

Ход выполнения лабораторной работы

Нам нужны следующие инструменты -

- Apache Airflow
- Apache NiFi
- Elasticsearch

Apache Airflow

В lab1.py реализует DAG, который сначала считывает и обрабатывает данные из определённой директории. Сохраняет все данные в едином файле. И после пересылает данные в ElasticSearch.

```
def read_and_transf_data():
    data = pd.DataFrame()
    for i in range(26):
        data = pd.concat([data, pd.read_csv(f"/opt/airflow/data/chunk{i}.csv")])

    data = data[(data['designation'].str.len() > 0) & (data['region_1'].str.len() > 0)]
    data['price'] = data['price'].replace(np.nan, 0)
    data = data.drop(['id'], axis=1)
    data.to_csv('/opt/airflow/data/data.csv', index=False)

read_and_transf_task = PythonOperator(
    task_id='read_and_transf_task',
    python_callable=read_and_transf_data,
    dag=dag)
```

Рисунок 1 – чтение и обработка данных

```
def load_to_elastic():
    es = Elasticsearch("http://elasticsearch-kibana:9200")
    data = pd.read_csv(f"/opt/airflow/data/data.csv")

    for i, row in data.iterrows():
        doc = {
            "country": row["country"],
            "description": row["description"],
            "designation": row["designation"],
            "points": row["points"],
            "price": row["price"],
            "province": row["province"],
            "region_1": row["region_1"],
            "region_2": row["region_1"],
            "taster_name": row["taster_name"],
            "taster_twitter_handle": row["taster_twitter_handle"],
            "title": row["title"],
            "variety": row["variety"],
            "winery": row["winery"],
        }

        if i < data.shape[0] - 1:
            es.index(index="wines", id=i, document=doc)

load_to_elastic_task = PythonOperator(

    task_id='load_to_elastic_task',

    python_callable=load_to_elastic,

    dag=dag)
```

Рисунок 2 – пересылка данных в Elasticsearch

Далее указываем порядок выполнения DAG.

```
read_and_transf_task >> load_to_elastic_task
```

