## Задача

**Анализ публикуемых новостей**

**Общая задача:**создать ETL-процесс формирования витрин данных для анализа публикаций новостей.

**Подробное описание задачи:**

* Разработать скрипты загрузки данных в 2-х режимах:

o   Инициализирующий – загрузка полного слепка данных источника

o   Инкрементальный – загрузка дельты данных за прошедшие сутки

* Организовать правильную структуру хранения данных

o   Сырой слой данных

o   Промежуточный слой

o   Слой витрин

В качестве **результата работы** программного продукта необходимо написать скрипт, который формирует витрину данных следующего содержания

* Суррогатный ключ категории
* Название категории
* Общее количество новостей из всех источников по данной категории за все время
* Количество новостей данной категории для каждого из источников за все время
* Общее количество новостей из всех источников по данной категории за последние сутки
* Количество новостей данной категории для каждого из источников за последние сутки
* Среднее количество публикаций по данной категории в сутки
* День, в который было сделано максимальное количество публикаций по данной новости
* Количество публикаций новостей данной категории по дням недели

**Источники:**

* <https://lenta.ru/rss/>
* <https://www.vedomosti.ru/rss/news>
* <https://tass.ru/rss/v2.xml>

## Выбор стека технологий

Apache Airflow — это надежный планировщик для программного создания, планирования и мониторинга рабочих процессов. Он предназначен для обработки и организации сложных конвейеров данных. Первоначально он был разработан для решения проблем, связанных с долгосрочными задачами cron и существенными скриптами, но он превратился в одну из самых мощных платформ конвейера данных на рынке.

Мы можем описать Airflow как платформу для определения, выполнения и мониторинга рабочих процессов. Мы можем определить рабочий процесс как любую последовательность шагов, которые вы предпринимаете для достижения определенной цели. Распространенной проблемой, возникающей в растущих командах по работе с большими данными, является ограниченная возможность объединить связанные задания в непрерывный рабочий процесс. До Airflow был Oozie, но у него было много ограничений, но Airflow превзошел его для сложных рабочих процессов.

Для решения задачи воспользуемся языком программирования Python. Это позволит запускать обрабтку лога прямо на web-сервере. Так же Python имеет мощные встроенные средства для работы с текстом и обширный набор подключаемых модулей.

В качестве хранилища данных для формирования витрины данных можно исползовать что угодно, но аналитики любят SQL. Поэтому используем PostgreSQL.

Применение более развитых технологий обрабтки данных (Spark, Hadoop и пр.) целесообразно только при необходимости обработки логов размером в 100 Гб и более. Применение компилируемых языков программирования (C, C#, Java, Scala и пр.) может позволить ускорить обработку данных, но тогда може возникнуть проблема оперативной модификации алгоритма обратки и устанвоки дополнительных библиотек, языковых средств.

## Реализация скрипта обработки логов

### Алгоритм работы

1. Спарсить данные из источников
2. Выполнить разбор данных по определенным тегам
3. Выполнить разбор полей
4. Выполнить обработку разобранной строки
5. Выполнить обобщение обработанных строк
6. Записать результат в файл

### Порядок работы

* 1. Поднять докер-контейнеры: все данные для этого представлены в файле: docker-compose.yaml
  2. Подключить необходимые библиотеки: данные в файле dockerfile, библиотеки в requirements.txt
  3. Создать DAG: airflow1t.py

Парсинг данных выполняется с использованием библиотеки feedparser.

Создаем следующие библиотеки:

parseRSS # функция получает линк на рсс ленту, возвращает распаршенную ленту с помощью feedpaeser

getHeadlines # функция для получения заголовков новости

getCategories # функция для получения описания новости

getLinks # функция для получения ссылки на источник новости

getDates # функция для получения даты публикации новости

Далее создаем списки для записи данных:

allheadlines = [] #заголовки  
allcategories = [] #категории  
alllinks = [] #ссылки  
alldates = [] #даты публикации

Функция для записи данных в csv файл:

write\_all\_news

Все новости записываются в csv файл:

f\_all\_news = 'allnews.csv'

Также реализована возможность записи новостей по нужным нам тегам. Они записываются в csv файл:

f\_certain\_news = 'certainnews.csv' #файл с новостями по ключевым словам  
vector1 = 'ДолЛАР|РубЛ|ЕвРО' # пример таргетов  
vector2 = 'ЦБ|СбЕРбАНК|курс'

Все данные функции реализуются в таске:

extract\_data

В Airflow в XCom передаются все данные файла f\_all\_news = 'allnews.csv’ . Они записываются в датафрейм df.

Далее в таске transform\_data выводятся строки данного датафрейма.

В таске load\_data происходит процесс загрузки данных в Posrgresql. В БД создается таблица под названием allnews со столбцами Title (Заголовки новостей), Category (Название категории), Links (Ссылка новости), Publication\_date (Время публикации новости).

## Формирование витрины данных

Получение данных из БД возможно с помощью запросов