

Лабораторная работа 3. Программные средства консолидации данных из различных источников с использованием Python и Apache Airflow

Цель: научиться работать с Apache Airflow для автоматизации процессов ETL (Extract, Transform, Load). На практике освоить настройку и выполнение DAG в Airflow для извлечения данных из различных форматов (CSV, Excel, JSON), их обработки на Python, загрузки в базу данных SQLite и отправки уведомлений по электронной почте.

Оборудование и ПО:

- Ubuntu (с Docker)
- Apache Airflow
- SQLite
- Python
- Docker
- email сервис

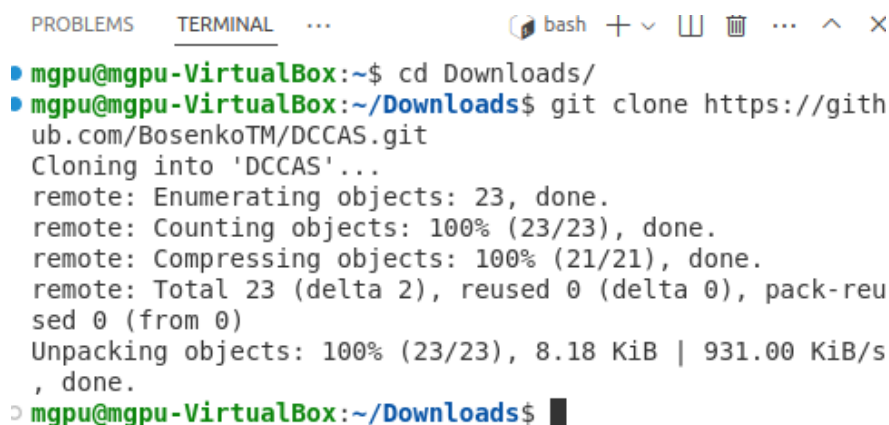
Исходные данные:

- Набор файлов CSV, Excel и JSON, содержащих данные для обработки.
- Конфигурация email для отправки уведомлений.

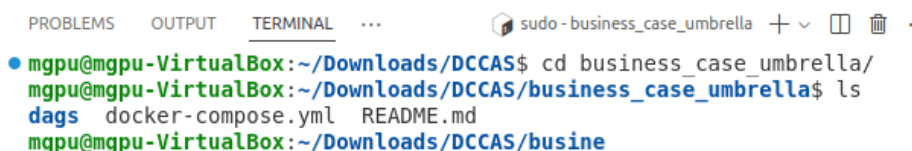
Ход работы

I. Развернуть ВМ *ubuntu_mgpu.ova* в *VirtualBox*. +

II. Клонировать на ПК *Umbrella* в домашний каталог ВМ.



```
PROBLEMS  TERMINAL  ...  bash + v [ ] [ ] ... ^ x
● mgpu@mgpu-VirtualBox:~$ cd Downloads/
● mgpu@mgpu-VirtualBox:~/Downloads$ git clone https://github.com/BosenkoTM/DCCAS.git
Cloning into 'DCCAS'...
remote: Enumerating objects: 23, done.
remote: Counting objects: 100% (23/23), done.
remote: Compressing objects: 100% (21/21), done.
remote: Total 23 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (23/23), 8.18 KiB | 931.00 KiB/s, done.
○ mgpu@mgpu-VirtualBox:~/Downloads$
```



```
PROBLEMS  OUTPUT  TERMINAL  ...  sudo-business_case_umbrella + v [ ] [ ] .
● mgpu@mgpu-VirtualBox:~/Downloads/DCCAS$ cd business_case_umbrella/
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella$ ls
dags  docker-compose.yml  README.md
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella$
```

