

Андреев Александр 6233

По шагам из репозитория загрузили и запустили все необходимые инструменты.

```
[+] Running 8/8
  Container airflow-docker-proxy-1      Started      0.6s
  Container airflow-postgres-1          Healthy      1.6s
  Container airflow-redis-1             Healthy      1.6s
  Container airflow-airflow-init-1       Exited       10.5s
  Container airflow-airflow-webserver-1  Started      11.3s
  Container airflow-airflow-triggerer-1  Started      11.6s
  Container airflow-airflow-scheduler-1  Started      11.1s
  Container airflow-airflow-worker-1     Started      11.7s

C:\Users\sasha\Desktop\Mara\2 род\DE\Prerequisites>docker compose -f docker-compose.nifi.yml up --build -d
[+] Running 13/13
  apache-nifi 12 layers [#####]      0B/0B      Pulled      170.9s
  99de9192b4af Pull complete          6.7s
  dabb03f19f5a Pull complete          7.8s
  76db90e0ce9d Pull complete          11.7s
  ec22c511cb53 Pull complete          11.8s
  d0bf00dc5a1f Pull complete          11.8s
  f52c80e7731e Pull complete          11.8s
  b7707083ba23 Pull complete          11.9s
  c861ed6b204d Pull complete          12.2s
  a00bf41f3b28 Pull complete          39.7s
  364328961ff8 Pull complete          168.3s
  429665d61370 Pull complete          168.3s
  4f4fb700ef54 Pull complete          168.3s
[+] Running 1/1
  Container nifi-apache-nifi-1          Started      0.8s

C:\Users\sasha\Desktop\Mara\2 род\DE\Prerequisites>docker compose -f docker-compose.elasticsearch.yml up --build -d
[+] Running 7/7
  elasticsearch-kibana 6 layers [#####] 0B/0B      Pulled      67.9s
  1efc276f4ff9 Pull complete          6.6s
  a2f2f93da482 Pull complete          6.7s
  12cca292b13c Pull complete          6.8s
  d73cf48caaac Pull complete          10.5s
  e0d30173d675 Pull complete          11.0s
  4ef79ea5ec0f Pull complete          65.7s
[+] Running 1/1
  Container elasticsearch-elasticsearch-kibana-1 Started      1.2s

C:\Users\sasha\Desktop\Mara\2 род\DE\Prerequisites>_
```

Рисунок 1 – Запуск сервисов

	airflow	Running (7/8)	5.65%	9 hours ago			
	nifi	Running (1/1)	1.63%	9 hours ago			
	elasticsearch	Running	1.63%	18080.18080			

Рисунок 2 – Запущенные сервисы

Далее в VS Code был написан код на Python для обработки датасета и отправки его в ElasticSearch.

```

8   with DAG("basic_etl_dag",
9
10      schedule_interval=None,
11
12      start_date=datetime(2023, 10, 7),
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

```

Рисунок 3 – Код Apache Airflow

С этой часть проблем не возникло.

В NiFi получилась следующая модель.

