Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное
учреждение высшего образования
«Самарский национальный исследовательский университет имени академика С.П. Королева»
Институт информатики, математики и электроники Факультет информатики
Кафедра технической кибернетики

Отчет по лабораторной работе №1
по дисциплине «Инженерия данных»

Тема: «Знакомство с основными инструментами построения пайплайнов: Apache Airflow и Apache Nifi»

Направление подготовки: 01.04.02 Прикладная математика и информатика: Магистерская программа «Наука о данных»

Профиль «Системы искусственного интеллекта»

Выполнила Шаина М. М.,
студентка группы 6231 – 010402D

Самара 2023

В рамках данной лабораторной работы предлагается построить простейший пайплайн (рисунок 1), собирающий воедино данные из нескольких файлов, обрабатывающий их и сохраняющий результат в no-sql базу данных.

Цель лабораторной работы – знакомство с основными инструментами построения пайплайнов: Apache Airflow и Apache Nifi.

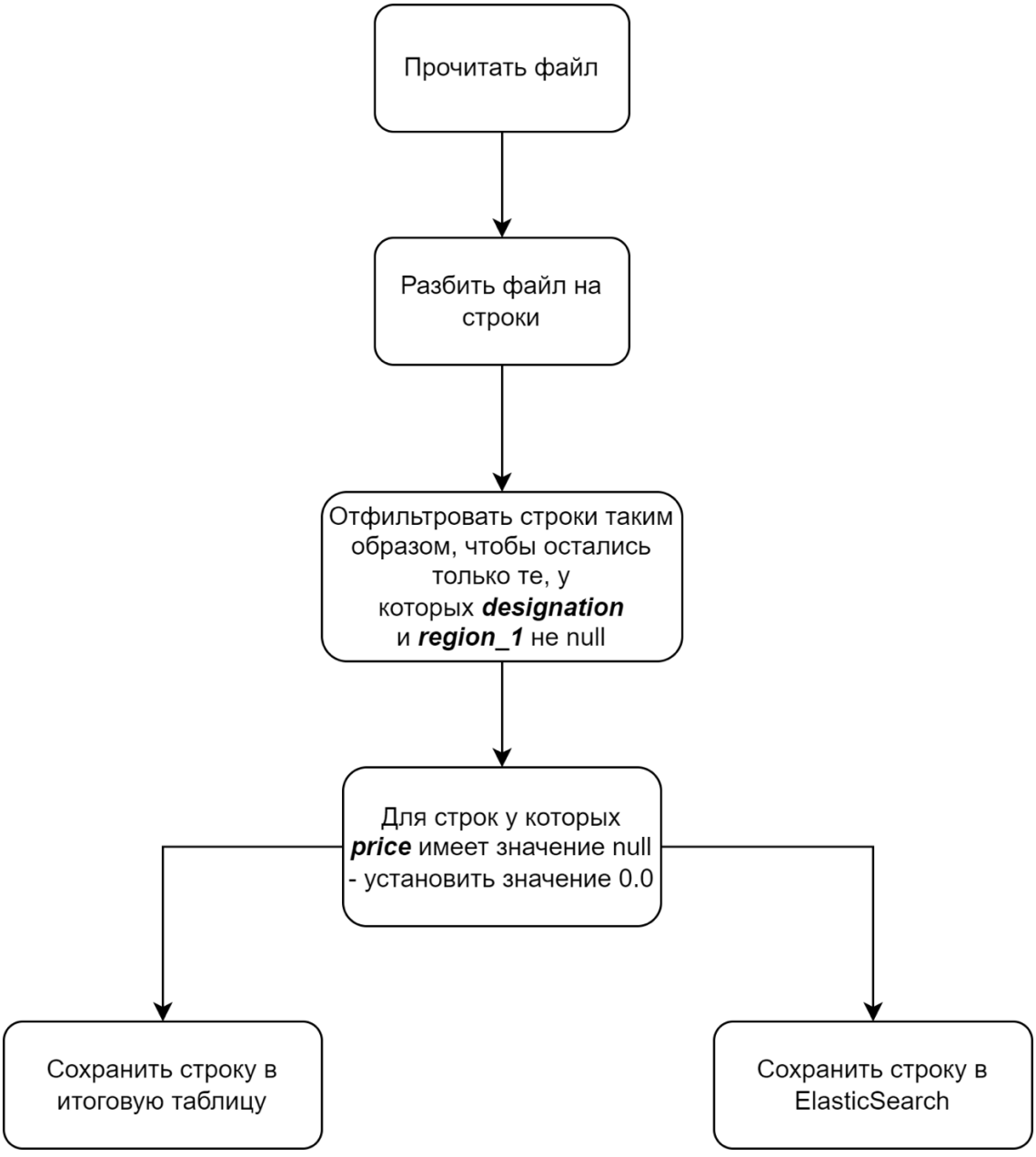

Рисунок 1 — Схема описывающая пайплайн, который необходимо построить в рамках лабораторной работы

Также необходимо построить гистограмму стоимости напитка к баллам средствами Kibana.

