МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное
образовательное учреждение высшего образования
«Самарский национальный исследовательский университет
имени академика С.П. Королева»
(Самарский университет)

Институт информатики и кибернетики
Кафедра суперкомпьютеров и общей информатики

Отчет по лабораторной работе №1

Дисциплина: «Инженерия данных»

Тема: **«Базовый пайплайн работы с данными»**

Выполнил: Неженский М.С.

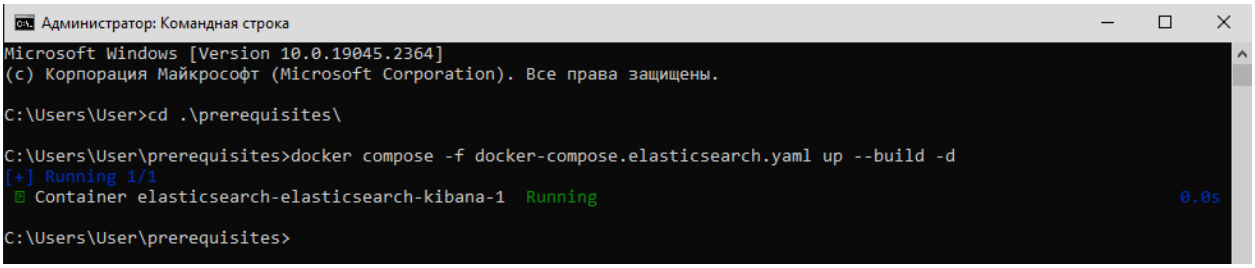
Группа: 6233-010402D

Самара 2023

1. Подготовка и проблемы

Все, команды необходимые для скачивания контейнеров брал тут: https://github.com/ssau-data-engineering/Prerequisites.

Установка Apache Airflow, Apache NiFi, ElasticSearch, MLflow, Postgresql:

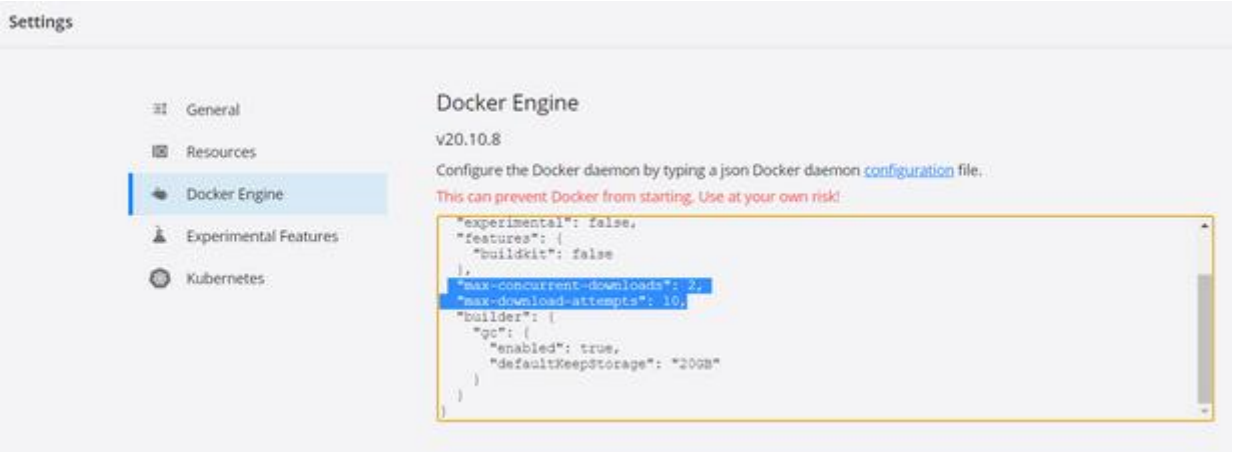


В ходе подготовки был установлен **vscode**, со следующими расширениями:

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

Для быстрой настройки среды воспользовался следующим профилем: Python Jupyter Remote Docker.code-profile.

Проблем была масса с Docker Desktop. Первая проблема была в медленном интернете, хотя скачивал через роутер. Решение – две сточки прописать на следующем рисунке в Docker Engine. Что-то про ограничение пакетов передачи данных, либо найти лучше интернет. При ограничении пакетов скорость становится еще ниже, вплоть до 2 часов загрузки всех 5 контейнеров. Прописывать строчки надо на одном уровне с "experimental": false через запятую, как показано далее. Ссылка на решение: https://stackoverflow.com/questions/53677592/docker-pull-unexpected-eof#comment114794645_56450064



Вторая проблема – оперативная память, если память как у меня – 8ГБ, то ограничение сразу надо ставить в 4ГБ, так как я поставил 6ГБ и когда запускал контейнеры, все настолько висло, что даже чат в телеграмме становился черным. Если нажать на приложение докера, то он зависнет и предложит завершить процесс, если его завершить, то все скачанные тобой контейнеры за два часа времени исчезнут. Скачивание начинается заново. После настройки ограничения вот такие показатели работы ПК. ЦП и Память загружена на 95-99%, но хотя бы люто не лагает и Windows всегда отвечает. Был создан файл C:/Users/%Имя пользователя%/.wslconfig со следующим содержимым:

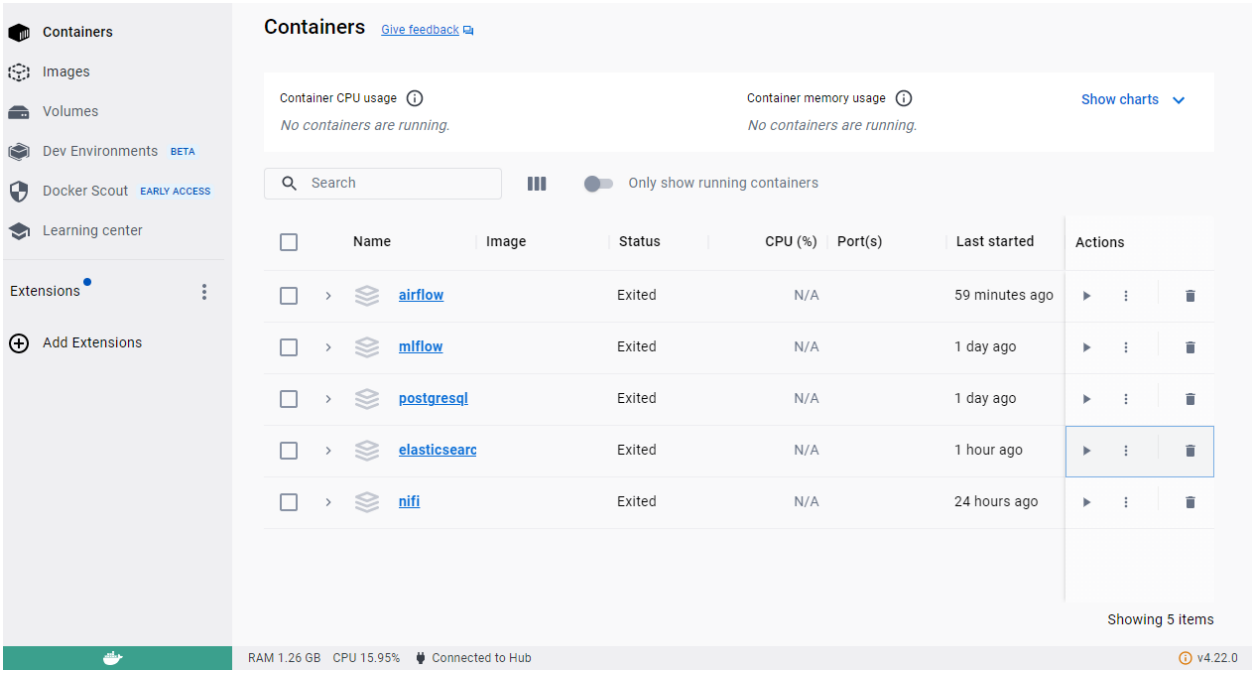
[wsl2]
memory=4GB
processors=4

Включать контейнеры надо всегда раздельно если конечно они не необходимы для совместной работы, как например: 1) Apache NiFi и ElasticSearch, 2) Apache Airflow и ElasticSearch в данной работе.

Третья проблема – это работа контейнеров. Решения я так и не нашел. В Nifi удалось зайти пару раз и еще три раза в Airflow. Сидишь и ждешь как у моря погоды и не знаешь заработают ли контейнеры по ссылке в этот раз или нет. В докере все отображается отлично. Я переустанавливал контейнеры, после удаления, перезапускал систему, выключал и запускал снова, через консоль, через докер и внутри каждого контейнера перезапускал

части. Бывает что сел делать лабораторную, а в итоге целый день просидел за запуском контейнера. А потом попробовал еще раз и заработала. С данной аномалией я так и не разобрался.

Добавленные контейнеры:



2. Apache NiFi

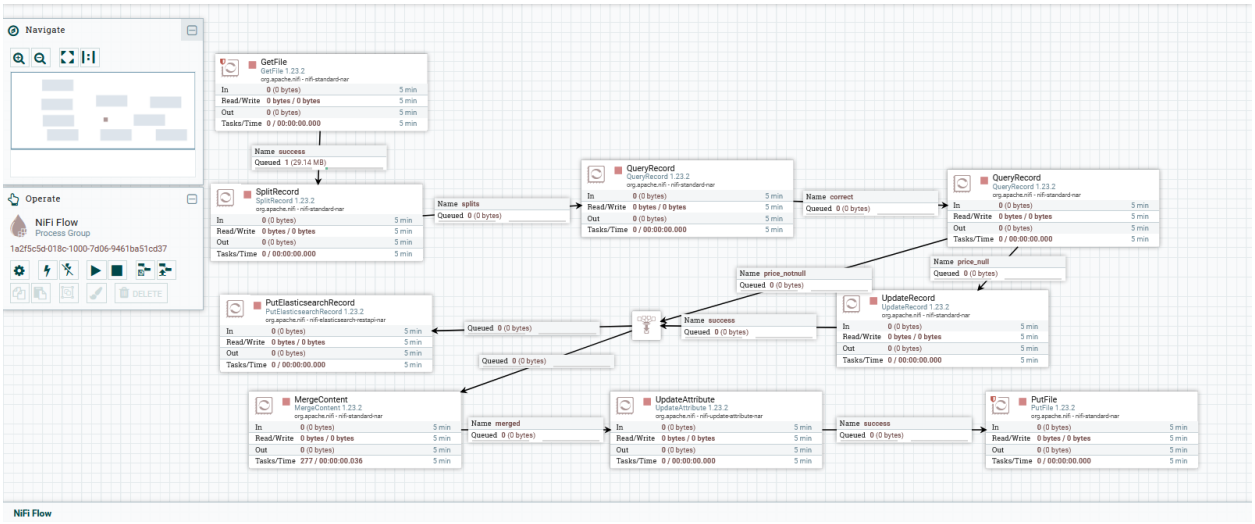
Знакомство с основными инструментами построения пайплайнов началось с Apache NiFi - простой в использовании, мощной и надежной система для обработки и распространения данных.

CSV файлы были перенесены в папку, к которой **NiFi** имеет доступ (в моем случае в папку `nifi/data/input`).