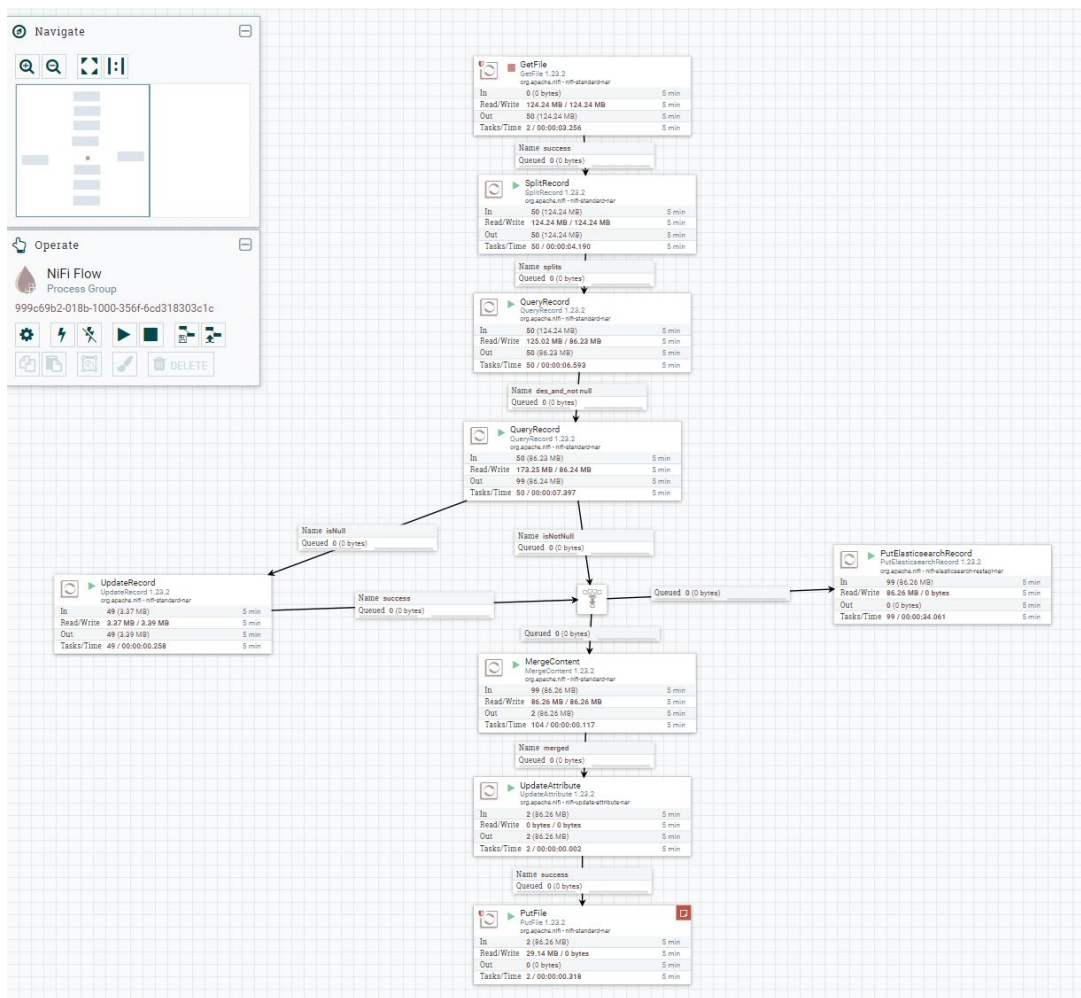


Рисунок 4 – Модель в NiFi

Тут первой проблемой стала обработка price, ибо непонятно было, как обрабатывать пустую строку. В итоге решилось все так:

Configure Processor
QueryRecord 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

Property	Value
Record Reader	CSVReader
Record Writer	CSVRecordSetWriter
Include Zero Record FlowFiles	false
Cache Schema	true
Default Decimal Precision	10
Default Decimal Scale	0
isNotNull	SELECT * FROM FLOWFILE WHERE price IS NOT NULL
isNull	SELECT * FROM FLOWFILE WHERE price IS NULL

CANCEL
APPLY

Рисунок 5 – Настройки QueryRecord

Еще одной проблемой стало то, что MergeContent мержит с заголовками. После нескольких часов поиска у меня не получилось найти, как можно пропускать первую строку. Была идея сделать через RouteOnAttribute, но я так и не понял почему он не находит строки по такому условию `${attribute:contains('id'):not()}`