III. Запустить контейнер с кейсом.

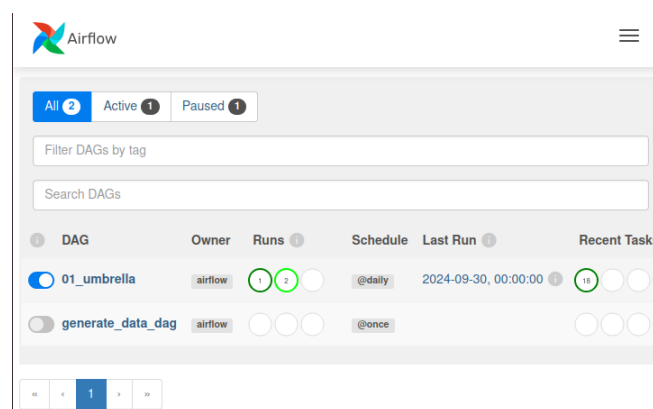
```

mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella$ sudo docker compose up -d
[+] Running 4/5
  .: Network business_case_umbrella_default        Created           3.4s
  ✓ Container business_case_umbrella-postgres-1    Started            1.3s
  ✓ Container business_case_umbrella-init-1         Started            2.5s
  ✓ Container business_case_umbrella-scheduler-1    Started            3.0s
  ✓ Container business_case_umbrella-webserver-1    Started            3.0s
mgpu@mgpu-VirtualBox:~/Downloads/DCCAS/business_case_umbrella$

```

IV. Изучить и описать основные элементы интерфейса Apache Airflow.

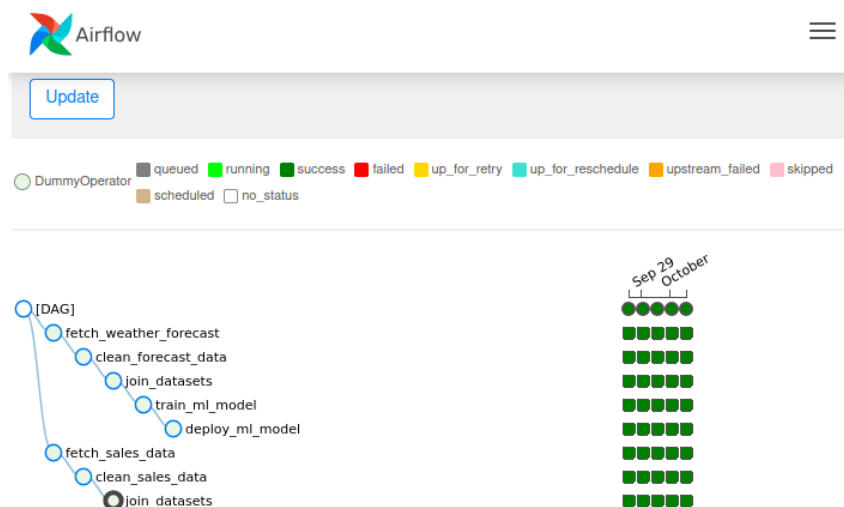
Основные элементы интерфейса **Apache Airflow**, представленные на скриншоте:



- Навигационная панель (с логотипом Airflow):** Расположена в верхней части экрана и служит для переключения между различными разделами интерфейса, такими как список DAGs (Directed Acyclic Graphs), задачи, журналы и т. д.
- Фильтры и поиск:**
 - Включает вкладки:
 - All: отображает все DAGs (2 DAG-а на скриншоте).
 - Active: показывает только активные DAGs (1 активный DAG).
 - Paused: показывает DAGs, которые приостановлены.
 - Filter DAGs by tag и Search DAGs: эти поля позволяют фильтровать DAGs по тегам или искать их по имени.
- Список DAGs:**
 - Таблица с DAGs включает несколько колонок:
 - DAG: имя DAG. Это 01_umbrella и generate_data_dag.

- Owner: имя владельца DAG. Владелец обоих DAGs — airflow.
- Runs: отображает количество выполнений DAG. В колонке есть индикаторы для текущего и завершенных запусков.
- Schedule: указывает расписание для выполнения DAG. @daily указывает, что DAG 01_umbrella запускается ежедневно.
- Last Run: показывает время последнего запуска DAG.
- Recent Tasks: отображает текущий статус последних задач DAG. Зеленый кружок указывает на успешное выполнение задачи.