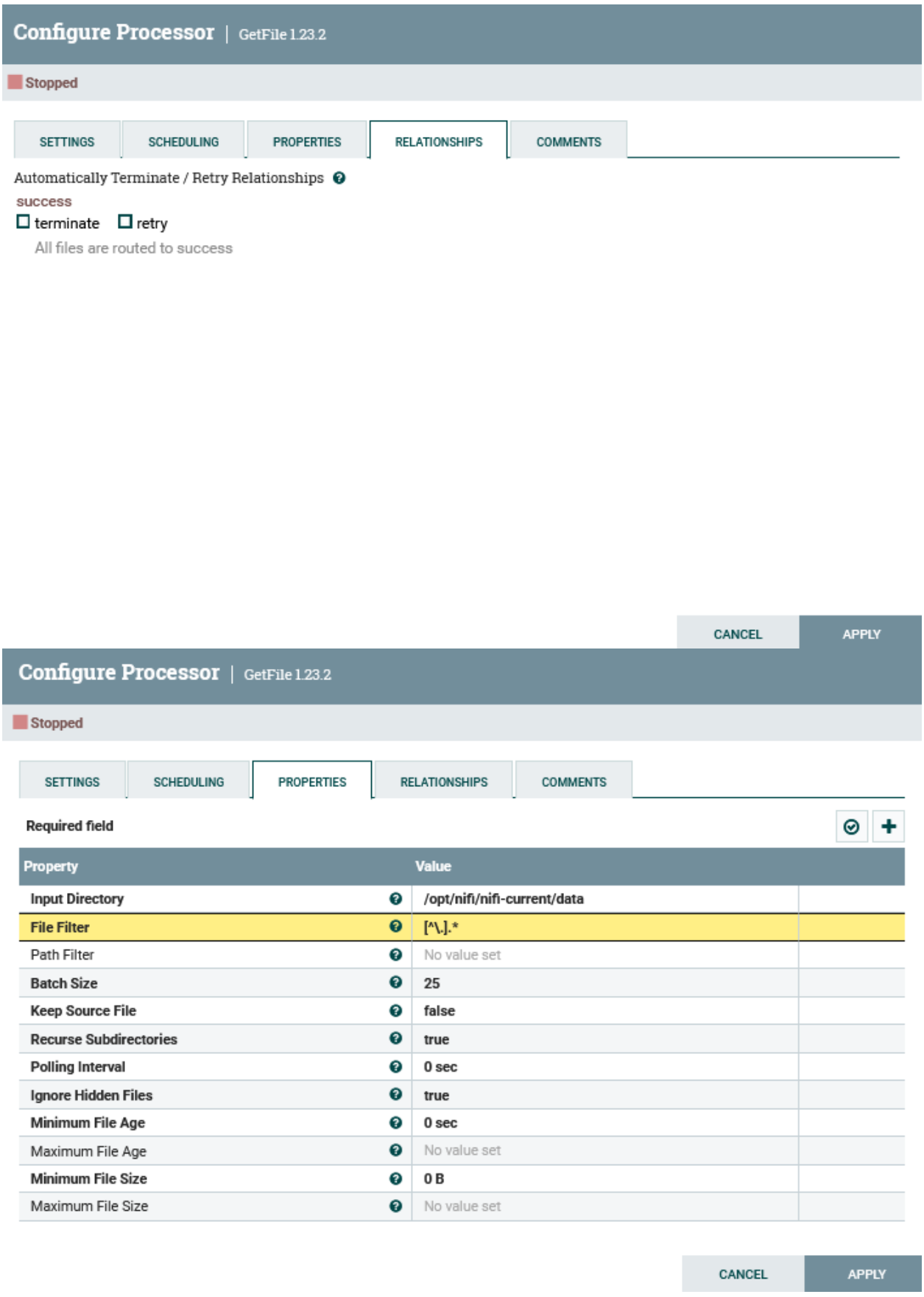
Для реализации пайплайна воспользовался следующими процессами:

- **GetFile** - создает FlowFiles из файлов в каталоге. NiFi будет игнорировать файлы, для которых у него нет хотя бы прав на чтение.
- **SplitRecord** - разбивает входной FlowFile в формате данных, ориентированном на записи, на несколько меньших FlowFile.
- **QueryRecord** - оценивает один или несколько запросов SQL по содержимому FlowFile. Первый QueryRecord нужен для того, чтобы поля designation и region_1 не были пустыми, второй для того, чтобы разделить поля, где price принимает значение null и не null.
- **UpdateRecord** - обновляет содержимое FlowFile, содержащего данные, ориентированные на запись (т. е. данные, которые можно прочитать через RecordReader и записать с помощью RecordWriter).
- **PutElasticsearchHttpRecord** - записывает записи из FlowFile в Elasticsearch, используя указанные параметры, такие как индекс для вставки и тип документа, а также тип операции (индекс, обновление, удаление и т. д.).
- **MergeContent** - объединяет группу FlowFile на основе определяемой пользователем стратегии и упаковывает их в один FlowFile.
- **UpdateAttribute** - этот процессор обновляет атрибуты FlowFile, используя свойства или правила, добавленные пользователем, то есть переименовывает файл в нашем случае.
- **PutFile** - записывает содержимое FlowFile в локальную файловую систему.

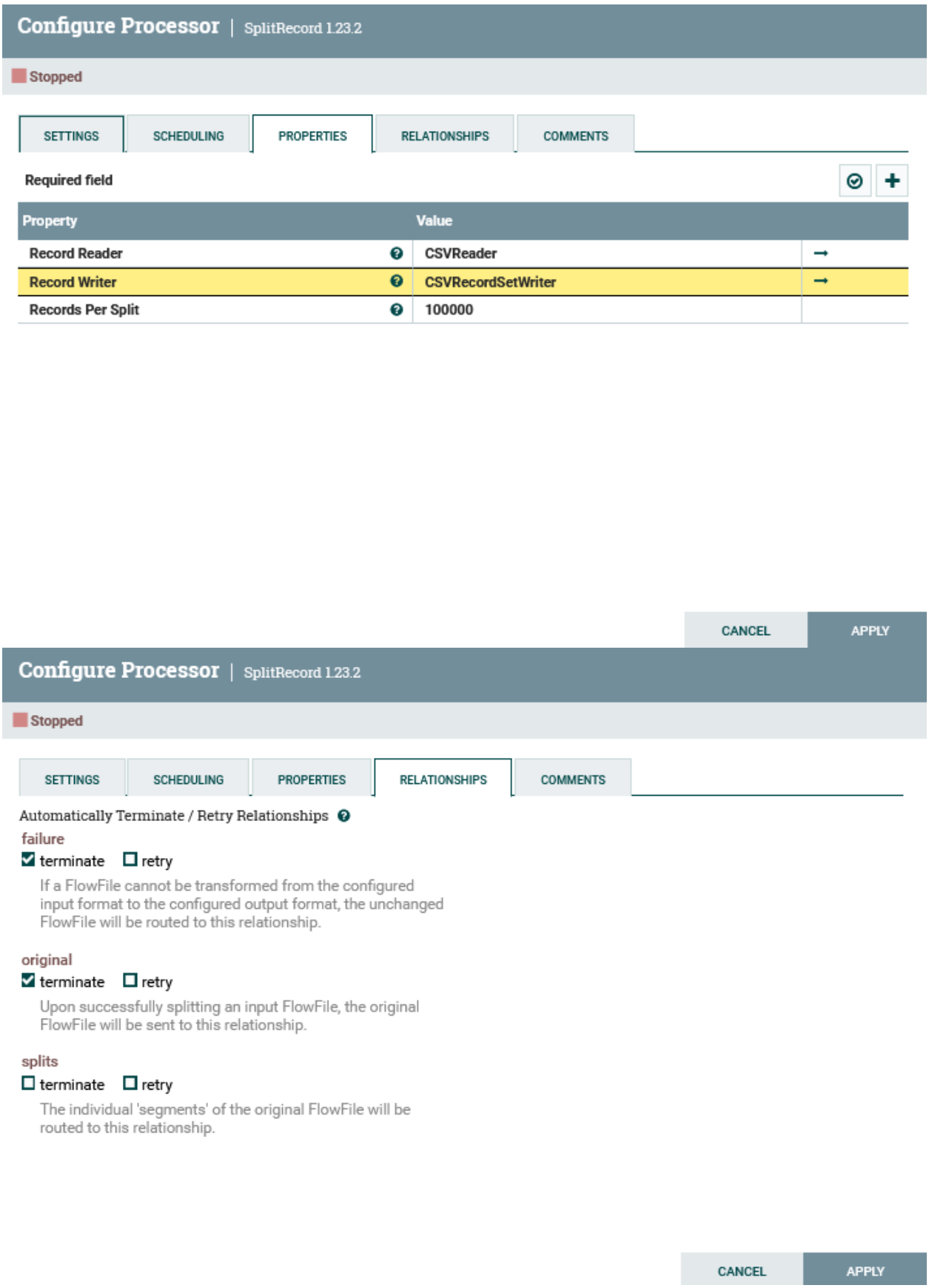
Итоговая схема построенная в Apache NiFi:



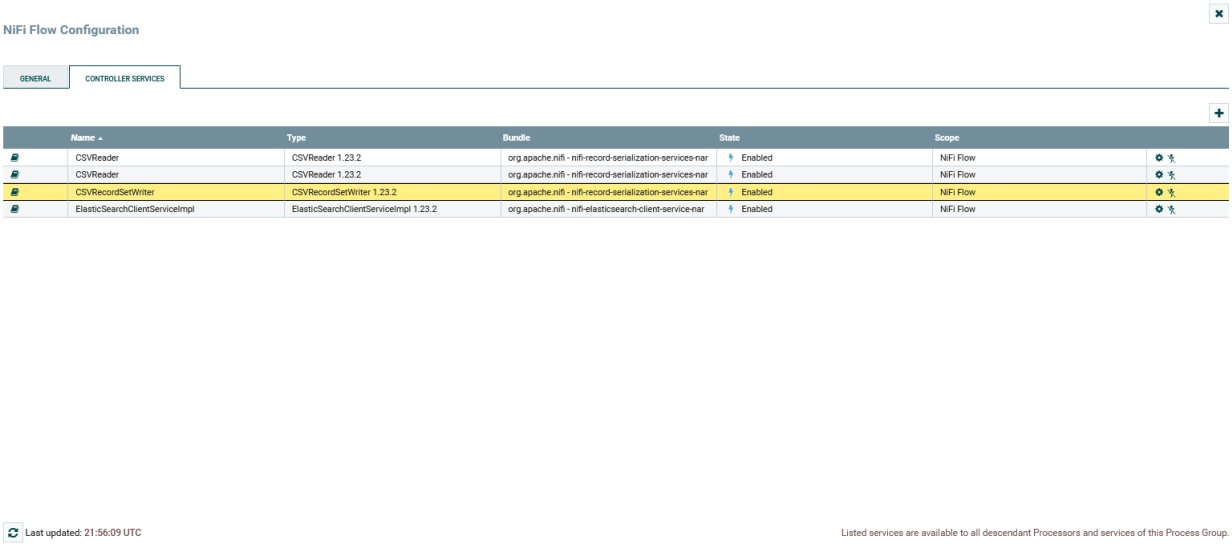
Далее приведены скриншоты для настройки данной схемы, приведены как настройки, так и отношения. В GetFile прописан путь загрузки входных данных и фильтр для используемых файлов.



Для чтения и записи используется CSVReader и CSVRecordSetWriter, после данных настроек следует нажать на стрелочку около CSVReader.

