating a scalable data warehouse that will host the vehicle trajectory data extracted by analysing footage taken by swarm drones and static roadside cameras.

The data warehouse should take into account future needs, organise data such that a number of downstream projects query the data efficiently. You should use the Extract Load Transform (ELT) framework using DBT. Unlike the Extract, Transform, Load (ETL), the ELT framework helps analytic engineers in the city traffic department setup transformation workflows on a need basis.

# Data

In [Downloads – pNEUMA | open-traffic (epfl.ch)](https://open-traffic.epfl.ch/index.php/downloads/#1599047632450-ebe509c8-1330) or from <https://zenodo.org/records/7426506> you can find a pNEUMA data: pNEUMA is an open large-scale dataset of naturalistic trajectories of half a million vehicles that have been collected by a one-of-a-kind experiment by a swarm of drones in the congested downtown area of Athens, Greece. Each file for a single (area, date, time) is ~87MB data.

You may refer to the following references to understand how the data is generated from video frames recorded with swarm drones.

* [PIA15\_poster.pdf (datafromsky.com)](https://datafromsky.com/wp-content/uploads/2015/03/PIA15_poster.pdf)
* [(PDF) Automatic vehicle trajectory extraction for traffic analysis from aerial video data (researchgate.net)](https://www.researchgate.net/publication/276857533_Automatic_vehicle_trajectory_extraction_for_traffic_analysis_from_aerial_video_data)

You may use the following github packages to visualise and interact with the data (and obtain other similar data)

* [tud-hri/travia: a Traffic data Visualization and Annotation tool (github.com)](https://github.com/tud-hri/travia)
* [JoachimLandtmeters/pNEUMA\_mastersproject: Written python files to work with pNEUMA dataset (github.com)](https://github.com/JoachimLandtmeters/pNEUMA_mastersproject)

# Expected Outcomes

Skills:

#### Create and maintain Airflow DAGs

#### Work with Apache Airflow, dbt, redash and a DWH

#### Apply ELT techniques to DWH

* Build data pipelines and orchestration workflows

Knowledge:

* Enterprise-grade data engineering - using Apache and Databricks tools

# Team

Instructors:

* Yabebal
* Emtinan
* Rehmet

# Key Dates

* **Discussion on the case** - 09:00 UTC time on Monday 29 Apr 2024. Use #all-week2 to ask questions.
* **Interim Submission** - 8:00 PM UTC time on Wednesday 01 May 2024.
* **Final Submission** - 8:00 PM UTC time on Saturday 04 May 2024.

# Leaderboard for the week

There are 100 points available for the week.

* 20 points - community growth and peer support.
* 30 points - presentation and reporting.
  + 15 points - interim submission. PDF slide or report format.
  + 15 points for the final submission. Blog entry or PDF with 5-8 pages.
* 50 points - Technical content
  + 20 points - Interim submission
  + 30 points - Final submission

Badges

Each week, one user will be awarded one of the badges below for the best performance in the category below.

In addition to being the badge holder for that badge, each badge winner will get +20 points to the overall score.

**Visualization** - the quality of visualizations, understandability, skimmability, choice of visualization

**Quality of code** - reliability, maintainability, efficiency, commenting - in the future this will be [CICD](https://en.wikipedia.org/wiki/CI/CD)

**An innovative approach to analysis** -using latest algorithms, adding in research paper content and other innovative approaches

**Writing and presentation** - clarity of written outputs, clarity of slides, overall production value

**Most supportive in the community** - helping others, adding links, tutoring those struggling

The goal of this approach is to support and reward expertise in different parts of the Machine learning engineering toolbox.

# Late Submission Policy

Our goal is to prepare successful learners for the work and submitting late when given enough notice, shouldn’t be necessary.

For interim submissions, those submitted 1-6 hours late will receive a maximum of 50% of the total possible grade. Those submitted >6 hours late may receive feedback, but will not receive a grade.

For final submissions, those submitted 1-24 hours late, will receive a maximum of 50% of the total possible grade. Those submitted >24 hours late may receive feedback, but will not receive a grade.