APACHE NIFI

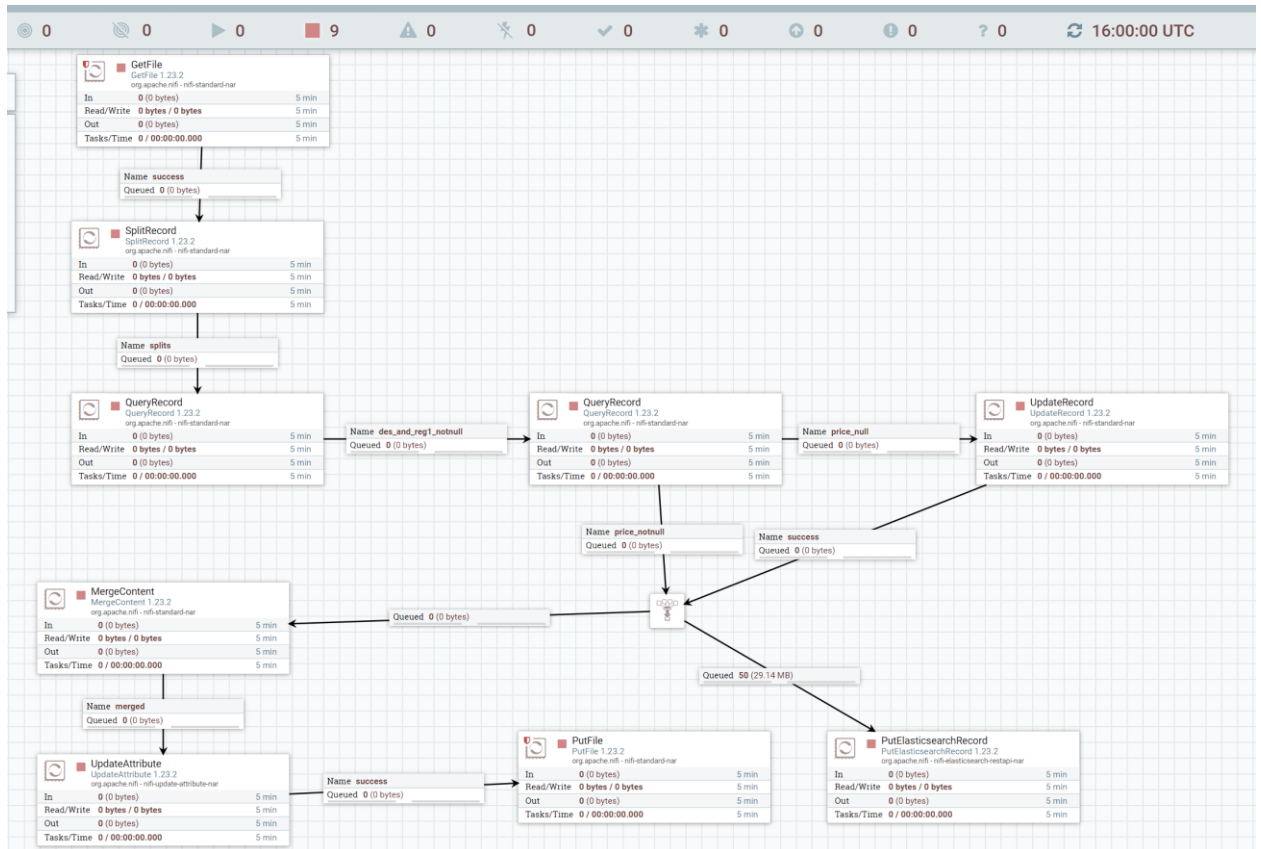


Рисунок 3 – схема NiFi

Делаем тоже что и в AirFlow. **GetFile** – считывает файлы. **SplitRecord** делит данные из файлов на батчи. **QueryRecord** – первая проверяет чтобы поля `designation` и `region_1` не были пустыми, а вторая разделяет строки с `price == null` и `price != null`. **UpdateRecord** заменяет все значения `price == null` на `price == 0.0`. **Funnel** собирает значения вместе. **MergeContent** собирает из всех файлов один файл. **UpdateAttribute** заменяет название получившегося файла на `output.csv` (Иначе файл имеет название одного из исходных файлов). **PutFile** сохраняет получившийся файл в ту же директорию. **PutElasticSearchRecord** пересылает данные в ElasticSearch.

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	?	[^\\].*\\.csv	
Path Filter	?	No value set	
Batch Size	?	25	
Keep Source File	?	true	
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

Рисунок 4 – Настройки GetFile

Configure Processor | SplitRecord 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

✔

+

Property		Value	
Record Reader	?	CSVReader	→
Record Writer	?	CSVRecordSetWriter	→
Records Per Split	?	100000	

CANCEL

APPLY

Рисунок 5 – Настройки SplitRecord

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	
des_and_reg1_notnull	?	SELECT * FROM FLOWFILE WHERE designation ...	🗑

CANCEL

APPLY

Рисунок 6 – Настройки QueryRecord

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	
price_notnull	?	SELECT * FROM FLOWFILE WHERE price is not ...	🗑
price_null	?	SELECT * FROM FLOWFILE WHERE price is null	🗑

CANCEL

APPLY

Рисунок 7 – Настройки QueryRecord

Configure Processor

UpdateRecord 1.23.2

Stopped

SETTINGS

SCHEDULING

PROPERTIES

RELATIONSHIPS

COMMENTS

Required field

✔

+

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

CANCEL

APPLY

Рисунок 8 – Настройки UpdateRecord

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

Рисунок 9 – Настройки MergeContent

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

Рисунок 10 – Настройки UpdateAttribute

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

Рисунок 11 – Настройки PutFile

Configure Processor | PutElasticsearchRecord 1.23.2

Stopped

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	

CANCEL

APPLY

Рисунок 12 – Настройки PutElasticSearchRecord

Kibana

После отправки данных в Elasticsearch переходим в Stack Management\Index Management. Можем увидеть поступившие данные.

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

Name	Health	Status	Primaries	Replicas	Docs count	Storage size	Data stream
nifi	yellow	open	1	1	101055	69.2mb	

На этой же странице переходим в kibana\index patterns. Создаём паттерн.

Stack Management

Index patterns

nifi*

Ingest Pipelines

Data

Index Management

Index Lifecycle Policies

Snapshot and Restore

Rollup Jobs

Transforms

Remote Clusters

Alerts and Insights

Rules and Connectors

Reporting

Machine Learning Jobs

Kibana

Index Patterns

Saved Objects

Tags

Search Sessions

Spaces

Advanced Settings

Stack

License Management

Upgrade Assistant

nifi*

Default

View and edit fields in nifi*. Field attributes, such as type and searchability, are based on [field mappings](#) in Elasticsearch.

Fields (30) | Scripted fields (0) | Field filters (0)

All field types

Add field

Name ↑	Type	Format	Searchable	Aggregatable	Excluded
._id	._id		•	•	
._index	._index		•	•	
._score					
._source	._source				
._type	._type		•	•	
country	text		•		
country.keyword	keyword		•	•	
description	text		•		
description.keyword	keyword		•	•	
designation	text		•		

Для визуализации переходим в visualize library. Строим график гистограммы стоимости напитка к баллам поставленными дегустаторами

