

Отчёт 1 лабораторной работы по дисциплине “Инженерия данных”

выполнил Доружинский Дмитрий из группы 6233-010402D.

1. Подготовительный этап

Выполнил все рекомендации из репозитория <https://github.com/ssau-data-engineering/Prerequisites>

Системные	требования	следующие:
Система		
Процессор:	AMD Ryzen 5 4500U with Radeon Graphics	2.38 GHz
Установленная память (ОЗУ):	8,00 ГБ (7,40 ГБ доступно)	
Тип системы:	64-разрядная операционная система, процессор x64	
Перо и сенсорный ввод:	Перо и сенсорный ввод недоступны для этого экрана	

В ходе скачивания необходимых компонентов для выполнения лабораторной работы было выяснено, что нужно освободить более 40 гб свободной памяти, а также после выполнения команд:

- `docker compose -f docker-compose.airflow.yaml up --build -d`
- `docker compose -f docker-compose.nifi.yaml up --build -d`
- `docker compose -f docker-compose.elasticsearch.yaml up --build -d`

Необходимо вручную останавливать в Docker Desktop контейнеры иначе ресурсов моего компьютера не хватает и происходит зависание с последующим обнулением Docker. Привожу скриншоты загрузки компонентов. Также хочу отметить, что время загрузки всех компонентов с мобильного интернета составило +- 2 часа.

```
C:\Windows\system32\cmd.exe - docker compose -f docker-compose.airflow.yaml up airflow-init
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Дмитрий>cd .\prerequisites\

C:\Users\Дмитрий\prerequisites>docker network create data-engineering-labs-network
f9c9362fd680b1d373adce0c19940f6e2042124b56b55b1ea46866da70d1bf92

[+] Running 5/21 prerequisites>
- redis 6 layers [=====] 0B/0B Pulling 70.9s
  - 3c6368585bf1 Waiting 60.6s
  - 3911d271d7d8 Waiting 60.6s
  - ac88aa9d4021 Waiting 60.6s
  - 127cd75a68a2 Waiting 60.6s
  - 4f4fb700ef54 Waiting 60.6s
  - f3993c1104fc Waiting 60.6s
- postgres 13 layers [=====] 10.57MB/37.22MB Pulling 70.9s
  - 1f7ce2fa46ab Downloading [=====] 6.22MB/29.15MB 60.6s
  - d8e29f57d52c Download complete 2.6s
  - c8a84bacdc9c Download complete 45.3s
  - 4c9c7f55c6be Download complete 20.3s
  - 4f2487cd6295 Downloading [=====] 4.352MB/8.068MB 60.6s
  - 5a5f050db629 Download complete 54.7s
  - 691624362977 Downloading [=====] 116B/116B 60.6s
  - f464a58894c5 Download complete 58.5s
  - a600810358c9 Waiting 60.6s
  - 285c975ab2ee Waiting 60.6s
  - 58308e6e68e7 Waiting 60.6s
  - 6277814ae501 Waiting 60.6s
  - 607fd62806ab Waiting 60.6s

2.6s
```

```
Выпустить C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19045.3693]
(c) Корпорация Майкрософт (Microsoft Corporation). Все права защищены.

C:\Users\Дмитрий>cd .\prerequisites\

C:\Users\Дмитрий\prerequisites>docker network create data-engineering-labs-network
f9c9362fd680b1d373adce0c19940f6e2042124b56b55b1ea46866da70d1bf92

[+] Running 31/32 prerequisites>
- redis 6 layers [=====] 0B/0B Pulled 314.4s
  - 3c6368585bf1 Pull complete 181.3s
  - 3911d271d7d8 Pull complete 114.5s
  - ac88aa9d4021 Pull complete 101.8s
  - 127cd75a68a2 Pull complete 100.4s
  - 4f4fb700ef54 Pull complete 291.8s
  - f3993c1104fc Pull complete 294.1s
- postgres 13 layers [=====] 0B/0B Pulled 562.5s
  - 1f7ce2fa46ab Pull complete 2.6s
  - d8e29f57d52c Pull complete 45.3s
  - c8a84bacdc9c Pull complete 28.3s
  - 4c9c7f55c6be Pull complete 92.8s
  - 4f2487cd6295 Pull complete 54.7s
  - 5a5f050db629 Pull complete 200.2s
  - 691624362977 Pull complete 58.5s
  - f464a58894c5 Pull complete 54.7s
  - a600810358c9 Pull complete 91.9s
  - 285c975ab2ee Pull complete 95.6s
  - 58308e6e68e7 Pull complete 97.0s
  - 607fd62806ab Pull complete 98.5s
[+] Building 1890.5s (7/7) FINISHED docker:default
=> [airflow-init internal] load .dockerignore 0.3s
=> transferring context: 2B 0.0s
=> [airflow-init internal] load build definition from Dockerfile 0.3s
=> transferring dockerfile: 103B 0.0s