4. Переключатель состояния DAG: Справа от имени каждого DAG есть переключатель, позволяющий включать или отключать DAG (запускать или приостанавливать его выполнение).



Основные элементы интерфейса:

- 1. Кнопка Update:** позволяет обновить статус задач в DAG, чтобы видеть актуальную информацию.
- 2. Легенда:** показывает различные состояния задач, которые отображаются с помощью цветных индикаторов:
 - queued (серый) — задача поставлена в очередь.
 - running (салатовый) — задача выполняется.
 - success (зеленый) — задача успешно выполнена.
 - failed (красный) — задача завершилась неудачей.
 - up_for_retry (желтый) — задача находится в процессе повторной попытки.
 - up_for_reschedule (бирюзовый) — задача будет перенесена.
 - upstream_failed (оранжевый) — не выполнена предыдущая задача.
 - skipped (розовый) — задача была пропущена.
 - scheduled (бежевый) — задача запланирована.
 - no_status (пустой) — нет статуса для задачи.
- 3. Графическая визуализация DAG:**
 - Показана структура DAG, представляющая собой дерево задач.
 - На скриншоте видно две ветки:

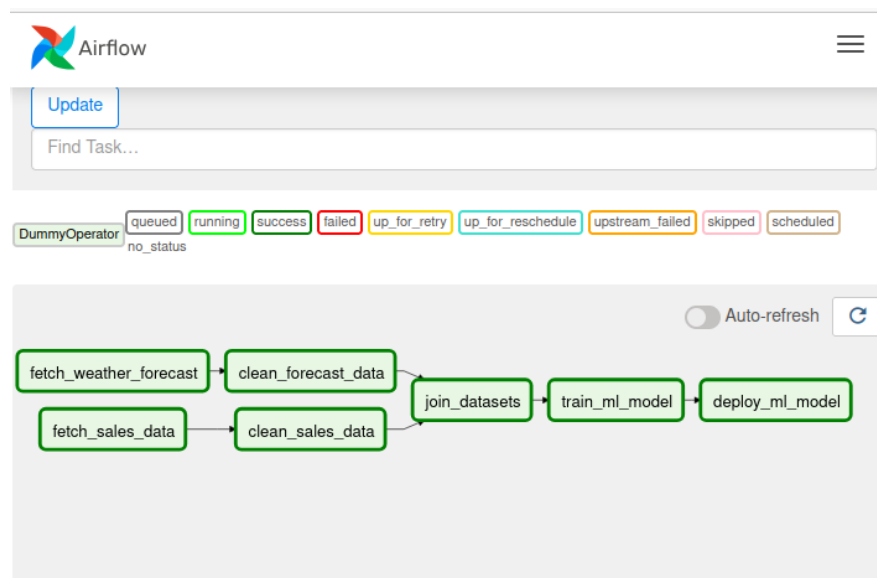
- Первая ветка начинается с задачи `fetch_weather_forecast`, затем задачи `clean_forecast_data`, `join_datasets`, `train_ml_model`, и `deploy_ml_model`.
- Вторая ветка начинается с задачи `fetch_sales_data`, затем задачи `clean_sales_data`, и завершается на задаче `join_datasets`.
- Связи между задачами показаны стрелками, указывающими порядок выполнения.

4. Календарь выполнения:

- Справа представлены круговые индикаторы, которые показывают выполнение DAG по дням. Каждый круг символизирует запуск DAG в определенный день.
- Зеленые индикаторы показывают, что все задачи DAG были успешно выполнены в указанный день.

5. Статусы задач:

- На данном графике видно, что большинство задач успешно выполнены (зеленые кружки).