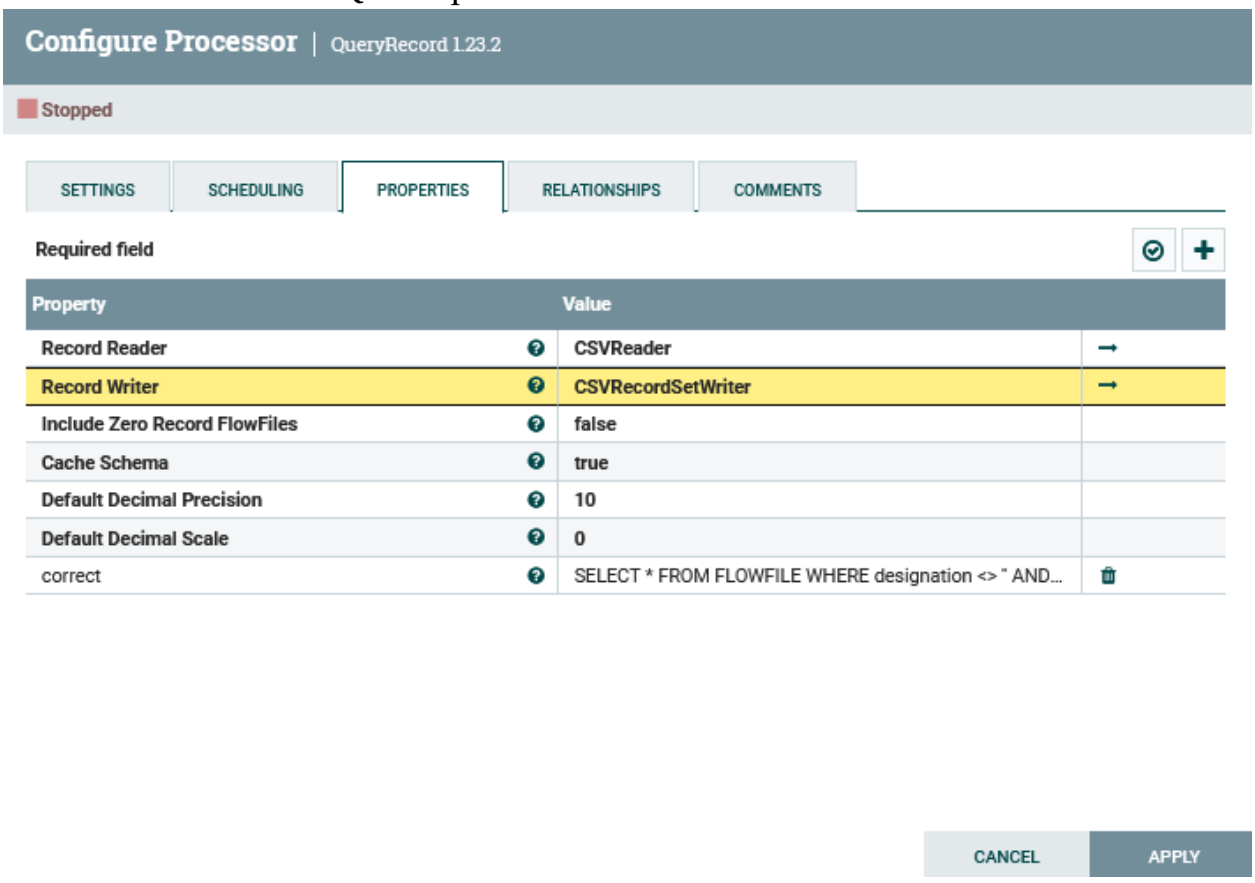


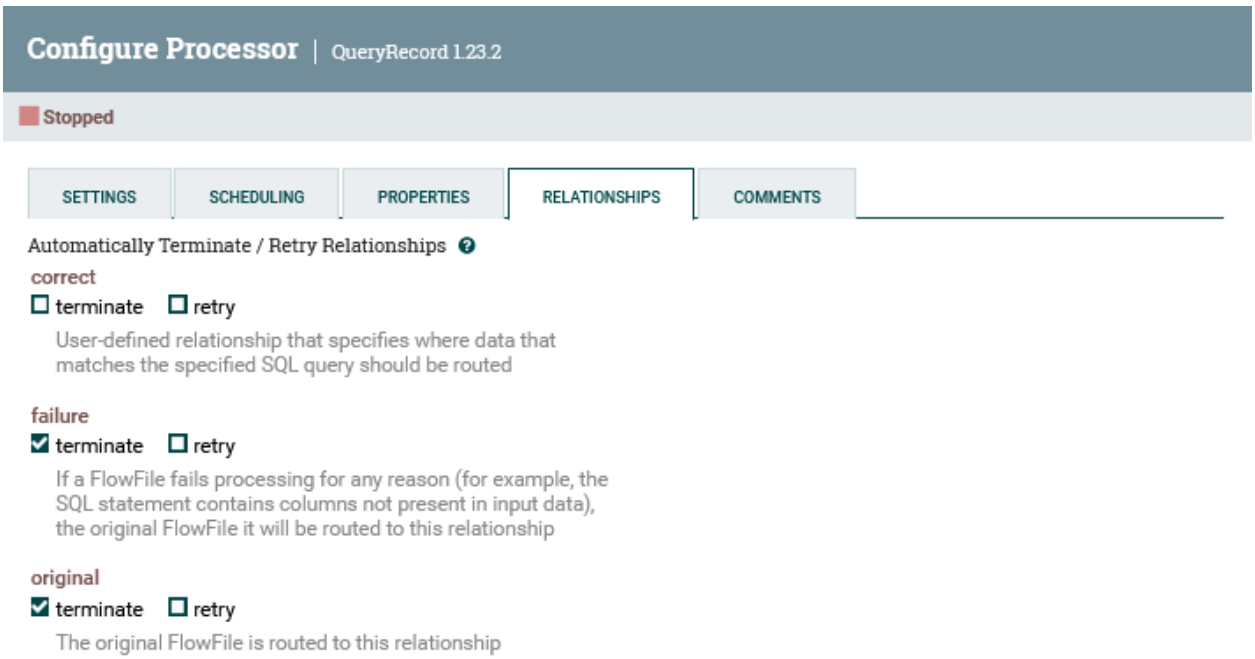
В появившемся окне стоит нажать на молнию и в появившимся окне нажать кнопку Enable для включения данных функций в программу, чтобы получилось вот так:

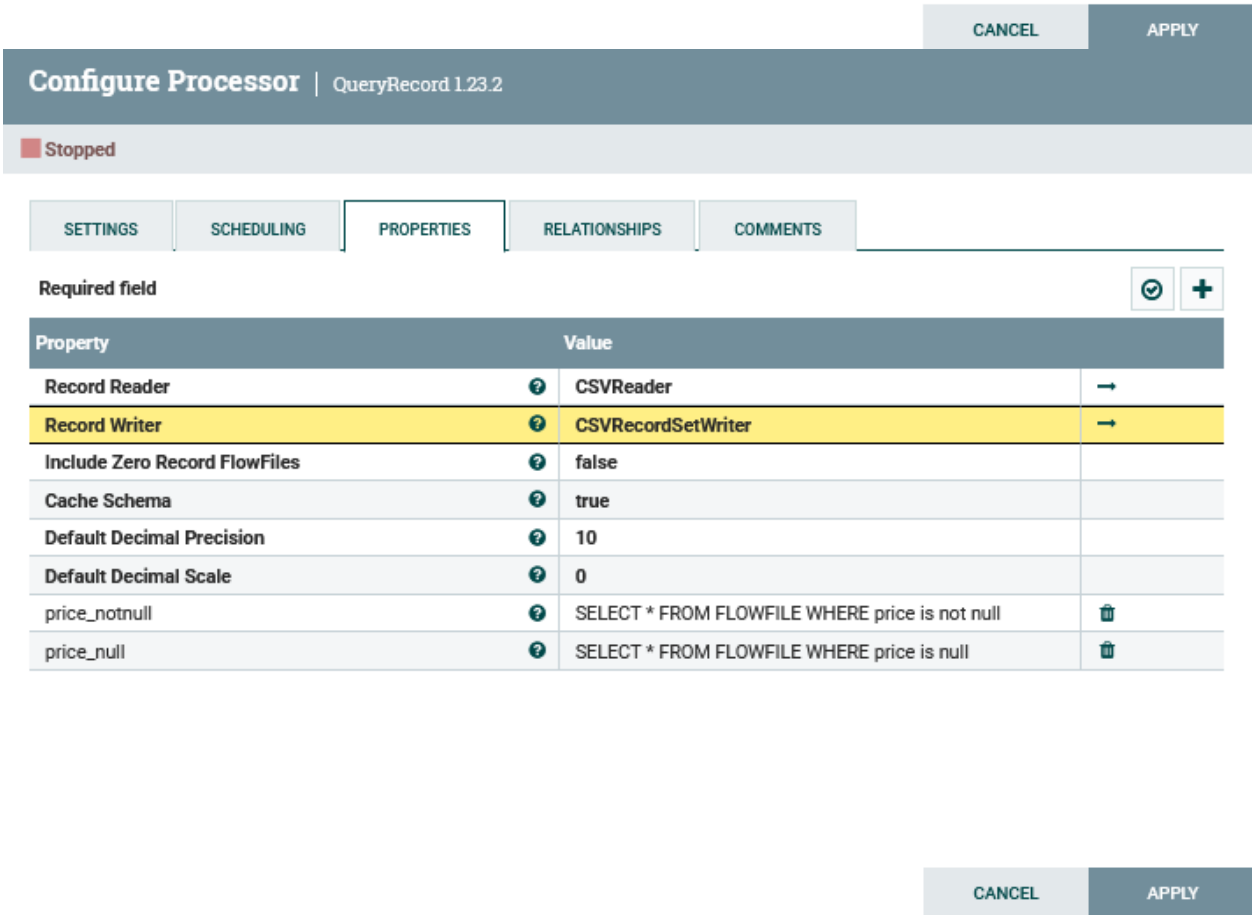


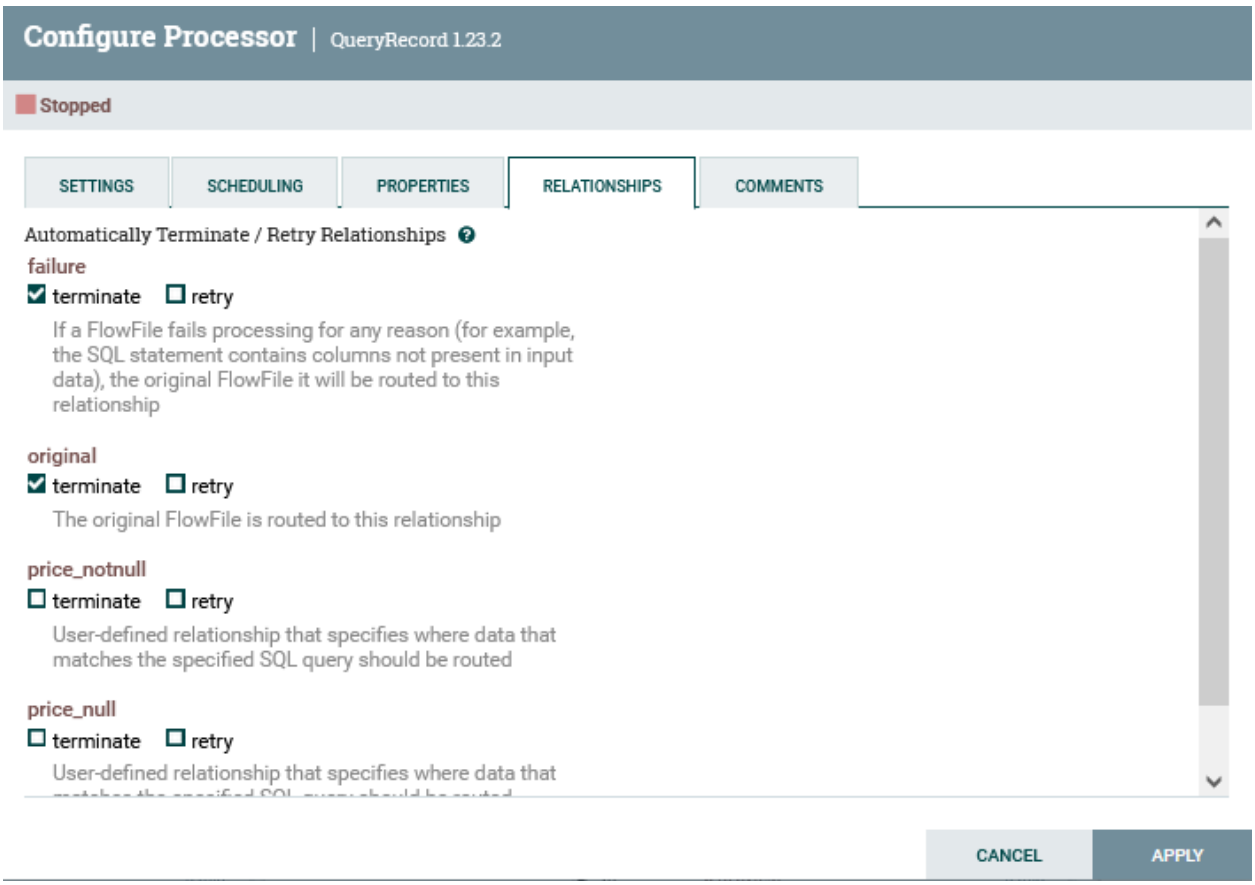
Конечно CSVReader должен быть один, но я забыл его удалить.

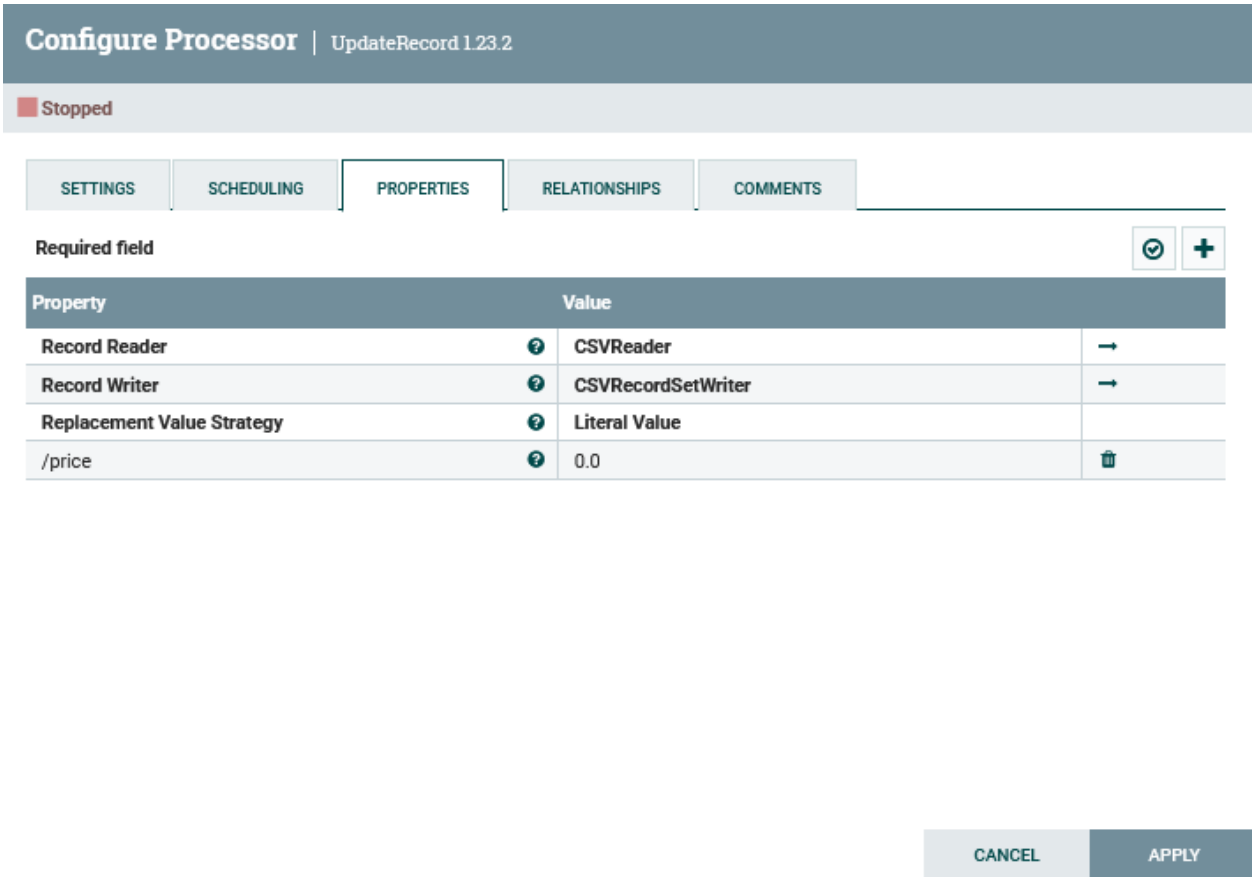
В QueryRecord мы нажимаем на плюсик и создаем новую строчку с названием correct и SQL-запросом в значении.