=> [airflow-init internal] load metadata for docker.io/apache/airflow:2.7.0 5.9s
=> [airflow-init auth] apache/airflow:pull token for registry-1.docker.io 0.0s
=> [airflow-init 1/2] FROM docker.io/apache/airflow:2.7.0@sha256:5958ce60a15a61f57612e9baec18f08c469192cb1d40ba6c82bf65 780.5s
=> resolve docker.io/apache/airflow:2.7.0@sha256:5958ce60a15a61f57612e9baec18f08c469192cb1d40ba6c82bf65:90f825 0.0s
=> sha256:1472c8cf7832865830f97a8d3492e2d1a68fbd22778f9a31dc6be4b4f12a9bc 31.42MB / 31.42MB 103.4s
=> sha256:2a18a514e348af4d3652161c54420b098c186c31162d8824bd1d63c77f1f087 4.22kB / 4.22kB 0.0s
=> sha256:9f47238f5dc1d67c33a350a81c0e31871780799c3c070a044f4fc2388c 12.50MB / 12.50MB 42.2s
=> sha256:5958ce60a15a61f57612e9baec18f08c469192cb1d40ba6c82bf65:90f825 743B / 743B 0.0s
=> sha256:4e29f76dcd178ed6c7f8921779da904c0a2a1ddcc7ba0f0e83187f984f 24.06kB / 24.06kB 4.1s
=> sha256:76076dc8a994ced154c0c6fbc4b1548ea31846375a23864db3b9efab24c0 1.08MB / 1.08MB 4.6s
=> sha256:1180a0064e4b61770bc2efc24de025c3a6536ad57403821c4549272f18 243B / 243B 16.1s
=> sha256:50711e002280e0faed37129292d6d5ebcd27d6c5d70ebf4f9b69818ed8f6e 3.14MB / 3.14MB 16.6s
=> sha256:40900413d3e260f113d148f1b39474abec4f46d7ae39b8412d272b790c4d4f 1.28kB / 1.28kB 215.8s
=> sha256:a0a1304531f685f2407c9ac0e1f52c873727818d54ab3dc90803273cf63dc095 58.39MB / 58.39MB 44.3s
=> sha256:8008f12b726c4b658679aefac35488251f5121019051255d60110334 907B / 907B 44.6s
=> sha256:c015eb2770ba6e39d35ea487cd35c9e2d402d3fbb6bdf4750a44d2a403ee83 2.50kB / 2.50kB 200.7s
=> sha256:c5d2d9f1ad21601847a7129f6599e1374e543e81ad2b3d3ad1bb2fca31e0a07 44.91MB / 44.91MB
```

```
C:\Windows\system32\cmd.exe
=> naming to docker.io/library/airflow-airflow-webserver 0.2s
=> [airflow-triggerer] exporting to image 0.2s
=> exporting layers 0.0s
=> writing image sha256:b786132455580c4b4d66731807f4d603e6ad47f9261d21d0f63d5e9b3e1 0.1s
=> naming to docker.io/library/airflow-airflow-triggerer 0.1s
=> [airflow-scheduler] exporting to image 0.2s
=> exporting layers 0.0s
=> writing image sha256:46c5355b6086067d5f4d0179bba1c5882f1bc1898dad5fa165943268864502 0.1s
=> naming to docker.io/library/airflow-airflow-scheduler 0.1s

[+] Running 8/8
- Container airflow-docker-proxy-1 Started 0.5s
- Container airflow-redis-1 Healthy 0.0s
- Container airflow-postgres-1 Healthy 0.0s
- Container airflow-airflow-init-1 Exited 0.0s
- Container airflow-airflow-triggerer-1 Started 0.6s
- Container airflow-airflow-webserver-1 Started 0.5s
- Container airflow-airflow-worker-1 Started 0.5s
- Container airflow-airflow-scheduler-1 Started 0.5s

C:\Users\Дмитрий\prerequisites>docker compose -f docker-compose.elasticsearch.yaml up --build -d
[+] Running 6/6
- elasticsearch-kibana 6 layers [=====] 0B/0B Pulled 1300.1s
  - 1efc2764f4f9 Pull complete 284.7s
  - a2f2f93da482 Pull complete 5.5s
  - 12cc4292b13c Pull complete 2.9s
  - d73cf48c0aac Pull complete 251.7s
  - e0d3013d675 Pull complete 175.2s
  - 4e7f9ea5ce0f Pull complete 107.4s
[+] Building 0.0s (0/0) docker:default
[+] Running 1/1
- Container elasticsearch-kibana-1 Started 3.9s

C:\Users\Дмитрий\prerequisites>docker compose -f docker-compose.nifi.yaml up --build -d
[+] Running 15/15
- apache-nifi 12 layers [=====] 0B/0B Pulled 2020.4s
  - 99de9192b4af Pull complete 141.8s
  - dabb63f19f5a Pull complete 41.3s
  - 76d9b9e0ce5d Pull complete 218.8s
  - ec221e1eb03 Pull complete 65.7s
  - d0b6f00c5a1f Pull complete 95.7s
  - f52c80b7731e Pull complete 67.4s
  - b707085ba23 Pull complete 60.1s
  - c431e02004d Pull complete 151.9s
  - a00bf41f3b28 Pull complete 428.1s
  - 3643289e1f78 Pull complete 2005.7s
  - 429665d61370 Pull complete 210.6s
  - 4f4fb700ef54 Pull complete 221.7s
[+] Building 0.0s (0/0) docker:default
[+] Running 1/1
- Container nifi-apache-nifi-1 Started 1.3s

C:\Users\Дмитрий\prerequisites>
```