Для построения пайплайна рассматриваются инструменты, такие как ElasticSearch, Kibana, Apache Airflow и Apache Nifi.

В качествен данных - будем использовать набор из нескольких CSV файлов, полученных из набора данных wine-review.

Описание основных шагов:

1. Настройка программ для выполнения лабораторной работы.
2. Выполнения пайплайна в Apache Nifi.
3. Выполнение пайплайна в Apache Airflow.

Работа выполнялась в Virtual Studio Code со следующими расширениями:

1. ms-python.python
2. ms-toolsai.jupyter
3. ms-vscode-remote.vscode-remote-extensionpack
4. ms-azuretools.vscode-docker

1 Выполнения пайплайна в Apache Nifi

Перед начало работы в Apache Nifi csv файлы необходимо перенести в папку, которой Nifi имеет доступ, например, в папку `nifi/data/lab_1/input` (рисунок 3, см. ниже).

Для построения пайплайна в Apache Nifi использовались следующие процессоры:

1. GetFile
2. SplitRecord
3. QueryRecord
4. UpdateRecord
5. MergeContent
6. PutFile
7. PutElasticsearcRecord

Более подробное описание процессоров приведено в таблице 1.

Таблица 1 — Описание процессоров Apache Nifi

Processor	**Name**	**Property**	**Value**
GetFile	GetFile	Input Directory	/opt/nifi/nifi-current/data/lab_1/input
		File Filter	[^\.].*
SplitRecord	SplitRecord	Record Reader	CSVReader
		Record Writer	CSVRecordSetWriter
		Record Per Split	100 000
QueryRecord	Drop designation&region_1 IsNull Filter	isnull	SELECT * FROM FLOWFILE WHERE designation IS NOT NULL AND region_1 IS NOT NULL
UpdateRecord	Update price FromNulltoZero	/price	${field.value:ifEmpty('0.0'):toNumber()}
MergeConten	MergeConten	Delimiter Strategy	Text
		Header	id,country,description,designation…
PutFile	PutFile	Output Directory	/opt/nifi/nifi-current/data/lab_1/output

Для объединения данных из разных ветвей в Funnel выбраны два процессора: PutFile и PutElasticsearchRecord . Полная схема, отображающая пайплайн в Apache Nifi, представлена на рисунке 2.