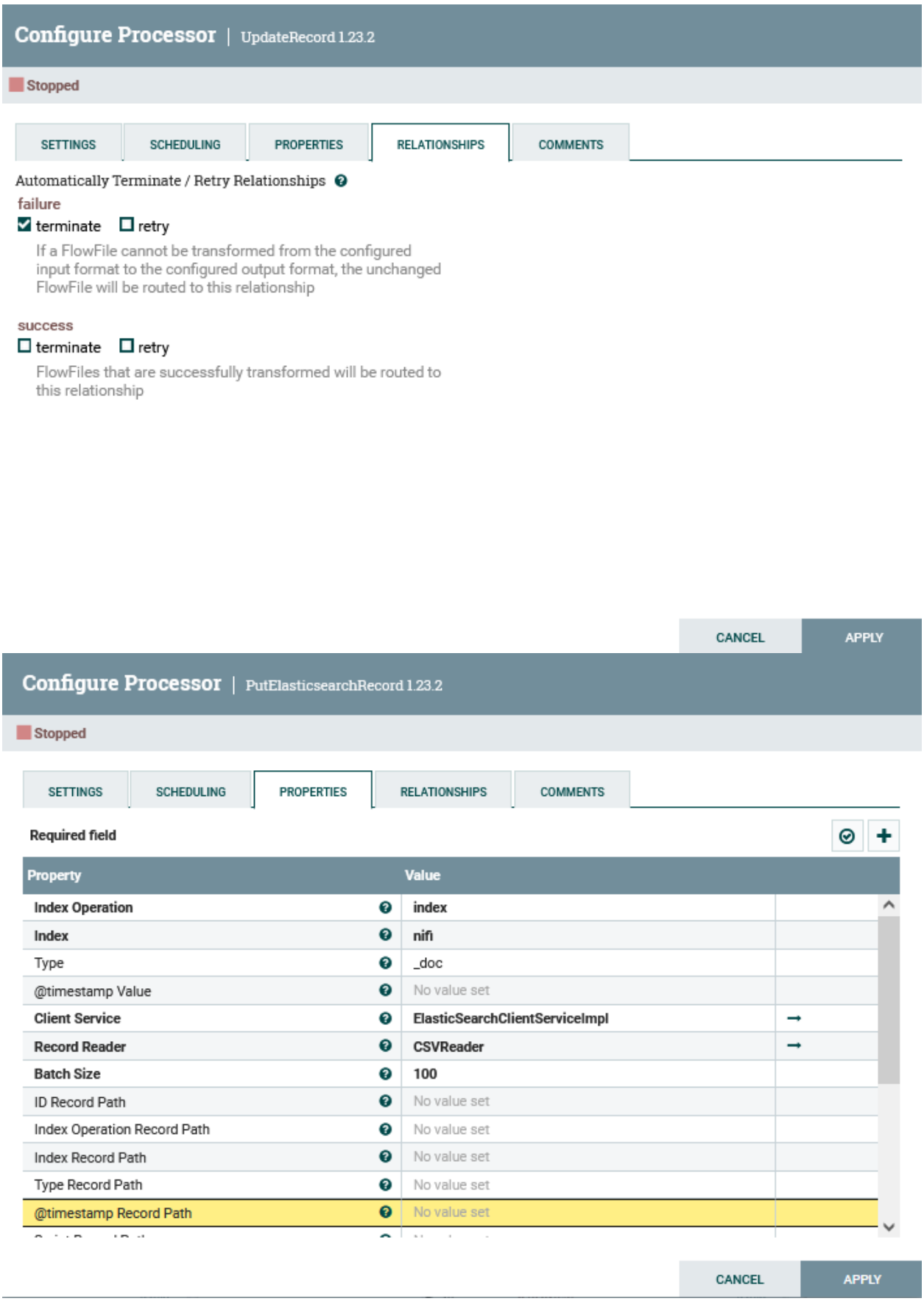


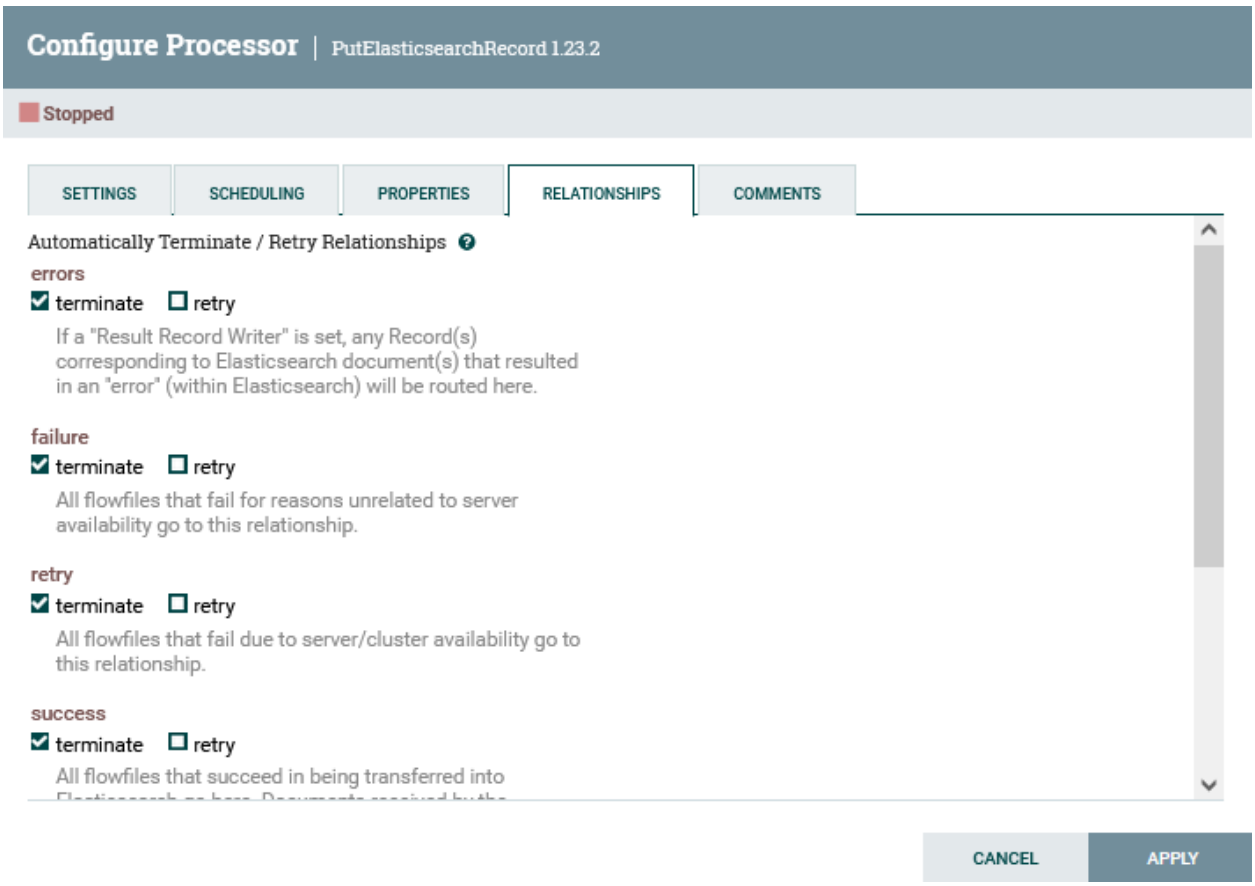

Configure Processor | QueryRecord 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Automatically Terminate / Retry Relationships
correct
terminate
retry
User-defined relationship that specifies where data that matches the specified SQL query should be routed
failure
terminate
retry
If a FlowFile fails processing for any reason (for example, the SQL statement contains columns not present in input data), the original FlowFile it will be routed to this relationship
original
terminate
retry
The original FlowFile is routed to this relationship
CANCEL
APPLY

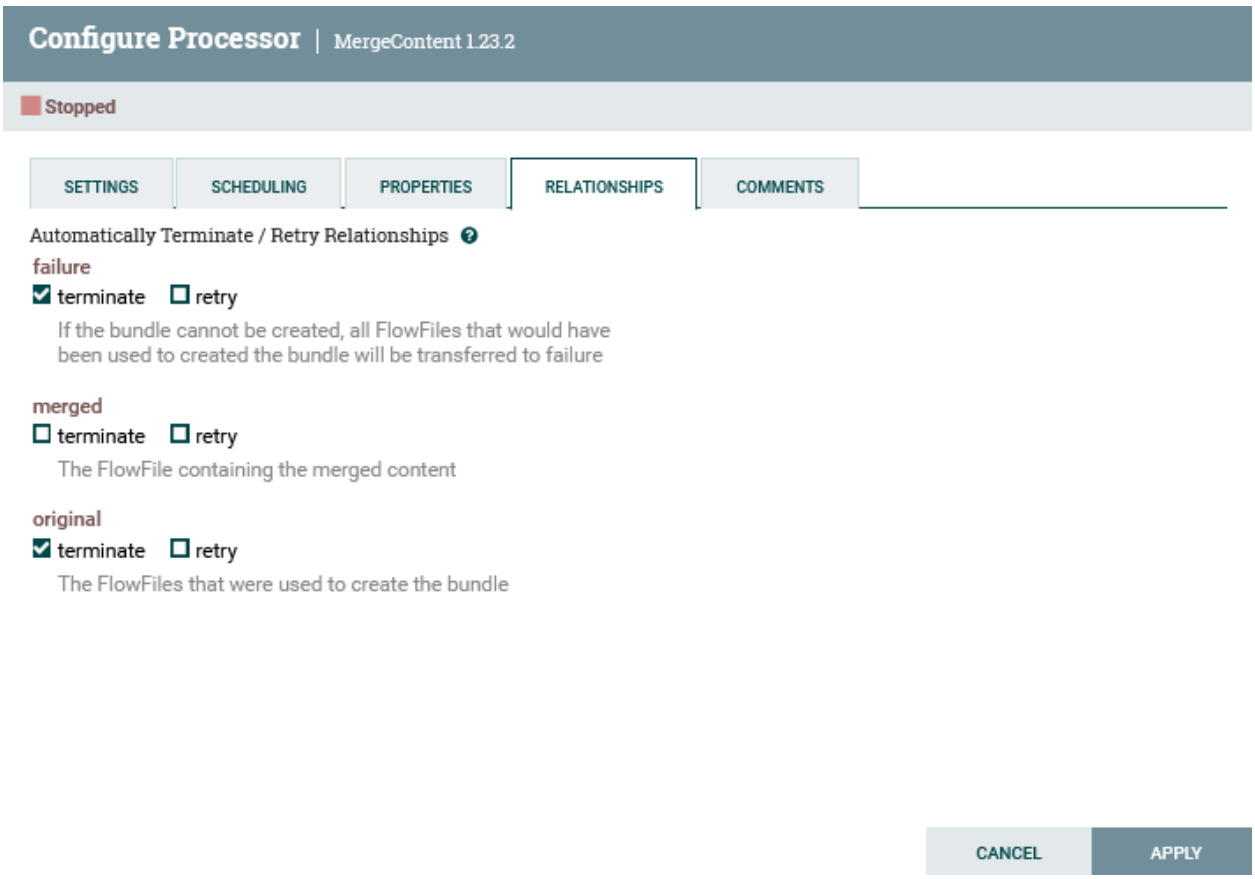

Configure Processor | QueryRecord 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Required field
Property
Value
Record Reader
CSVReader
Record Writer
CSVRecordSetWriter
Include Zero Record FlowFiles
false
Cache Schema
true
Default Decimal Precision
10
Default Decimal Scale
0
price_notnull
SELECT * FROM FLOWFILE WHERE price is not null
price_null
SELECT * FROM FLOWFILE WHERE price is null
CANCEL
APPLY

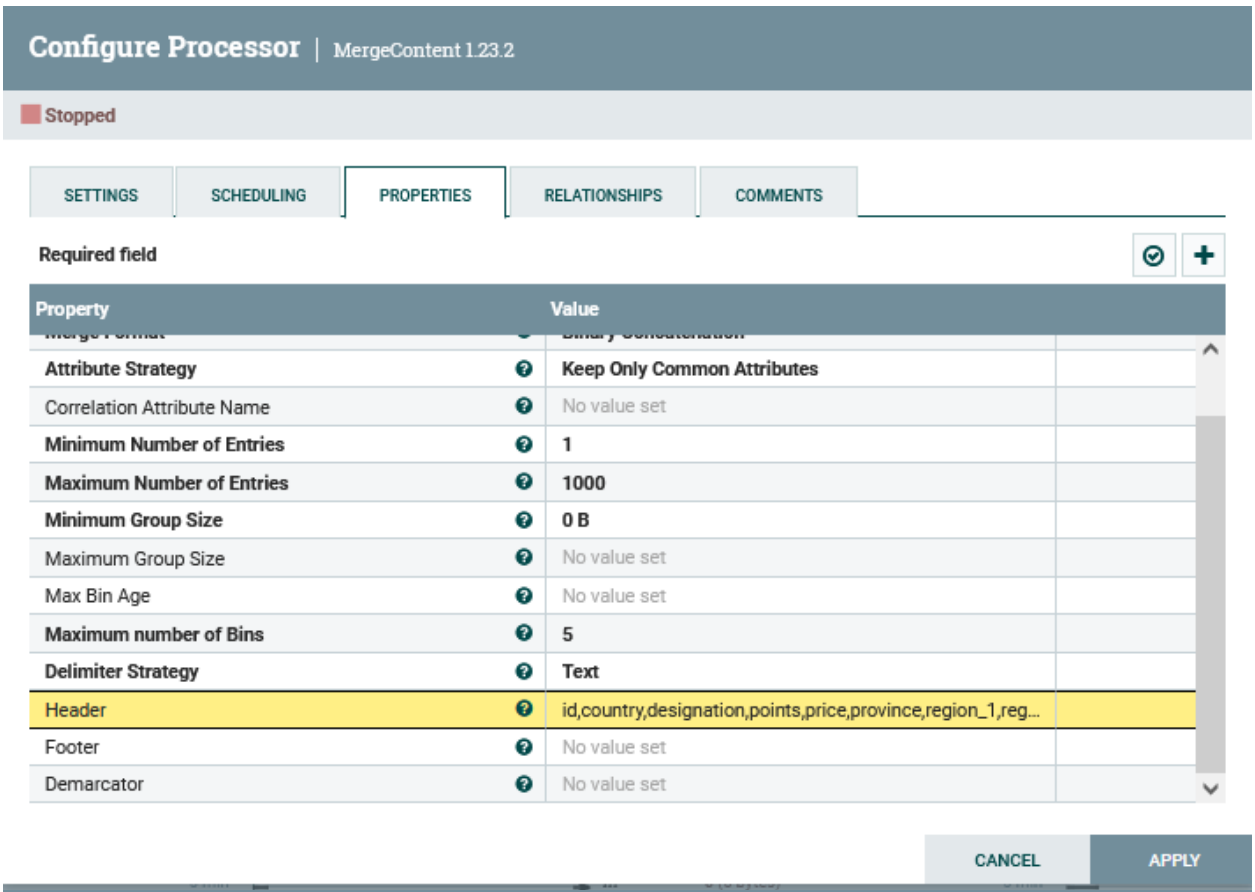

Configure Processor | QueryRecord 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Automatically Terminate / Retry Relationships
failure
terminate
retry
If a FlowFile fails processing for any reason (for example, the SQL statement contains columns not present in input data), the original FlowFile it will be routed to this relationship
original
terminate
retry
The original FlowFile is routed to this relationship
price_notnull
terminate
retry
User-defined relationship that specifies where data that matches the specified SQL query should be routed
price_null
terminate
retry
User-defined relationship that specifies where data that
CANCEL
APPLY