После проверил корректно ли запускаются контейнеры с NiFi, Airflow и Elastic.

localhost:8080/login/

Airflow 19:19 UTC → Log In

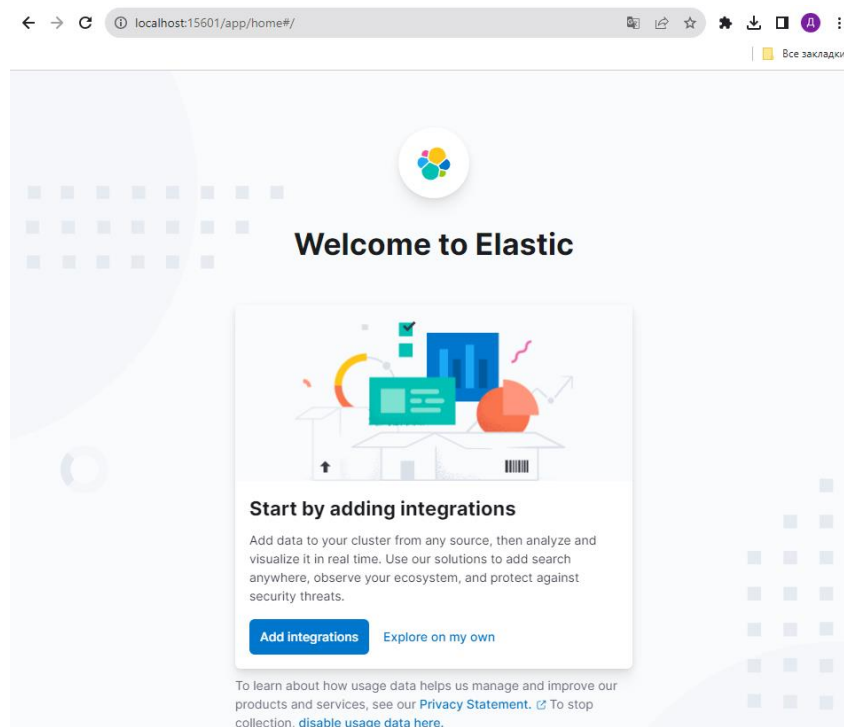
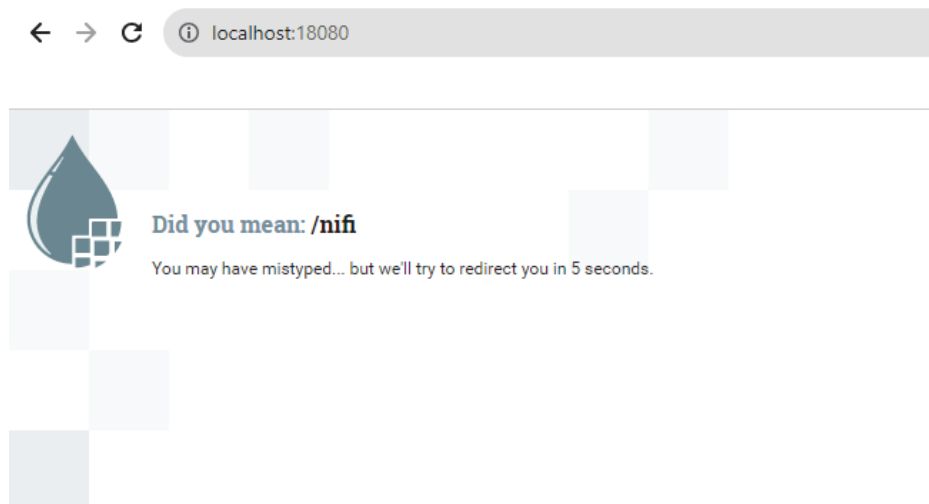
Sign In