Вкладка **Graph View** в Apache Airflow предоставляет графическое представление потока выполнения DAG (Directed Acyclic Graph). Она отображает все задачи в DAG, их зависимости и текущий статус выполнения. Основные элементы и функции этой вкладки включают:

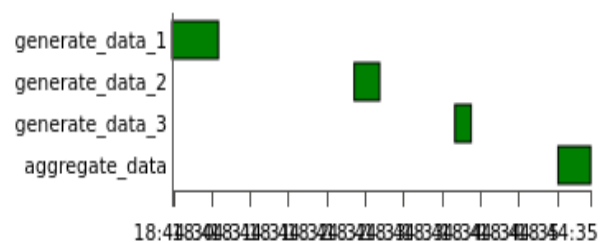
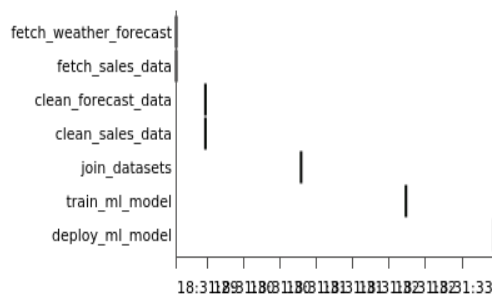
1. Графическая схема DAG:

- Задачи представлены в виде узлов (прямоугольников).
- Зависимости между задачами изображаются стрелками, которые показывают, какая задача зависит от выполнения предыдущей задачи.
- Задачи выполняются последовательно, в соответствии с направлением стрелок, начиная с корневой задачи и двигаясь по графу.

2. Цветовые индикаторы: цвет задач указывает их текущий статус:

- `queued` (серый) — задача поставлена в очередь.
- `running` (салатовый) — задача выполняется.
- `success` (зеленый) — задача успешно выполнена.
- `failed` (красный) — задача завершилась неудачей.

- `up_for_retry` (желтый) — задача находится в процессе повторной попытки.
 - `up_for_reschedule` (бирюзовый) — задача будет перенесена.
 - `upstream_failed` (оранжевый) — не выполнена предыдущая задача.
 - `skipped` (розовый) — задача была пропущена.
 - `scheduled` (бежевый) — задача запланирована.
 - `no_status` (пустой) — нет статуса для задачи.
3. **Увеличение/уменьшение масштаба и перемещение:** можно увеличивать или уменьшать масштаб схемы, а также перемещать граф в пределах окна просмотра.
 4. **Интерактивность:**
 - Нажав на узел задачи, можно увидеть дополнительную информацию о задаче, такую как логи выполнения, расписание, статус и другие метаданные.
 - Можно инициировать запуск задачи, перезапустить или приостановить выполнение прямо из графического интерфейса.
 5. **Структура DAG:** граф дает наглядное представление о том, как задачи взаимодействуют друг с другом. Это помогает понять порядок выполнения задач и логику обработки данных.



Основные особенности этой вкладки:

1. Диаграмма Ганта:

- **Оси времени:** горизонтальная ось представляет собой временную шкалу, которая показывает время выполнения задач. Она позволяет визуально оценить, как долго выполнялась каждая задача и в какие моменты времени они начинались и заканчивались.
 - **Задачи:** Каждая задача представлена горизонтальной полосой. Начало полосы соответствует времени старта задачи, а её длина указывает на продолжительность выполнения.
2. **Цветовые индикаторы:** подобно другим видам в Airflow, цвет полос задач на диаграмме Ганта показывает их статус.
 3. **Интерактивность:** нажав на любую полосу на диаграмме Ганта, можно получить более детальную информацию о задаче.

Анализ зависимости времени выполнения: диаграмма Ганта помогает анализировать, как задачи DAG выполняются во времени относительно друг друга. Можно увидеть, какие задачи выполнялись параллельно, какие задачи задержали выполнение других задач, и как оптимизировать граф для более эффективного выполнения.

V. Спроектировать верхнеуровневую архитектуру аналитического решения задания Бизнес кейс Umbrella в draw.io. Необходимо использовать:

- **Source Layer** - слой источников данных.
- **Storage Layer** - слой хранения данных.
- **Business Layer** - слой для доступа к данным бизнес-пользователей.