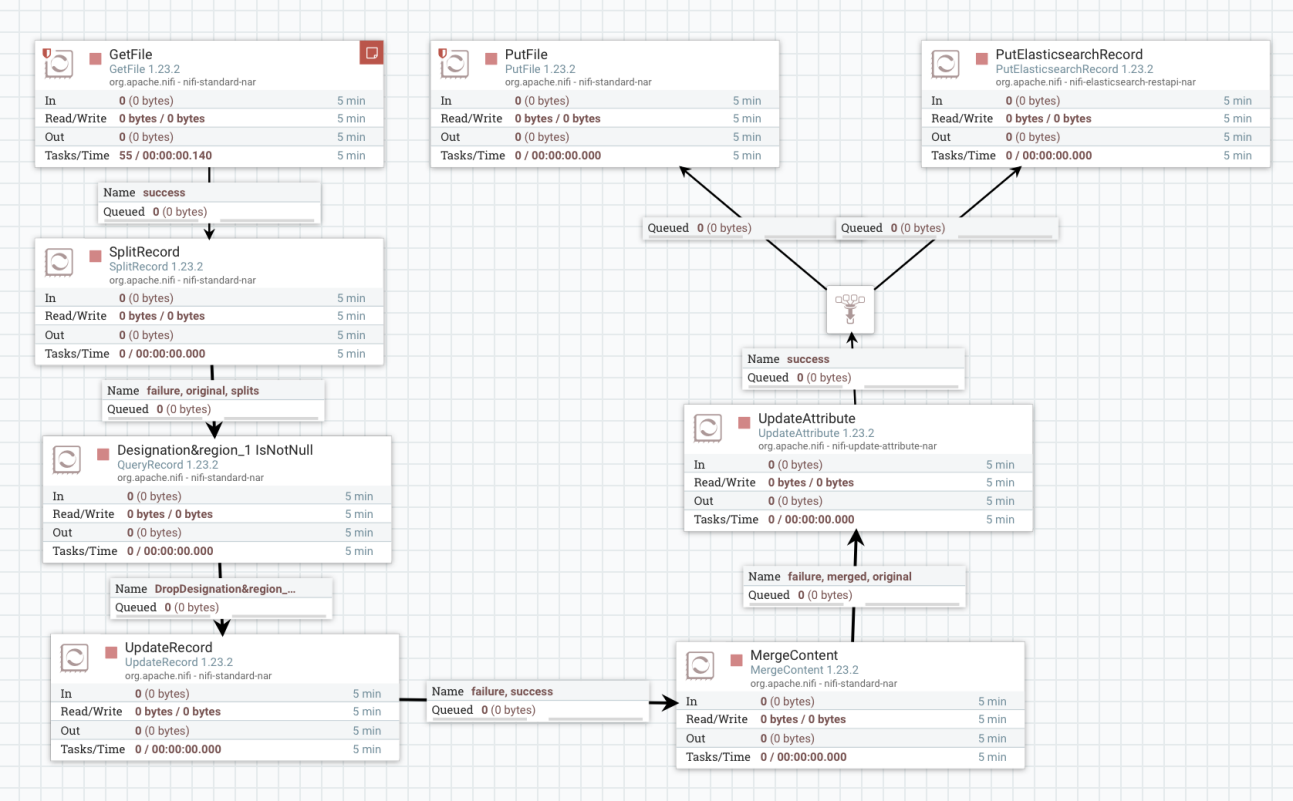

Рисунок 2 — Граф в Apache Nifi

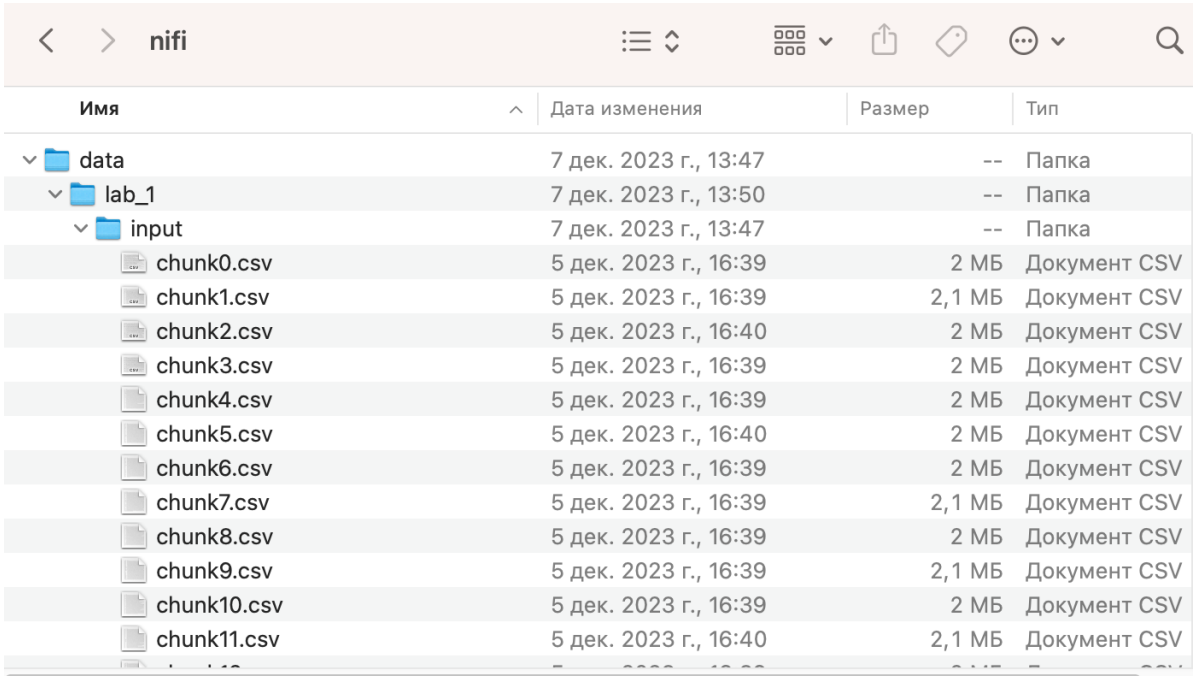

Рисунок 3 — Результат клонирования CSV файлов на локальный компьютер из репозитория Lab-1

Рассмотрим процесс построения DAGa в Apach Nifi более детально. На рисунках ниже изображены параметры процессоров, которые были использованы для построения DAGa. Первом шагом в построении пайплайна является чтение файла. Для этого воспользуемся процессором GetFile (рисунок 5, см. ниже).