Enter your login and password below:

Username:

Password:

Sign In

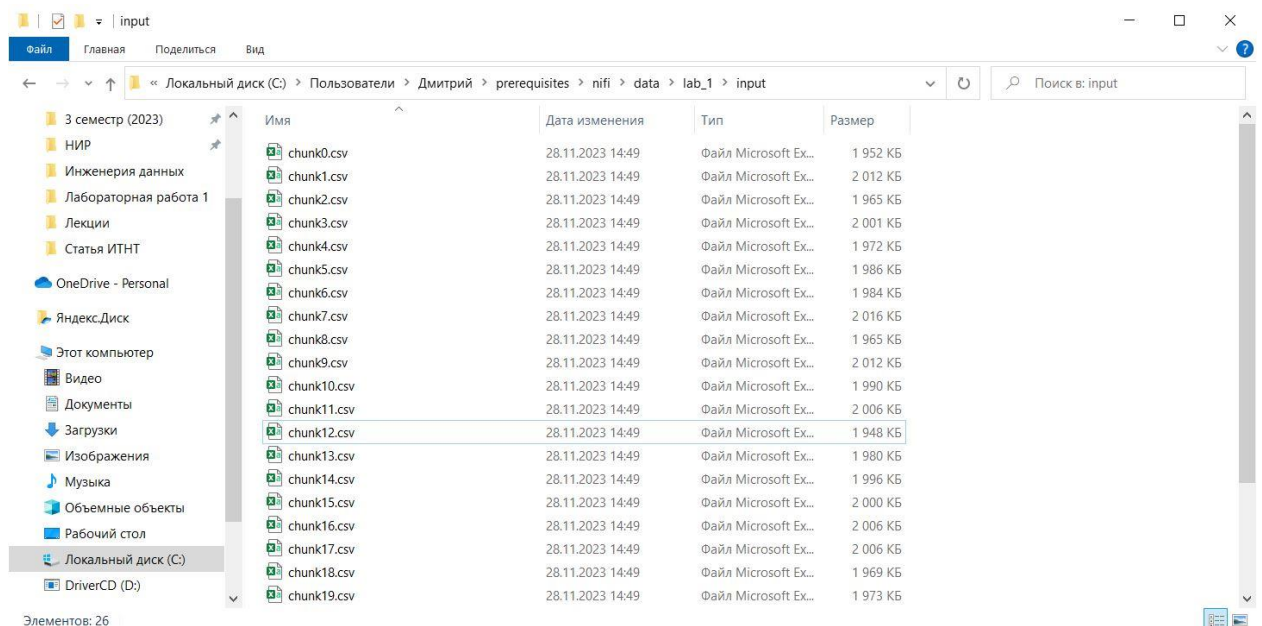


2. Построение пайплайна в NiFi

Для построения пайплайна в NiFi использовал рекомендуемые процессы, а именно:

- GetFile – для считывания файлов
- SplitRecord – для разделения данных на батчи
- QueryRecord – необходима для проверки в данных полей designation и region_1, чтобы они не были пустыми. После нужна для разделения строк с price == null и price != null
- UpdateRecord – изменяет имя получившегося файла, без этого файл будет с названием исходного файла.
- MergeContent – предназначен для объединения всех файлов в один
- UpdateAttribute – необходим для переименование получившегося файла
- PutFile – сохраняет файл в директорию где находятся исходные файлы
- PutElasticsearchHttp – нужна для переброса данных в Elasticsearch для построения графиков

Перед работой перенес предложенный набор данных в папку *nifi/data/lab_1/input*



Ниже привожу настройку каждого процесса

Configure Processor | GetFile 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

Property	Value	
Input Directory	/opt/nifi/nifi-current/data	
File Filter	[^\\].*	
Path Filter	No value set	
Batch Size	10	
Keep Source File	false	
Recurse Subdirectories	true	
Polling Interval	0 sec	
Ignore Hidden Files	true	
Minimum File Age	0 sec	
Maximum File Age	No value set	
Minimum File Size	0 B	
Maximum File Size	No value set	

CANCEL

APPLY

Processor Details | SplitRecord 1.23.2

Running

STOP & CONFIGURE

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

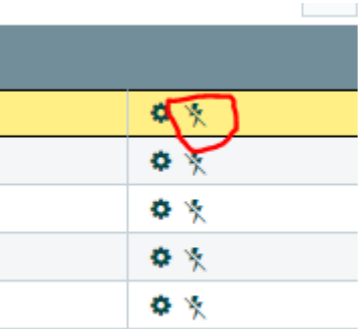
Required field

Property	Value	
Record Reader	CSVReader	→
Record Writer	CSVRecordSetWriter	→
Records Per Split	100	

OK

В настройке SplitRecord стоит упомянуть, что после выбора параметров

CSVReader и CSVRecordSetWriter необходимо нажать на стрелку и после в появившемся окне нажать на молнию, активировав тем самым Enable.



Processor Details | QueryRecord 1.23.2

▶ Running

⚙ STOP & CONFIGURE

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

Property		Value	
Record Reader	?	CSVReader	→
Record Writer	?	CSVRecordSetWriter	→
Include Zero Record FlowFiles	?	true	
Cache Schema	?	true	
Default Decimal Precision	?	10	
Default Decimal Scale	?	0	
correct	?	SELECT * FROM FLOWFILE WHERE designation <> ' AND r...	

OK

Поле correct содержит SELECT * FROM FLOWFILE WHERE designation <> ' AND region_1 <> '

Processor Details | QueryRecord 1.23.2

▶ Running

⚙ STOP & CONFIGURE

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

Property		Value	
Record Reader	?	CSVReader	→
Record Writer	?	CSVRecordSetWriter	→
Include Zero Record FlowFiles	?	true	
Cache Schema	?	true	
Default Decimal Precision	?	10	
Default Decimal Scale	?	0	
price_notnull	?	SELECT * FROM FLOWFILE WHERE price is not null	
price_null	?	SELECT * FROM FLOWFILE WHERE price is null	

OK

Processor Details | UpdateRecord 1.23.2

▶ Running

⚙ STOP & CONFIGURE

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

Property		Value	
Record Reader	?	CSVReader	→
Record Writer	?	CSVRecordSetWriter	→
Replacement Value Strategy	?	Literal Value	
/price	?	0.0	

OK

Configure Processor | MergeContent 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field