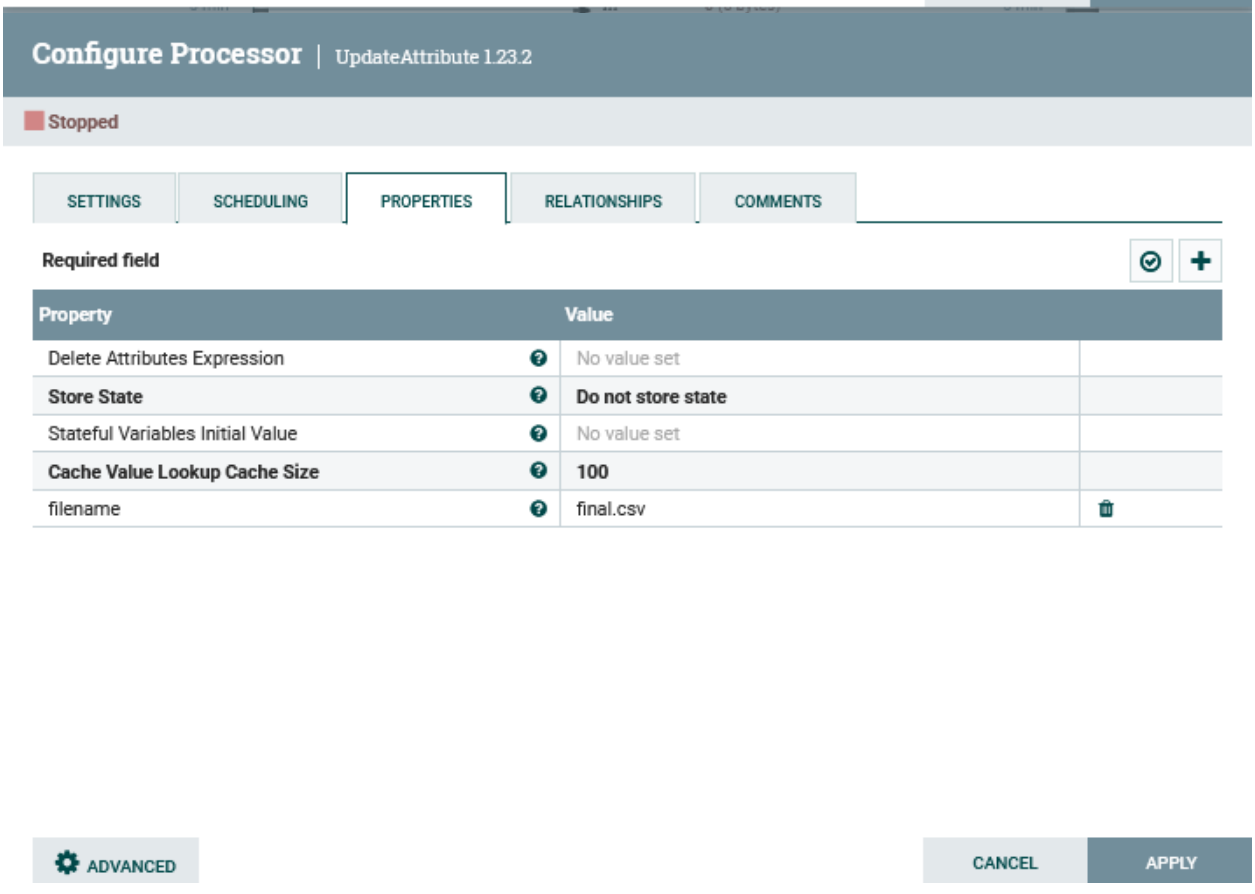

Configure Processor | UpdateRecord 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Required field
Property
Value
Record Reader
CSVReader
Record Writer
CSVRecordSetWriter
Replacement Value Strategy
Literal Value
/price
0.0
CANCEL
APPLY

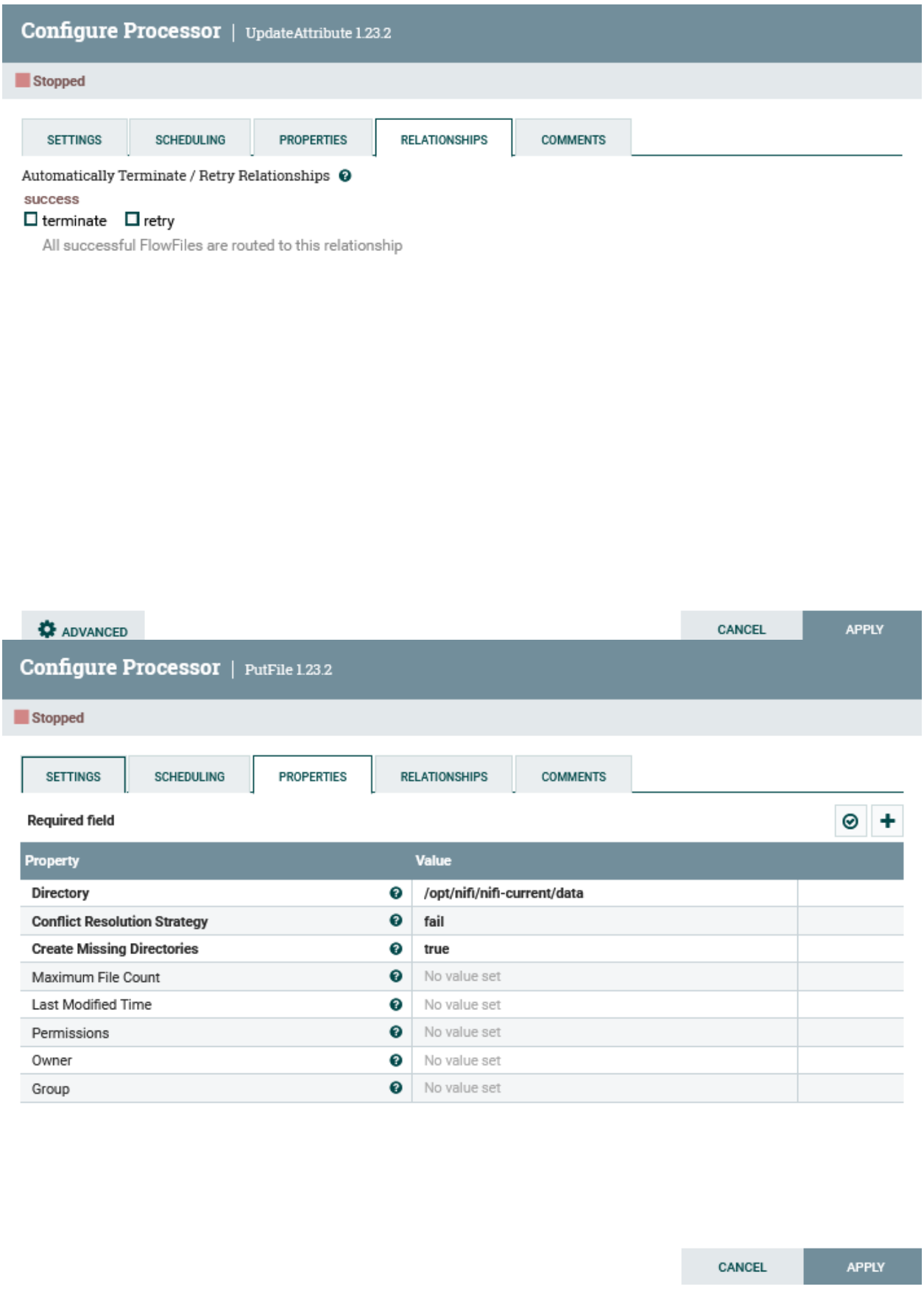

Configure Processor | UpdateRecord 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Automatically Terminate / Retry Relationships
failure
terminate
retry
If a FlowFile cannot be transformed from the configured input format to the configured output format, the unchanged FlowFile will be routed to this relationship
success
terminate
retry
FlowFiles that are successfully transformed will be routed to this relationship
CANCEL
APPLY
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

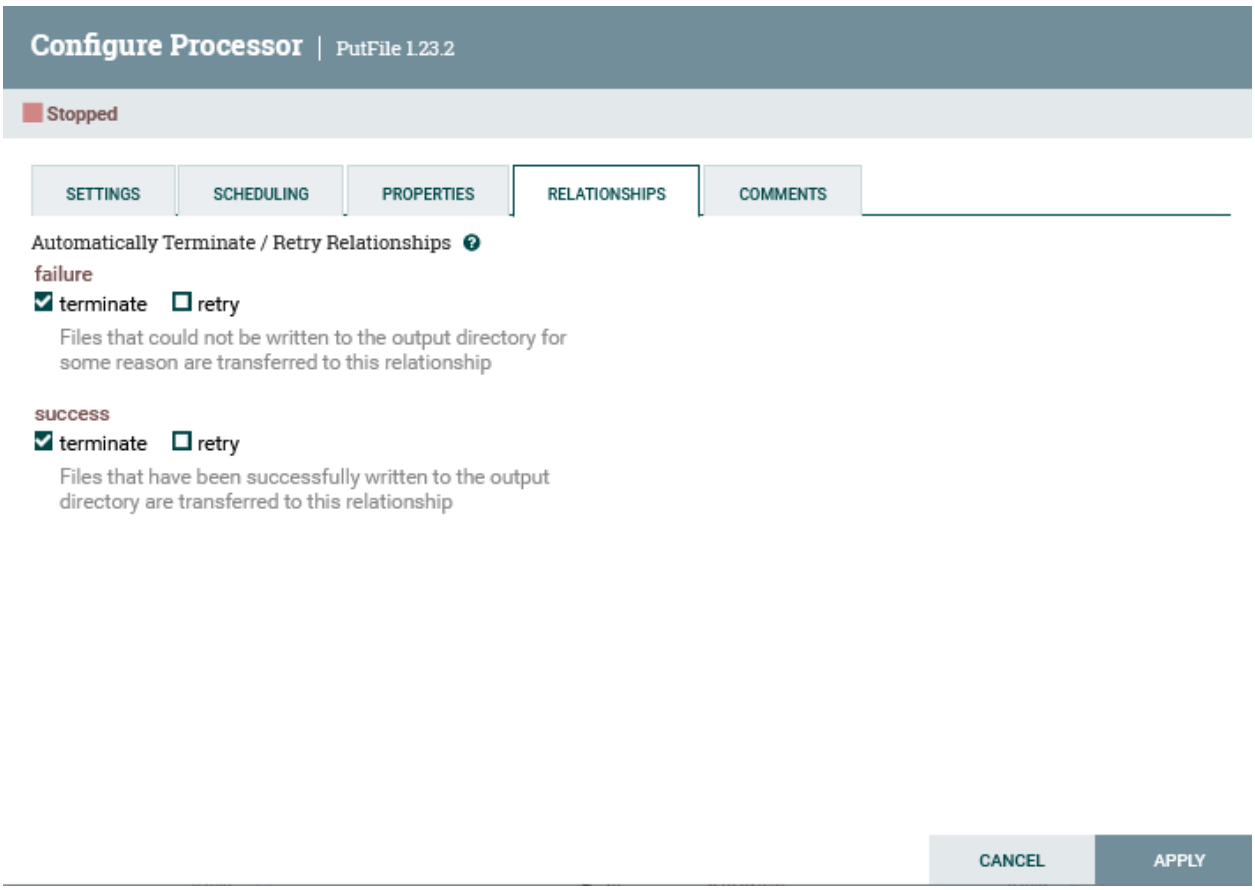

Configure Processor | PutElasticsearchRecord 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Automatically Terminate / Retry Relationships
errors
terminate
retry
If a "Result Record Writer" is set, any Record(s) corresponding to Elasticsearch document(s) that resulted in an "error" (within Elasticsearch) will be routed here.
failure
terminate
retry
All flowfiles that fail for reasons unrelated to server availability go to this relationship.
retry
terminate
retry
All flowfiles that fail due to server/cluster availability go to this relationship.
success
terminate
retry
All flowfiles that succeed in being transferred into
CANCEL
APPLY


Configure Processor | MergeContent 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Automatically Terminate / Retry Relationships
failure
terminate
retry
If the bundle cannot be created, all FlowFiles that would have been used to created the bundle will be transferred to failure
merged
terminate
retry
The FlowFile containing the merged content
original
terminate
retry
The FlowFiles that were used to create the bundle
CANCEL
APPLY