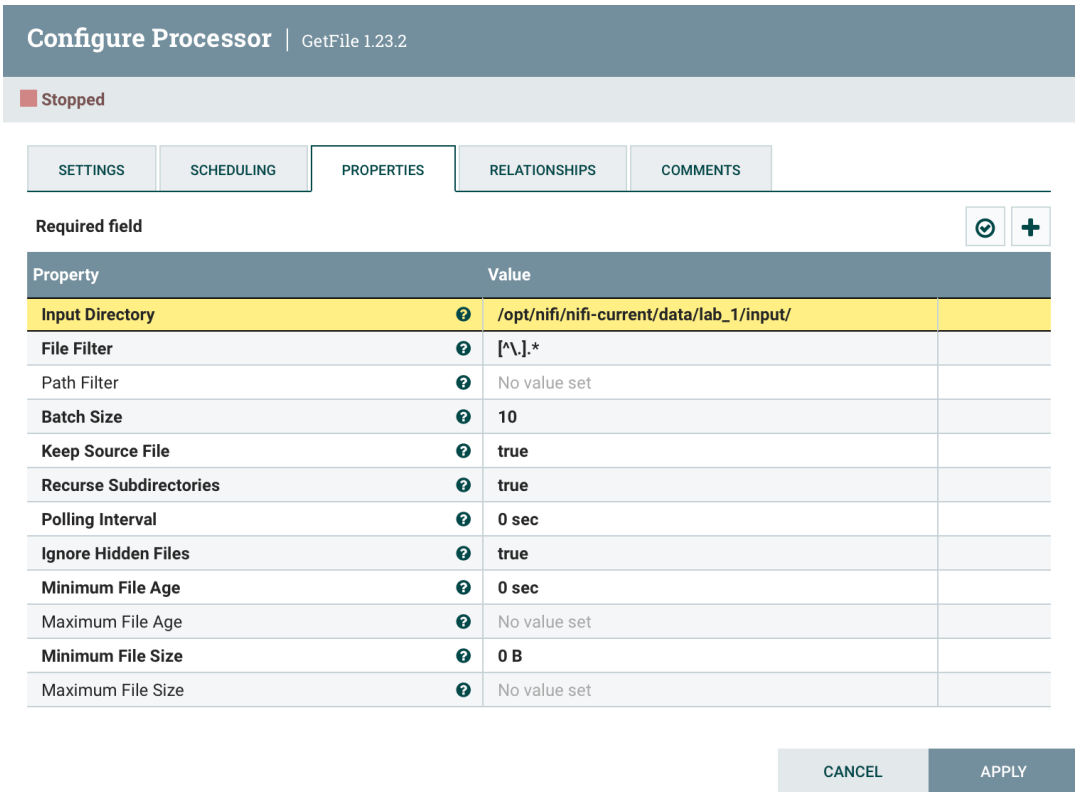

Рисунок 4 — Параметры процессора GetFile

Свойство Keep Source File должно быть установлено в значении True, что файл не стёрся сразу после обработки.

Для разделения на строки воспользуемся процессором SplitRecord. Для успешной работы потока данных необходимо установить CSVReader и CSVRecordSetWriter в статус Enabled (рисунок 5).