# Instructions

The fundamental tasks in this week’s challenge are the following - building data warehouse techstack

* Consisting of
  + A “data warehouse” (PostgreSQL)
  + An orchestration service (Airflow)
  + An ELT tool ([dbt](https://www.getdbt.com/))
  + A reporting environment ([redash](https://redash.io/))
* Set it up locally using
  + fully dockerized

Complete the following tasks:

1. Create a DAG in Airflow that uses the bash/python operator to load [the data files](https://open-traffic.epfl.ch/index.php/downloads/#1599047632450-ebe509c8-1330) into your database. Think about a useful separation of Prod, Dev and Staging
2. Connect dbt with your DWH and write transformations codes for the data you can execute via the Bash or Python operator in Airflow. Write proper documentation for your data models and access the dbt docs UI for presentation.
3. Check additional modules of dbt that can support you with data quality monitoring (e.g. great\_expectations, dbt\_expectations or re-data).
4. Connect the reporting environment and create a dashboard out of this data
5. Write a short article about your approach and what were the most important decisions along the way

Consider the following elements when doing the above tasks

* AIRFLOW:
  + If you want to use templates in Airflow, what is a good way to manage metadata and variables within your DAGS? (read about context)
  + Automated Alerting - what happens if the DAG is failing, e.g. a slack or email alert
  + Build hard circuit breaker pipelines with dbt (e.g. if a test fails, do not update the production tables)
* dbt
  + Automate the generation of dbt docs and make it available via web frontend
  + Explore macros and write your own to create dynamic documentation and functions
  + Automate the dbt to Airflow connection by automatically creating DAGS out of dbt metadata (see here:<https://www.astronomer.io/blog/airflow-dbt-2>)
* Redash
  + Build a version control script system by hitting the API, download the queries and built an automated git storage process

# Tutorials Schedule

In the following, the colour **purple** indicates morning sessions, and **blue** indicates afternoon sessions.

## Monday: Introduction to Week Challenge

Here the trainees will understand the week’s challenge.

* Introduction to Week Challenge (Yabebal)
* Data Models (kedro, data vault, star schema, database normalisation) (Emtinan)

Key Performance Indicators:

* Understanding week’s challenge
* Understanding Data Warehousing
* Ability to reuse previous knowledge
* Sharing references and content around data warehousing
* Getting familiar with data models and frameworks

## Tuesday: dbt Orchestration with Airflow

Here the trainees will understand Airflow and how to use it to schedule and orchestrate dbt.

* Analytics Engineering using DBT (Emtinan)
* Scheduling and Orchestration using Airflow (Rehmet)

Key Performance Indicators:

* Understanding directed acyclic graphs (DAGs) in Airflow
* Using Airflow as a scheduler to orchestrate dbt
* Sharing references and content around dbt and airflow

## Wednesday: From Data Lakehouse to BI Dashboards

Here the trainees will understand the different data warehouse and data lake tech stacks

* Kedro Data Layers - Data Lakehouse (Dibora)
* Building Dashboards using Redash (Abel)

Key Performance Indicators:

* Understanding Data Warehouse, Data Lake, and Data Warehouse
* Understanding redash
* Sharing references and content around redash and data warehousing concepts

## Thursday: Data Models and Tools

Here the students will understand Redash.

* Using DBT and Tableau/Looker/Microsoft BI to build BI Dashboard
* Conversation with the Data Engineering team (Betty et al.) - Adludio Team
* AWS Cloud Ecosystem for Data Engineering (Nabil et al.)

Key Performance Indicators:

* Introduction to AWS S3, Redshift, RDS, Athena, Glue
* Understanding Amazon Athena (managed presto) & Amazon Glue (managed HIve)
* Sharing references and content around Snowflakes, Databricks Data Lakes, Google Big Query, Amazon Redshift and other data warehousing and data lake technologies.
* Learning the data engineering tools ecosystem
* Sharing references and content around redash and data warehousing concepts

## TBD: Data models, tools, and frameworks

* ELT vs ETL
* Analytics Engineering with DBT
* Data Models for scalable data warehouse
* Data Lakes vs Data Warehouse: Tools and principles
  + Snowflake
  + Amazon Athena (managed presto) & Amazon Glue (managed HIve)
  + Databricks
  + Google BigQuery
  + Amazon Redshift

Key Performance Indicators:

* Learning the data engineering tools ecosystem
* Getting familiar with data models and frameworks

# Submission

## Interim: Due Wednesday 01 May 20:00 UTC

1. Link to your code in GitHub. You should have a screenshot folder that includes a screenshot of your data lineage from dbt
2. Submit a two pages max document that shows the tech-stack flow diagram, and explanation of the different elements

## Final Due Saturday 04 May 20:00 UTC

1. Link to your code in GitHub. Your readme should contain the link to your deployed dbt data warehouse documentation. You should also have a screenshot folder that includes a screenshot of the data view you built
2. A blog or report explaining the process you followed to build the tech stack. What are the challenges? What can be improved with more time?