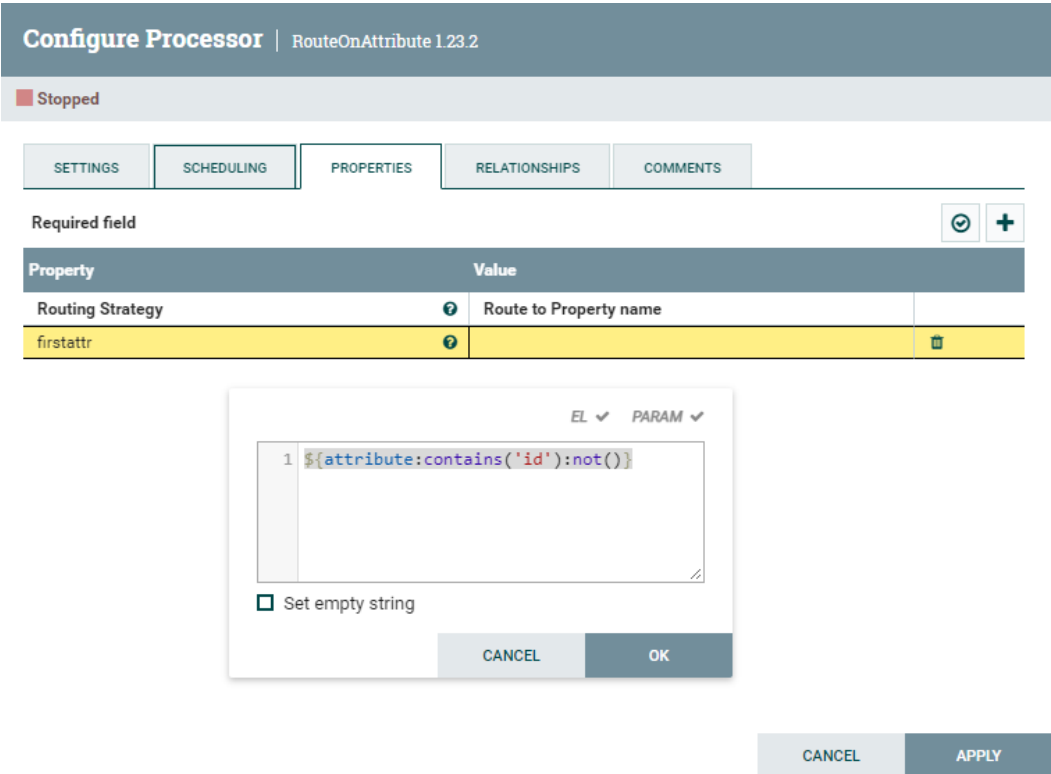


Рисунок 6 – Попытка убрать хедер

Однако несмотря на это отобразить графики получилось корректно.

Про настройки каждого процессора не вижу смысла говорить, там все стандартно и проблем с остальным не возникло.

Index Management

[Index Management docs](#)

Indices | Data Streams | Index Templates | Component Templates

Update your Elasticsearch indices individually or in bulk. [Learn more.](#)

☐ Include rollup indices ☐ Include hidden indices

Search

Reload indices

<input type="checkbox"/>	Name	Health	Status	Primaries	Replicas	Docs count	Storage size	Data stream
<input type="checkbox"/>	fromifi	<div><div></div><div>yellow</div></div>	open	1	1	285915	168.2mb	
<input type="checkbox"/>	data	<div><div></div><div>yellow</div></div>	open	1	1	141529	97mb	

Rows per page: 10

Рисунок 7 – Данные в Elastic

В Kibana создали паттерн и построили гистограмму.