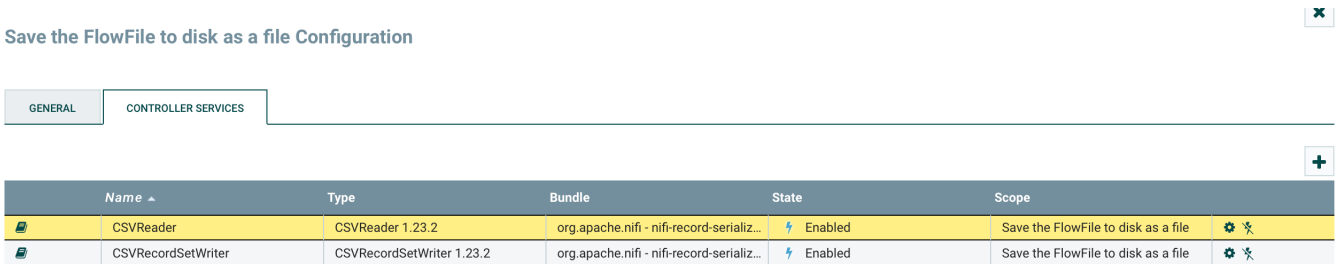

Рисунок 5 —Настройки параметров процессора SplitRecord

Теперь мы можем обрабатывать каждую строку с помощью процессора QueryRecord (рисунок 6). С помощью SQL запроса оставляем только те строки, которые не содержат пустые значения в столбцах designation и region_1.

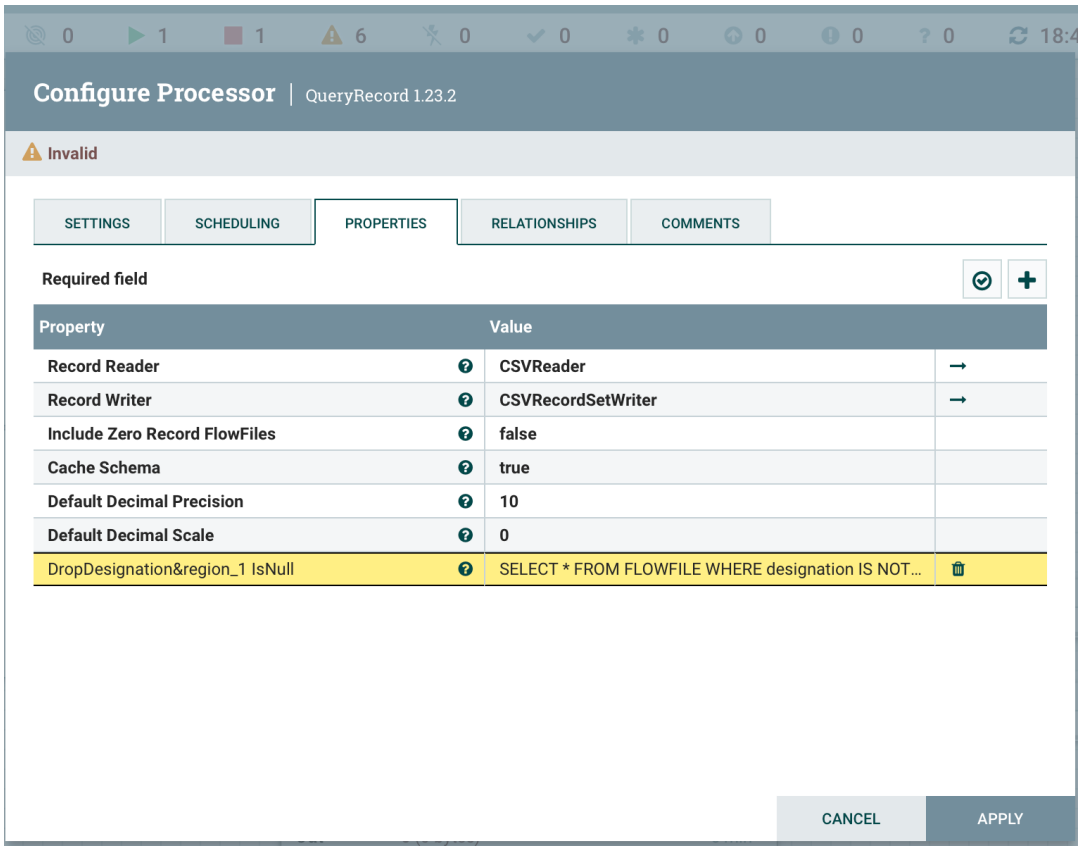

Рисунок 6 —Настройки параметров процессора QueryRecord

QueryRecord предназначен для выполнения запросов к данным, но он не дает возможности изменить содержание строки. Поэтому, на следующем шаге, воспользуемся процессором UpdateRecord.

2 Выполнения пайплайна в Apache Airflow

Перед началом работы необходимо авторизироваться, с предоставленным логином и паролем: http://localhost:18080/. После выполнения этого шага можно начать работу.

Используя Directed acyclic graph (DAG), реализуем пайплан для обработки и индексации CSV-файлов в Elasticsearch в Apache Airflow. Основная обработка в Python – data_processing: удаление строк с пропущенными значениями в столбцах «designation» и «region_1», а также заполнение пропущенных значений в столбце «price» нулями. Код программы на языке Python представлен в файле airflow-lab1.py. Граф внутри Apache Airflow представлен на рисунке 10.