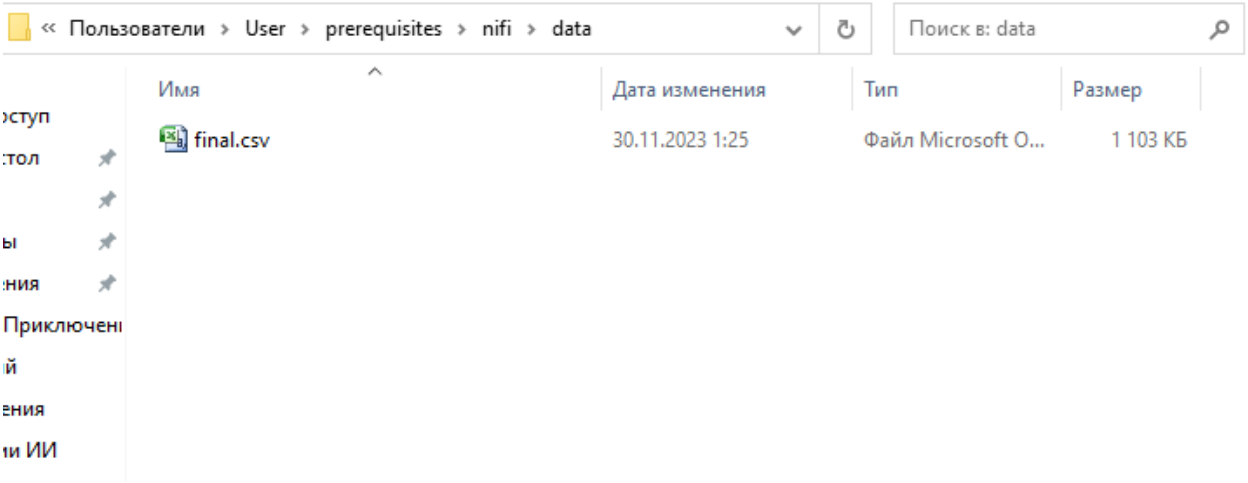

Configure Processor | MergeContent 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Required field
Property
Value
Attribute Strategy
Keep Only Common Attributes
Correlation Attribute Name
No value set
Minimum Number of Entries
1
Maximum Number of Entries
1000
Minimum Group Size
0 B
Maximum Group Size
No value set
Max Bin Age
No value set
Maximum number of Bins
5
Delimiter Strategy
Text
Header
id,country,designation,points,price,province,region_1,reg...
Footer
No value set
Demarcator
No value set
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
final.csv
ADVANCED
CANCEL
APPLY


Configure Processor | UpdateAttribute 1.23.2
Stopped
SETTINGS
SCHEDULING
PROPERTIES
RELATIONSHIPS
COMMENTS
Automatically Terminate / Retry Relationships
success
terminate
retry
All successful FlowFiles are routed to this relationship
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

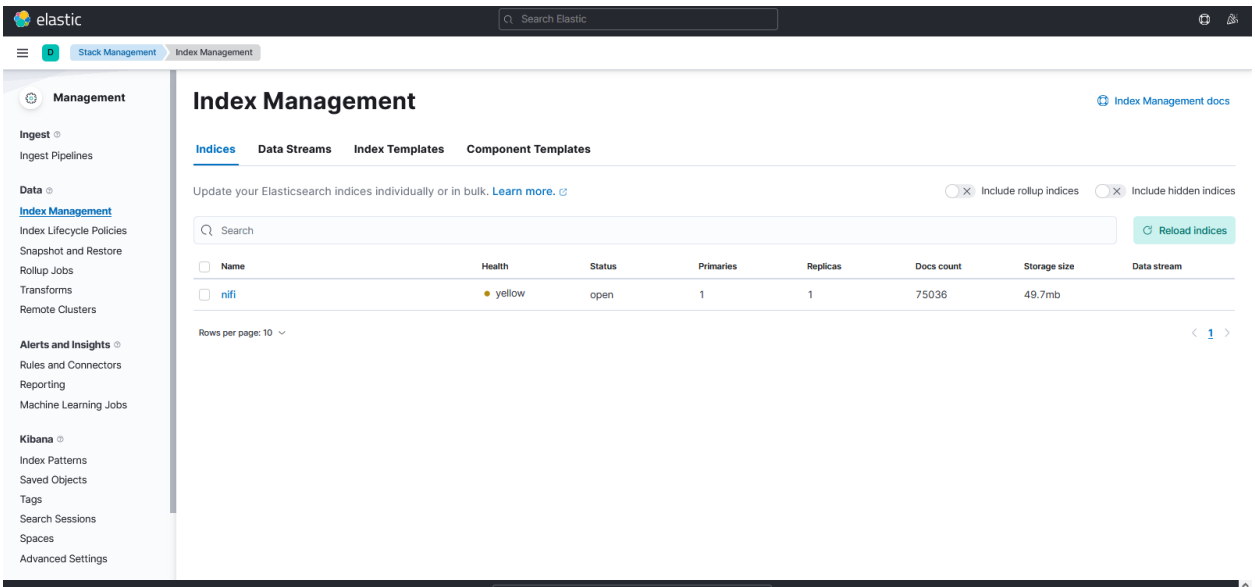


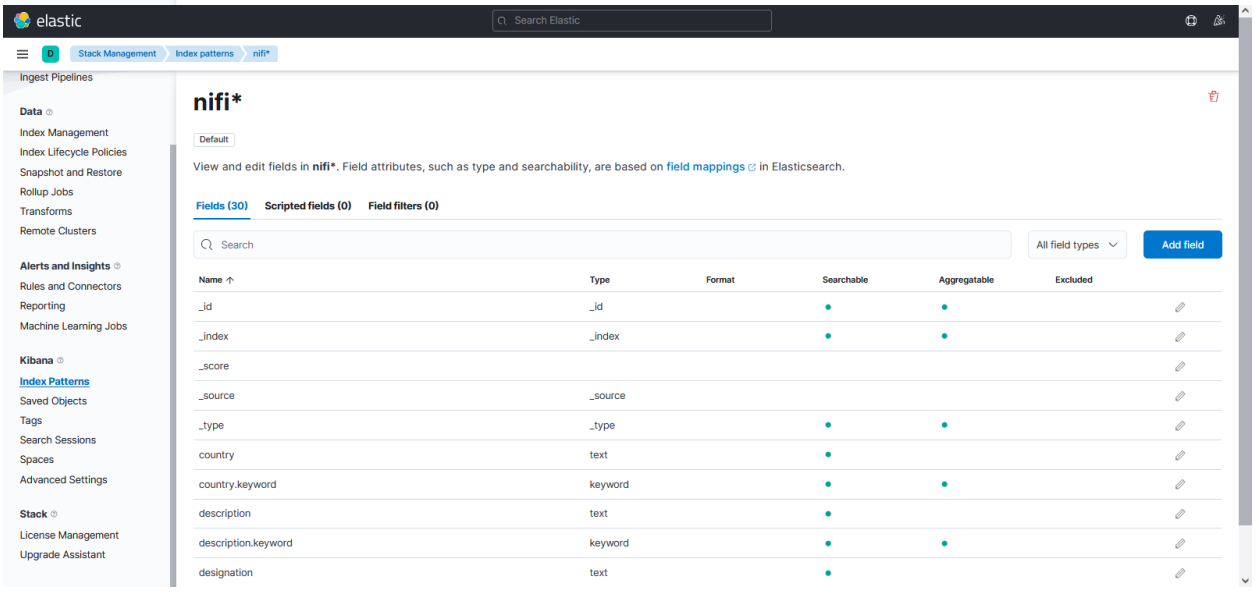
Появившийся итоговый файл в нашей дирректории.



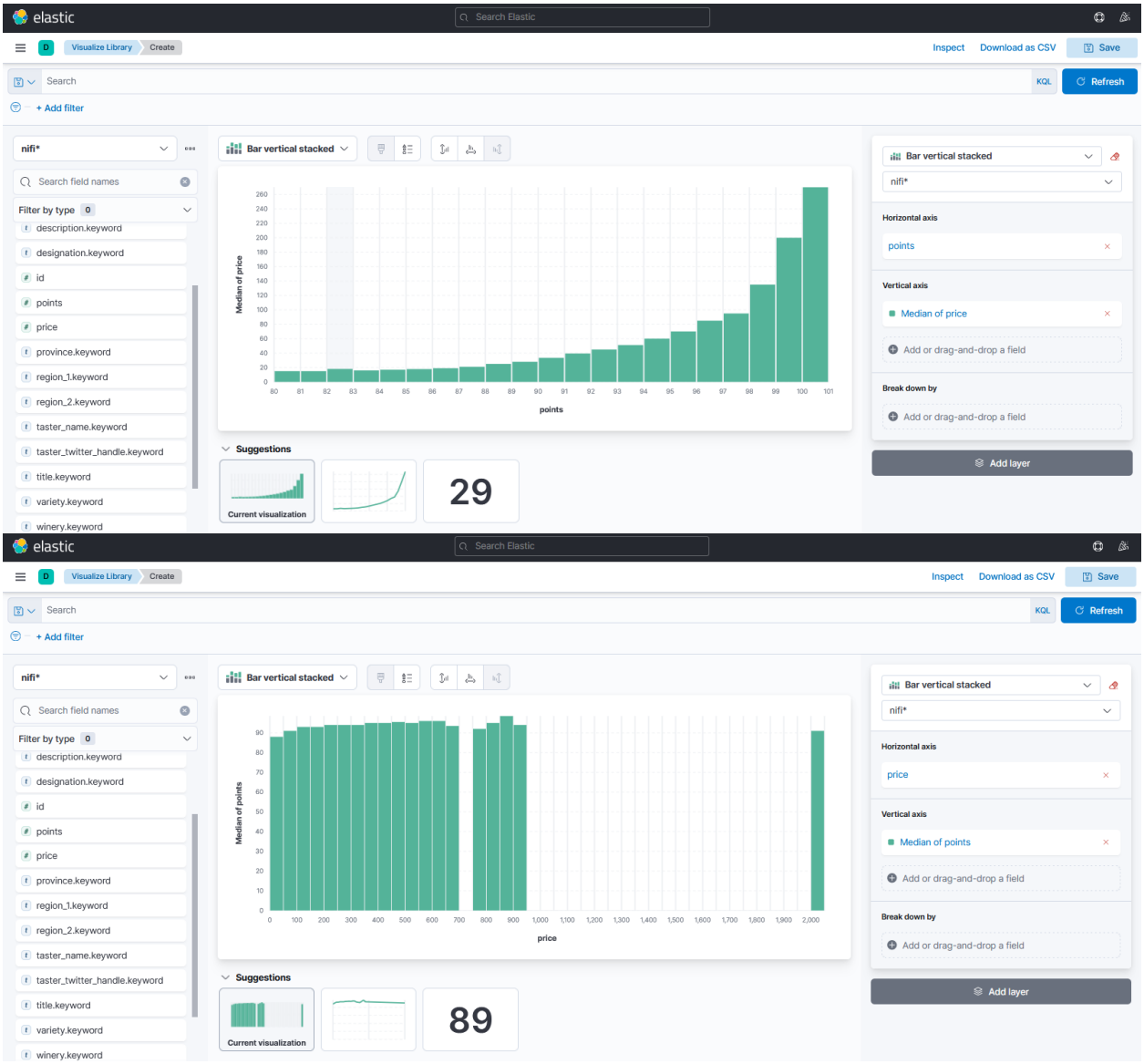
Сама схема будет лежать в формате .xml в моем репозитории.

После заходим в ElasticSearch и проверяем,что данные обработаны верно. Был создан Index с полями внутри.


elastic
Search Elastic
Stack Management
Index Management
Management
Ingest
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
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
75036
49.7mb
Rows per page: 10
1


elastic
Search Elastic
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)
Search
All field types
Add field
Name
Type
Format
Searchable
Aggregatable
Excluded
_id
_index
_score
_source
_type
country
country.keyword
description
description.keyword
designation

Построенны несколько графиков с использованными данными.



3. Apache Airflow

Apache Airflow – это платформа для создания пайплайнов. Для генерации которых надо написать код и запустить его. Код представлен далее и будет лежать в моем репозитории. В коде представлены функции для предворительной предобработки данных и загрузки в ElasticSearch. CSV файлы были перенесены в папку, к которой **Airflow** имеет доступ, аналогично Nifi. Ниже представлен код Dag. Данный код перенесен в папку dags как можно видеть, для того, чтобы в интерфейсе Airflow он был виден.