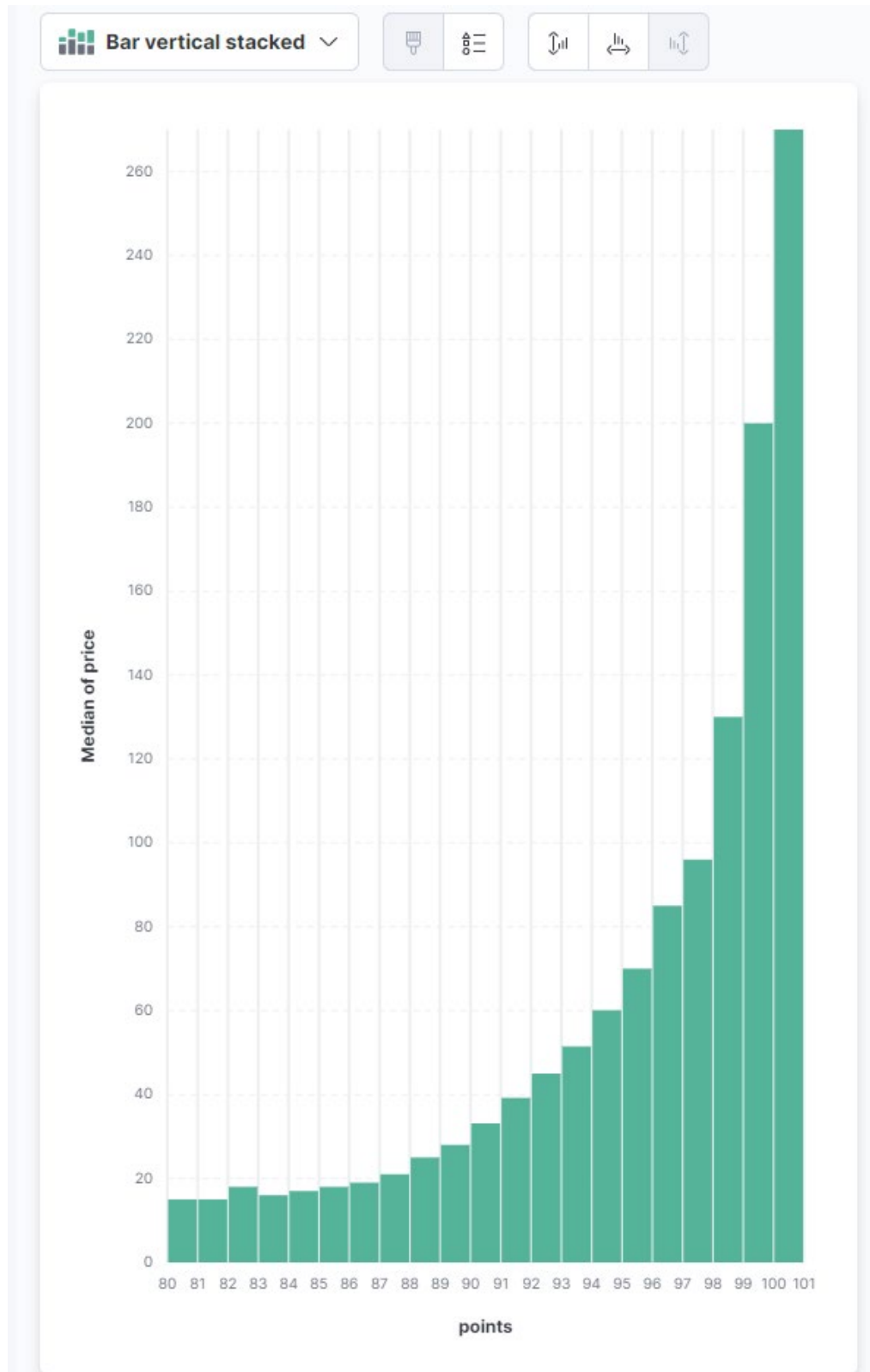


Рисунок 8 – Итоговая гистограмма

От себя пару слов: Apache NiFi ужасно неудобная программа. Я лично не понял зачем ей пользоваться, если для работы и обработки датасетов есть

Python с более или менее хорошей документацией. С Apache Airflow работать оказалось куда приятнее.

Также в обоих случаях корректно все данные корректно сохранились компьютер.

<< Prerequisites >

nifi >

data

Поиск в: data





	Имя	Дата изменения	Тип	Размер
доступ	 data.csv	04.11.2023 22:35	Файл Microsoft E...	29 842 КБ
1 стол	 chunk7.csv	04.11.2023 13:23	Файл Microsoft E...	2 016 КБ
л	 chunk1.csv	04.11.2023 13:23	Файл Microsoft E...	2 012 КБ
нты	 chunk9.csv	04.11.2023 13:23	Файл Microsoft E...	2 012 КБ

Рисунок 9 – Результирующие данные на компьютере