Property		Value	
Merge Strategy	?	Bin-Packing Algorithm	
Merge Format	?	Binary Concatenation	
Attribute Strategy	?	Keep Only Common Attributes	
Correlation Attribute Name	?	No value set	
Minimum Number of Entries	?	1	
Maximum Number of Entries	?	1000	
Minimum Group Size	?	0 B	
Maximum Group Size	?	No value set	
Max Bin Age	?	No value set	
Maximum number of Bins	?	5	
Delimiter Strategy	?	Do Not Use Delimiters	

CANCEL

APPLY

Configure Processor | UpdateAttribute 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field



Property		Value	
Delete Attributes Expression	?	No value set	
Store State	?	Do not store state	
Stateful Variables Initial Value	?	No value set	
Cache Value Lookup Cache Size	?	100	
filename	?	output.csv	



ADVANCED

CANCEL

APPLY

Configure Processor | PutFile 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field



Property		Value
Directory	?	/opt/nifi/nifi-current/data
Conflict Resolution Strategy	?	fail
Create Missing Directories	?	true
Maximum File Count	?	No value set
Last Modified Time	?	No value set
Permissions	?	No value set
Owner	?	No value set
Group	?	No value set

CANCEL

APPLY

Processor Details | PutElasticsearchRecord 1.23.2

Running

STOP & CONFIGURE

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

Property		Value
Index Operation	?	index
Index	?	nifi
Type	?	_doc
@timestamp Value	?	No value set
Client Service	?	ElasticSearchClientServiceImpl →
Record Reader	?	CSVReader →
Batch Size	?	100
ID Record Path	?	No value set
Index Operation Record Path	?	No value set
Index Record Path	?	No value set
Type Record Path	?	No value set
@timestamp Record Path	?	No value set

OK

Для корректной загрузки данных в Elasticsearch необходимо добавить параметр HTTP Hosts, для этого нажимаем на стрелочку, далее на

шестеренку напротив ElasticSearchClientServiceImpl и в появившемся окне
ВВОДИМ. _____ адрес.