Для видимости данных внутри пайплайна поместим файлы в директорию Prerequisites\airflow\data. Для этого клонируем репозиторий с первой лабораторной работой, в котором находятся необходимые csv файлы (рисунок 7). Копируем фалы в папку командой cp -r /путь/к/исходной/папке/* / Prerequisites/airflow/data. Результат отображения файлов представлен на рисунке 8.

```
[root@2245699-hf08008:/home/masha# git clone https://github.com/ssau-data-enginee]
ring/Lab-1.git
Cloning into 'Lab-1'...
remote: Enumerating objects: 60, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 60 (delta 8), reused 4 (delta 4), pack-reused 46
Receiving objects: 100% (60/60), 16.33 MiB | 10.15 MiB/s, done.
Resolving deltas: 100% (8/8), done.
```

Рисунок 7 — Клонирование репозитория с первой лабораторной работой

Для того, чтобы продолжить работу в Apache Airflow, необходимо поместить наш DAG в папку dags текущего репозитория. Это можно сделать командной строкой: cp /путь/к папке/airflow_lab1.py /путь/Prerequisites/airflow/dags.

После выполнения данной команды мы увидим наш DAG в разделе DAGs Apache Airflow и сможем запустить его (рисунок 9).

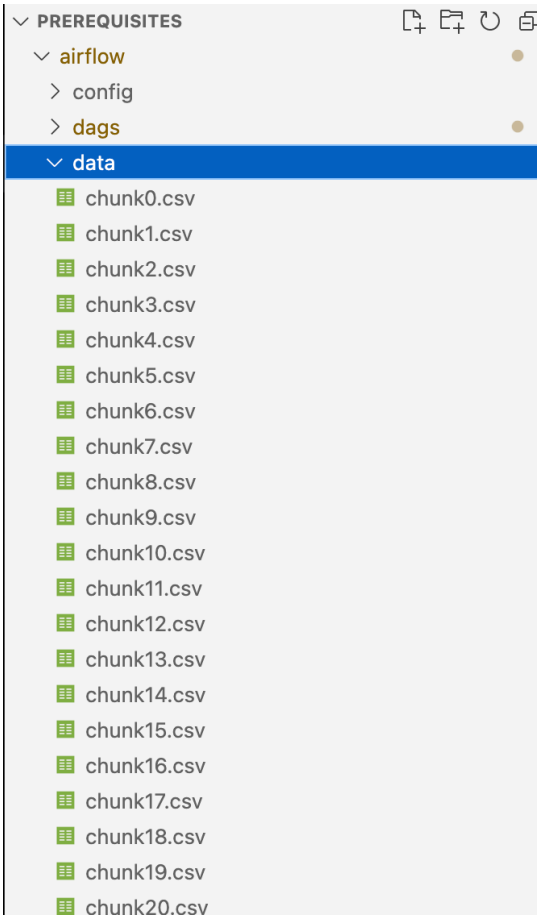

Рисунок 8 — Размещение файлов текущей директории в VS Code

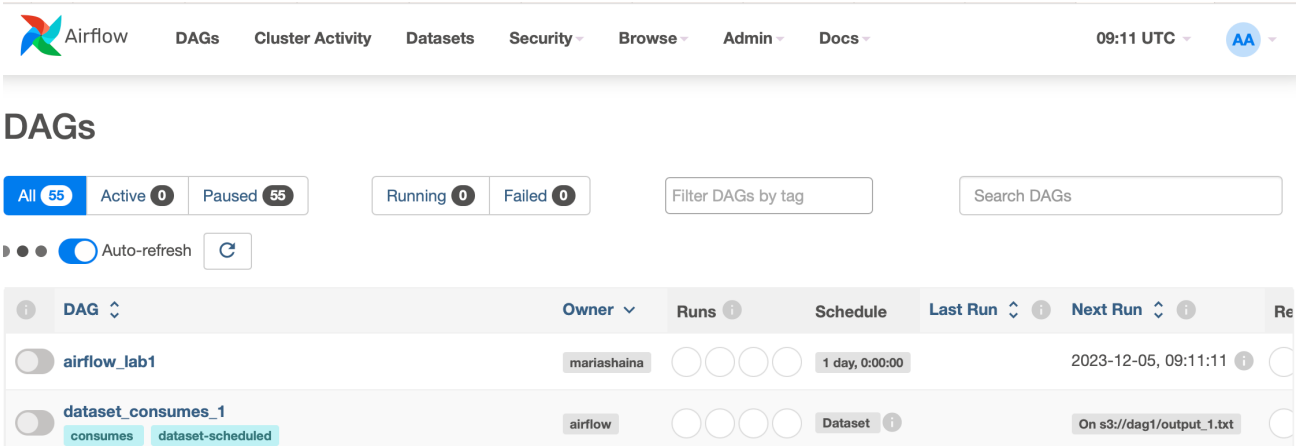

Рисунок 9 — Размещение DAG в Apache Airflow

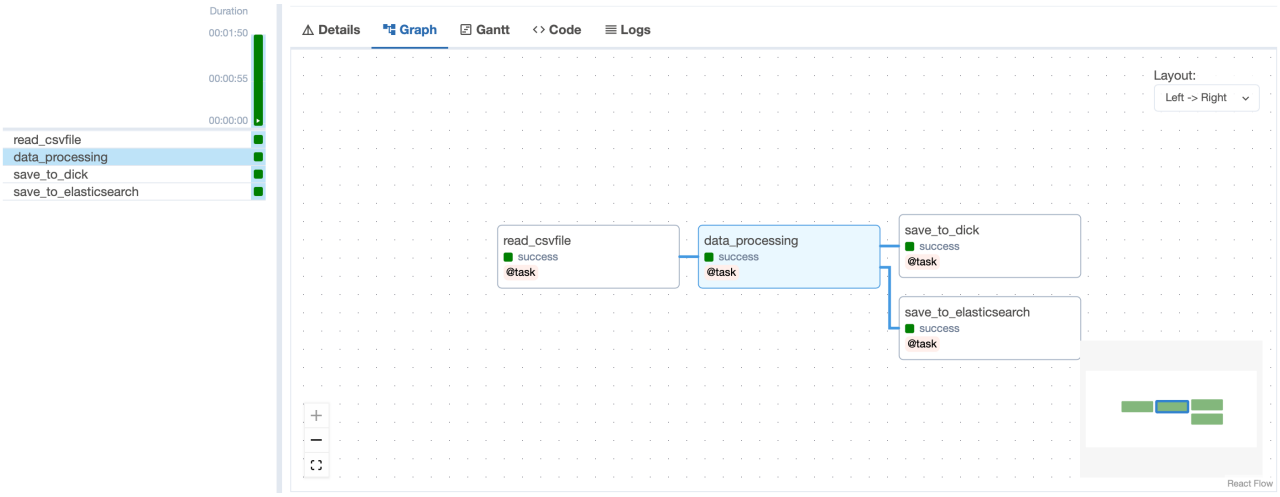

Рисунок 10 — Запуск DAG внутри Apache Airflow

Для того, чтобы построить гистограмму стоимости напитка к баллам поставленными дегустаторами средствами Kibana откроем интерфейс Elasticsearch. Я пользуюсь репозиторием Prerequisites, поэтому все контейнеры объединены в одну сеть и для обращения к Elasticsearch используется адрес http://elasticsearch-kibana:9200. Переходим на сайт и ищем наш DAG по индексу (метод _search), который мы указали. В моем случае это airflow1 (рисунок 11).