### Feedback

You will receive comments/feedback in addition to a grade.

References

**Data Warehouse, Data Models, and Data Tools**

* [Data Warehouse Testing 101 | Panoply](https://panoply.io/data-warehouse-guide/data-warehouse-testing-101/)
* [Building a Data Vault (matillion.com)](https://documentation.matillion.com/docs/2775397)
* [DataHub Open Source Metadata Platform](https://datahubproject.io/)
* [Modern Data Warehouses: Functions, Architecture, & Examples | Estuary](https://estuary.dev/modern-data-warehouse/#:~:text=Modern%20data%20warehouses%20integrate%20and%20consolidate%20data%20from,then%20cleaned%20and%20organized%20into%20a%20unified%20repository.)

**dbt:**

General Information:

1. Installing dbt <https://docs.getdbt.com/dbt-cli/installation/#pip>
2. [dbt(Data Build Tool) Tutorial · Start Data Engineering](https://www.startdataengineering.com/post/dbt-data-build-tool-tutorial/)
3. [dbt - Package hub (getdbt.com)](https://hub.getdbt.com/elementary-data/elementary/latest/)
4. Introduction Videos on dbt <https://www.youtube.com/playlist?list=PLy4OcwImJzBLJzLYxpxaPUmCWp8j1esvT>
5. Redshift config; <https://docs.getdbt.com/reference/resource-configs/redshift-configs/>
6. Docs from Gitlab: <https://about.gitlab.com/handbook/business-ops/data-team/platform/dbt-guide/>
7. CLI command reference: <https://docs.getdbt.com/reference/dbt-commands/>
8. Basic Introduction to dbt https://www.kdnuggets.com/2021/07/dbt-data-transformation-tutorial.html

Articles:

1. <https://medium.com/the-telegraph-engineering/dbt-a-new-way-to-handle-data-transformation-at-the-telegraph-868ce3964eb4>
2. <https://medium.com/hashmapinc/dont-do-analytics-engineering-in-snowflake-until-you-read-this-hint-dbt-bdd527fa1795>

Repo Examples:

1. <https://github.com/mattermost/mattermost-data-warehouse>
2. <https://gitlab.com/gitlab-data/analytics/-/tree/master/>

How to structure repo's

1. <https://discourse.getdbt.com/t/how-we-structure-our-dbt-projects/355>
2. <https://discourse.getdbt.com/t/should-i-have-an-organisation-wide-project-a-monorepo-or-should-each-work-flow-have-their-own/666/2>
3. <https://discourse.getdbt.com/t/how-to-configure-your-dbt-repository-one-or-many/2121>
4. <https://medium.com/photobox-technology-product-and-design/practical-tips-to-get-the-best-out-of-data-building-tool-dbt-part-1-8cfa21ef97c5>

**Airflow:**

1. h[ttps://livebook.manning.com/book/data-pipelines-with-apache-airflow/chapter-1/v-6](https://livebook.manning.com/book/data-pipelines-with-apache-airflow/chapter-1/v-6)
2. <https://www.linkedin.com/in/marclamberti/>

**Docker:**

1. <https://www.youtube.com/watch?v=fqMOX6JJhGo>
2. <https://docker-curriculum.com/#docker-images>

**Redash:**

1. <https://github.com/dwyl/learn-redash>
2. <https://fitdevops.in/how-to-setup-redash-dashboard-on-ubuntu>

**Virtual environments:**

1. <https://www.ianmaddaus.com/post/manage-multiple-versions-python-mac/>
2. <https://www.codeblocq.com/2016/01/Search-through-history-in-OSX-terminal/>
3. <https://janakiev.com/blog/jupyter-virtual-envs/>
4. <https://medium.com/@blessedmarcel1/how-to-install-jupyter-notebook-on-mac-using-homebrew-528c39fd530f>