Controller Service Details | ElasticSearchClientServiceImpl 1.23.2

SETTINGS

PROPERTIES

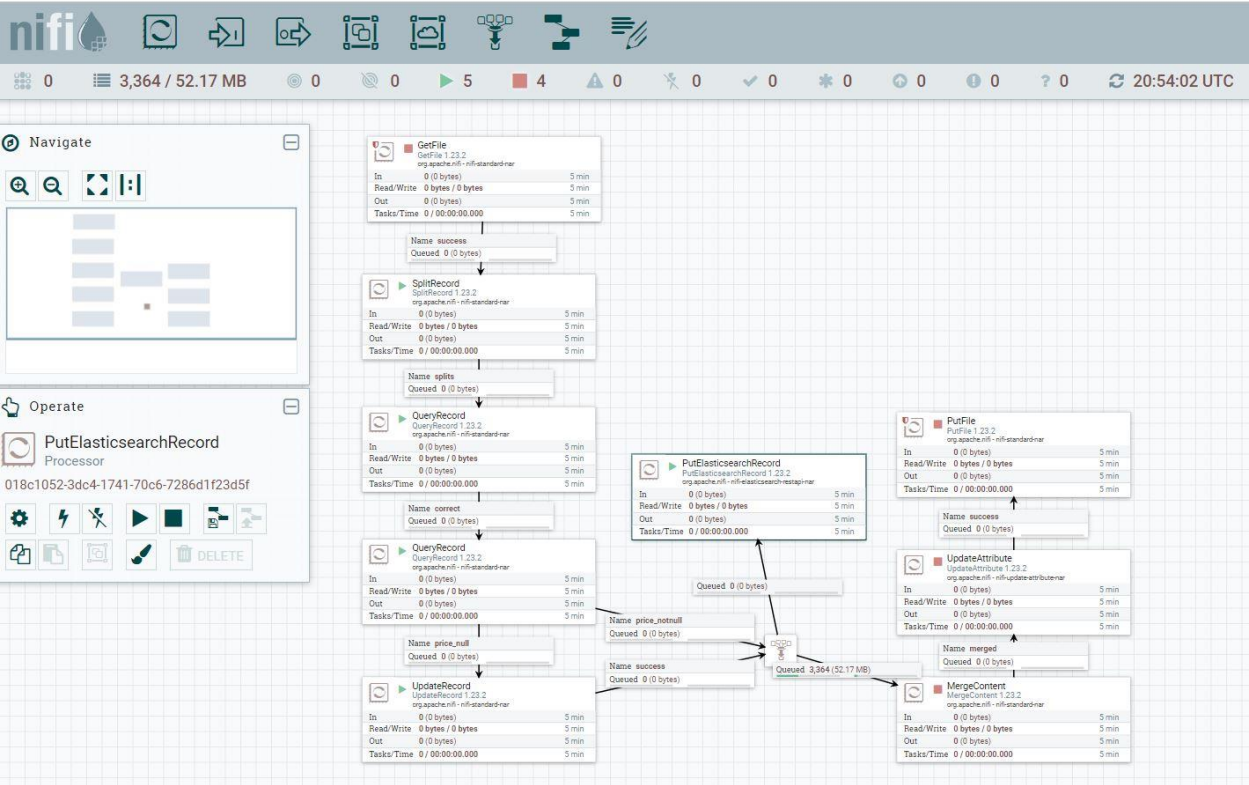
COMMENTS

Required field

Property	Value
HTTP Hosts	http://elasticsearch-kibana:9200
Path Prefix	No value set
Authorization Scheme	Basic
Username	No value set
Password	No value set
SSL Context Service	No value set
Proxy Configuration Service	No value set
Connect timeout	5000
Read timeout	60000
Retry timeout	60000
Charset	UTF-8
Suppress Null/Empty Values	Always Suppress
Enable Compression	false
Send Meta Header	true

OK

localhost:18080/nifi/?processGroupId=root&componentIds=018c1052-3dc4-1741-70c6-7286d1f23d5f

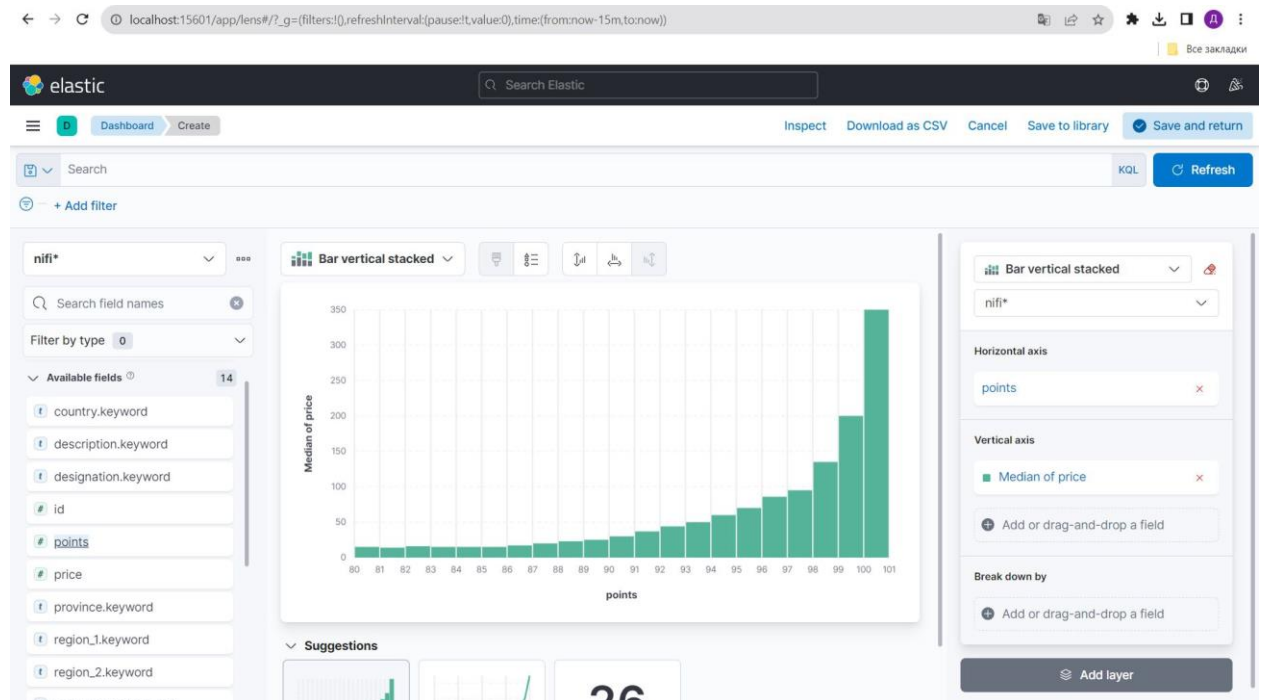


После убеждаемся, что данные пришли в Elasticsearch, для этого переходим в Index Management