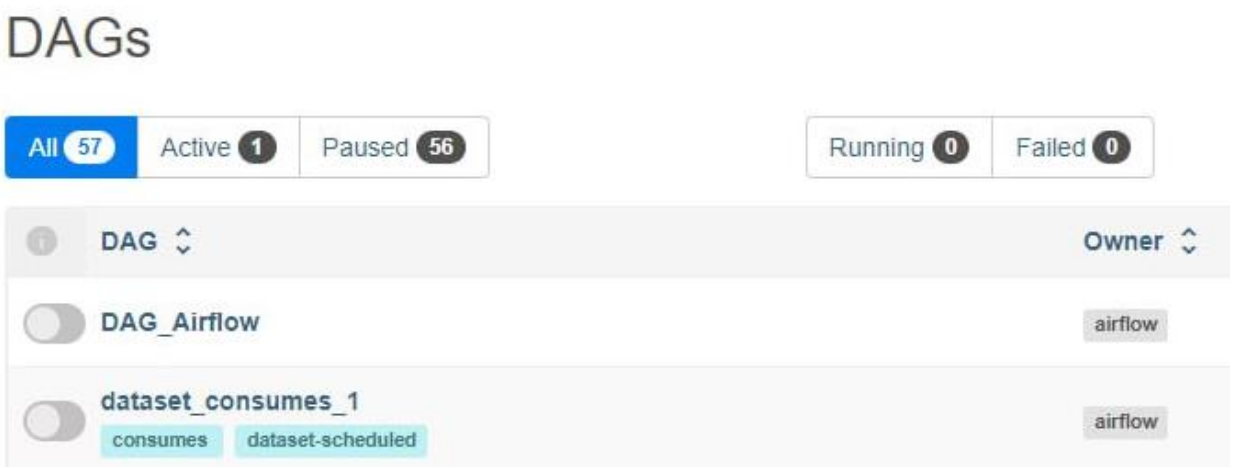
pip.py 3 ×

C: > Users > User > prerequisites > airflow > dags > pip.py > ...

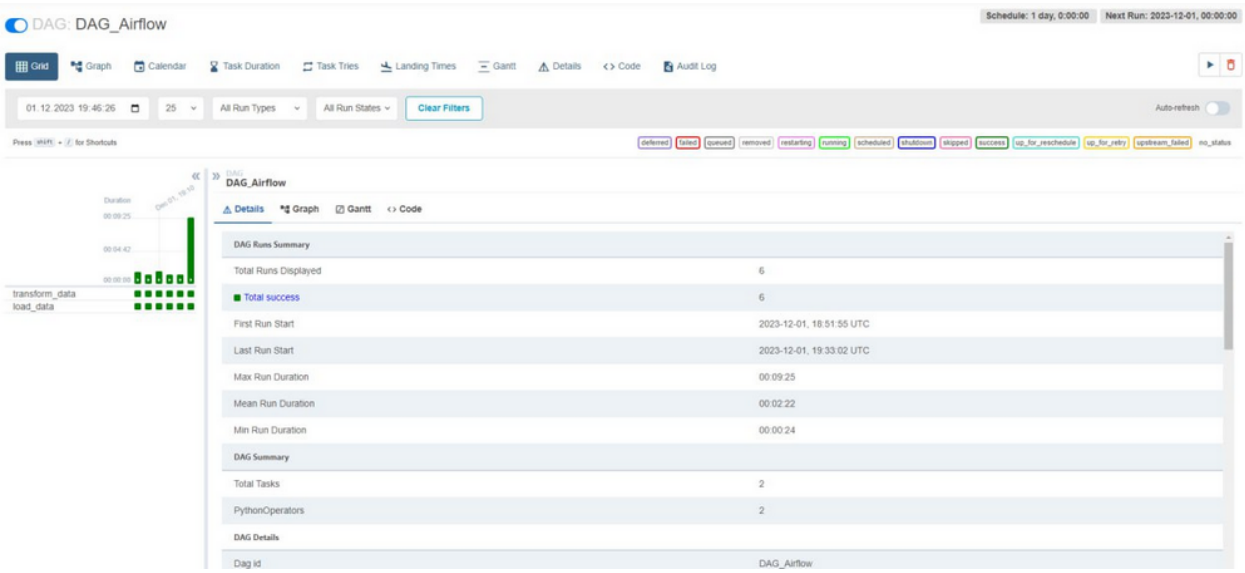
```python
from airflow import DAG
from airflow.operators.python import PythonOperator
from elasticsearch import Elasticsearch
from datetime import datetime
import pandas as pd
import numpy as np

with DAG('DAG_Airflow',
         schedule_interval=None,
         start_date=datetime(2023, 12, 1),
         catchup=False) as dag:

    def transform_data():
        result = pd.DataFrame()
        for i in range(26):
            result = pd.concat([result, pd.read_csv(f"/opt/airflow/data/chunk{i}.csv")])
        result = result[result['region_1'].notnull()]
        result = result[result['designation'].notnull()]
        result['price'] = result['price'].replace(np.nan, 0)
        result = result.drop(['id'], axis=1)
        result.to_csv('/opt/airflow/data/data.csv', index=False)
```

```python
def load_data():
    elastic = Elasticsearch("http://elasticsearch-kibana:9200")
    data = pd.read_csv('/opt/airflow/data/data.csv')
    data = data.fillna('')
    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"],
            "taster_name": row["taster_name"],
            "taster_twitter_handle": row["taster_twitter_handle"],
            "title": row["title"],
            "variety": row["variety"],
            "winery": row["winery"],
        }

        elastic.index(index="wines", id=i, body=doc)

transform_data = PythonOperator(
    task_id='transform_data',
    python_callable=transform_data,
    dag=dag
)

load_data = PythonOperator(
    task_id='load_data',
    python_callable=load_data,
    dag=dag
    )

transform_data >> load_data
```

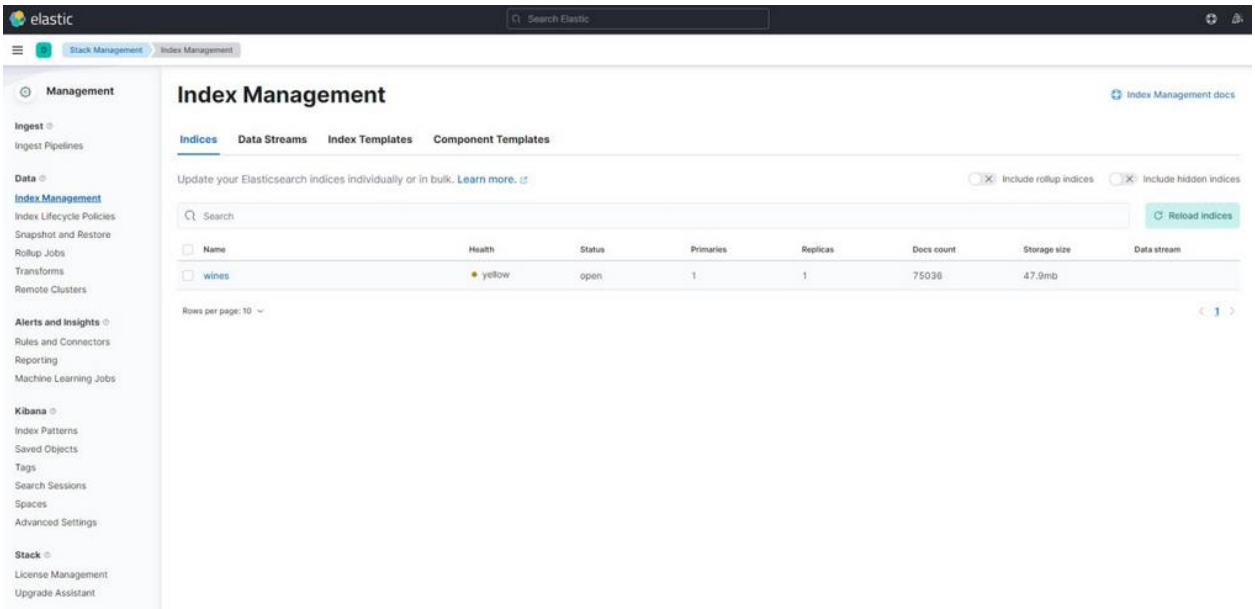
Интерфейс Airflow с загруженным Dag с названием DAG_Airflow.



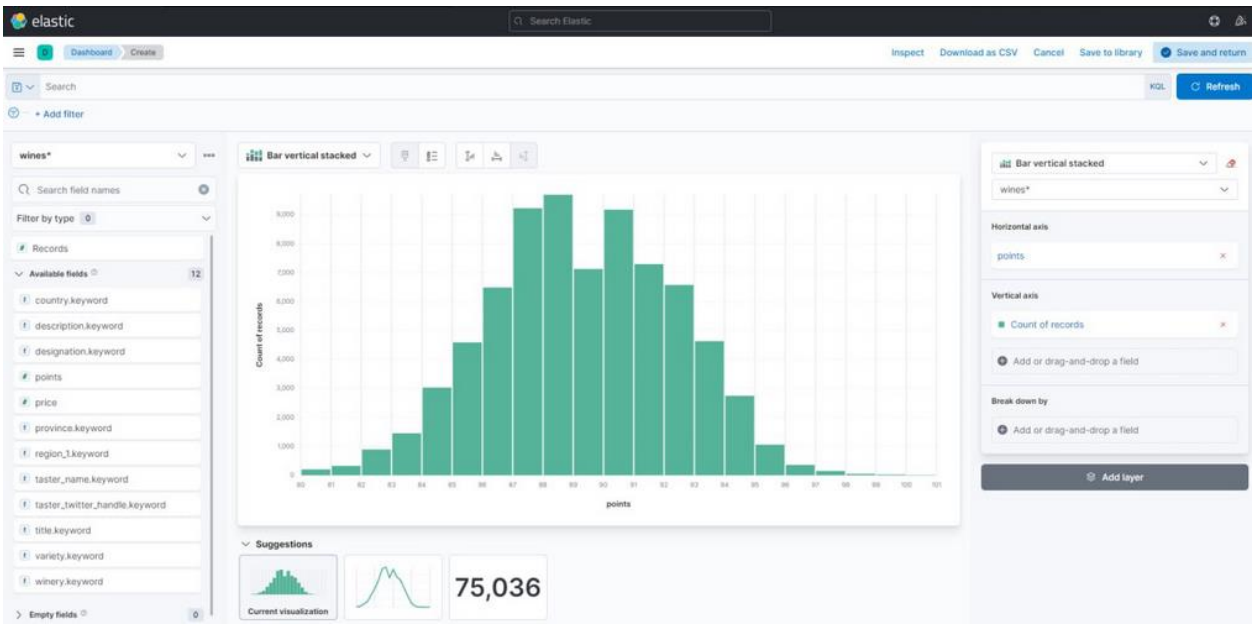
Отработанный Dag:



Создал Index для набора данных из Airflow с названием wines*:



Визуализация в ElasticSearch для данных из Airflow:



ЗАКЛЮЧЕНИЕ

В результате выполнения лабораторной работы были получены навыки по работе с Docker, Apache Airflow, Apache NiFi, ElasticSearch. Была создана схема пайплайна в Apache NiFi, код для генерации данных в Apache Airflow и использован ElasticSearch для визуализации данных.