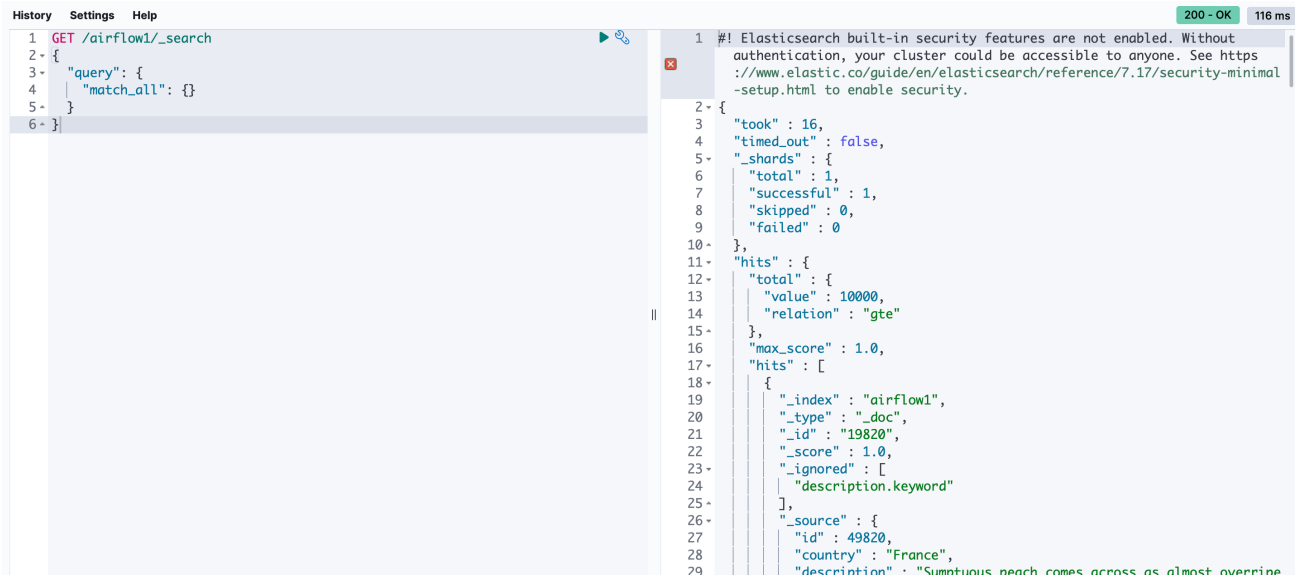

Рисунок 11 — Dev Tools в Elasticsearch

Далее в боковом меню переходим в раздел Dashboard и нажимаем «Создать визуализацию». Устанавливаем необходимые параметры (например, вид графика, вертикальную и горизонтальную оси и другие). Визуализация, созданная средствами Kibana представлена на рисунке 12.

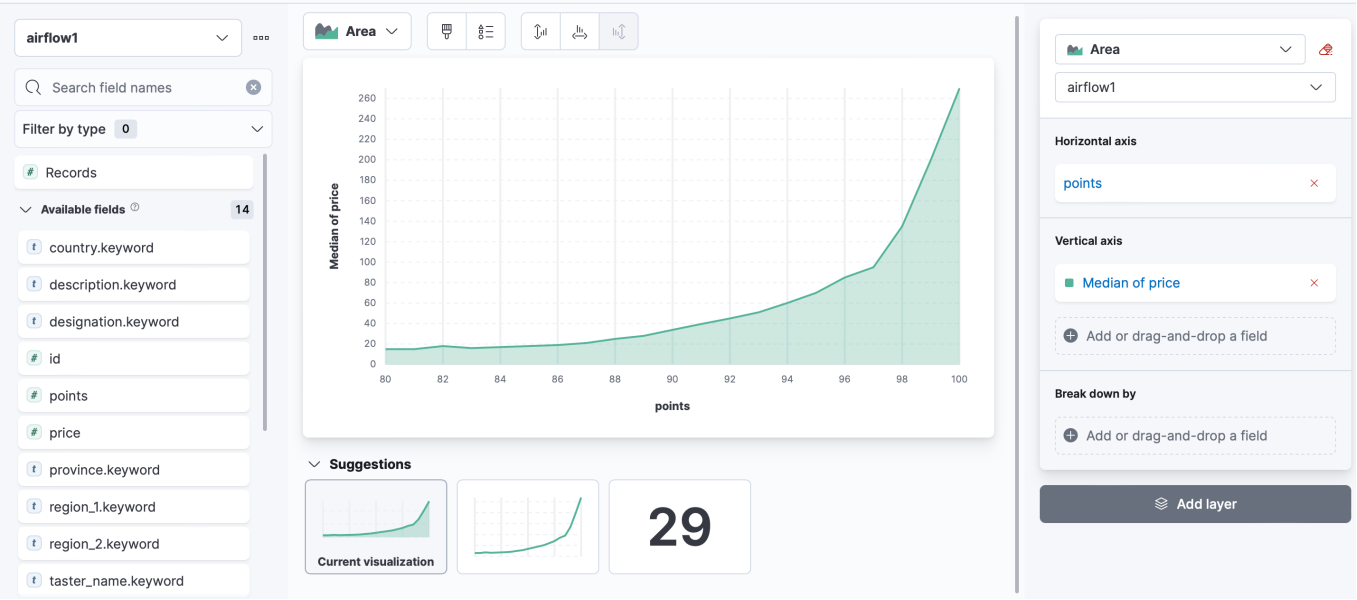

Рисунок 11 — Гистограмма стоимости напитка к баллам поставленными дегустаторами