The screenshot shows the Elastic Index Management interface. The left sidebar contains navigation links for Management, Ingest, Data, Alerts and Insights, and Kibana. The main panel is titled 'Index Management' and has tabs for Indices, Data Streams, Index Templates, and Component Templates. The 'Indices' tab is active, showing a table of indices. The table has columns for Name, Health, Status, Primaries, Replicas, Docs count, Storage size, and Data stream. One index named 'nifi' is listed with a health status of 'yellow', status 'open', 1 primary, 1 replica, 39047 documents, and 53.7mb storage size. There is a search bar and a 'Reload indices' button at the top of the table.

Name	Health	Status	Primaries	Replicas	Docs count	Storage size	Data stream
nifi	yellow	open	1	1	39047	53.7mb	

Далее создаём паттерн и строим график



3. Построение пайплайна в Airflow

Построение пайплайна в Airflow необходимо написать код на Python, использовал для этого VS Code со следующими расширениями:

- ms-python.python
- ms-toolsai.jupyter
- ms-vscode-remote.vscode-remote-extensionpack
- ms-azuretools.vscode-docker

Был написан следующий код для обработки предложенного датасета

```
C:\Users\Дмитрий\Desktop\Учеба\3 семестр (2023)\Инженерия данных\1 лаба\lab1.py > ...
1  from airflow import DAG
2  from airflow.operators.python import PythonOperator
3  from elasticsearch import Elasticsearch
4  from datetime import datetime
5  import pandas as pd
6  import numpy as np
7
8  with DAG('basic_etl_dag',
9
10         schedule_interval=None,
11
12         start_date=datetime(2023, 28, 11),
13
14         catchup=False) as dag:
15
16     def read_data():
17         result = pd.DataFrame()
18         for i in range(26):
19             result = pd.concat([result, pd.read_csv(f"/opt/airflow/data/chunk{i}.csv")])
20         return result
21
22     def transform_data():
23
24         result = read_data()
25         result = result[(result['designation'].str.len() > 0)]
26         result = result[(result['region_1'].str.len() > 0)]
27         result['price'] = result['price'].replace(np.nan, 0)
28         result = result.drop(['id'], axis=1)
29         result.to_csv('/opt/airflow/data/data.csv', index=False)
30
31     def load_to_elastic():
32         patch = Elasticsearch("http://elasticsearch-kibana:9200")
33         data = pd.read_csv(f"/opt/airflow/data/data.csv")
34
35         for i, row in data.iterrows():
36             doc = {
37                 "country": row["country"],
38                 "description": row["description"],
39                 "designation": row["designation"],
40                 "points": row["points"],
41                 "price": row["price"],
42                 "province": row["province"],
43                 "region_1": row["region_1"],
44                 "region_2": row["region_1"],
45                 "taster_name": row["taster_name"],
46                 "taster_twitter_handle": row["taster_twitter_handle"],
47                 "title": row["title"],
48                 "variety": row["variety"],
49                 "winery": row["winery"],
50             }
51
52         if i < data.shape[0] - 1:
53             patch.index(index="wines", id=i, document=doc)
54
55     transform_data = PythonOperator(
56
57         task_id='transform_data',
58
59         python_callable=transform_data,
60
61         dag=dag)
62
63     load_to_elastic = PythonOperator(
64
65         task_id='load_to_elastic',
66
67         python_callable=load_to_elastic,
68
69         dag=dag)
70
71     #task order
72     transform_data >> load_to_elastic
```

4. Вывод

В ходе проделанной работы были получены практические навыки работы с такими фреймворками как Apache Airflow и Nifi. Удалось построить пайплайн в NiFi с помощью графического интерфейса, а в Airflow с помощью кода Airflow. Так же получен полезный опыт работы с контейнерами в Docker Desktop