


← → ↻ localhost:8080/home ☆ 📄 📧 📁 ☰







 Airflow DAGs Security Browse Admin Docs 11:52 UTC AA

DAGs

All 1 Active 1 Paused 0

Filter DAGs by tag

Search DAGs

DAG	Owner	Runs	Schedule	Last Run	Recent Tasks	Actions	Links
 01_umbrella	airflow		@daily	2024-03-02, 11:51:03		   ...	


«

1


»


Showing 1-1 of 1 DAGs


На вкладке "Tree View" в Apache Airflow DAG показана иерархия задач (tasks) . Можно увидеть все задачи в DAG, их зависимости друг от друга и порядок выполнения. Это очень удобно для отслеживания потока выполнения задач и проверки зависимостей между ними.


 Airflow DAGs Security Browse Admin Docs 11:54 UTC AA


DAG: 01_umbrella Umbrella example with DummyOperators.


 Tree View


 Graph View




 Task Duration

 Task Tries

 Landing Times

 Gantt

 Details











<> Code


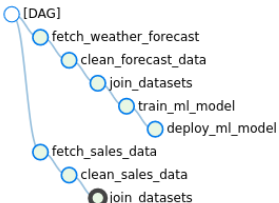
2024-03-02T11:51:03Z

Runs 25

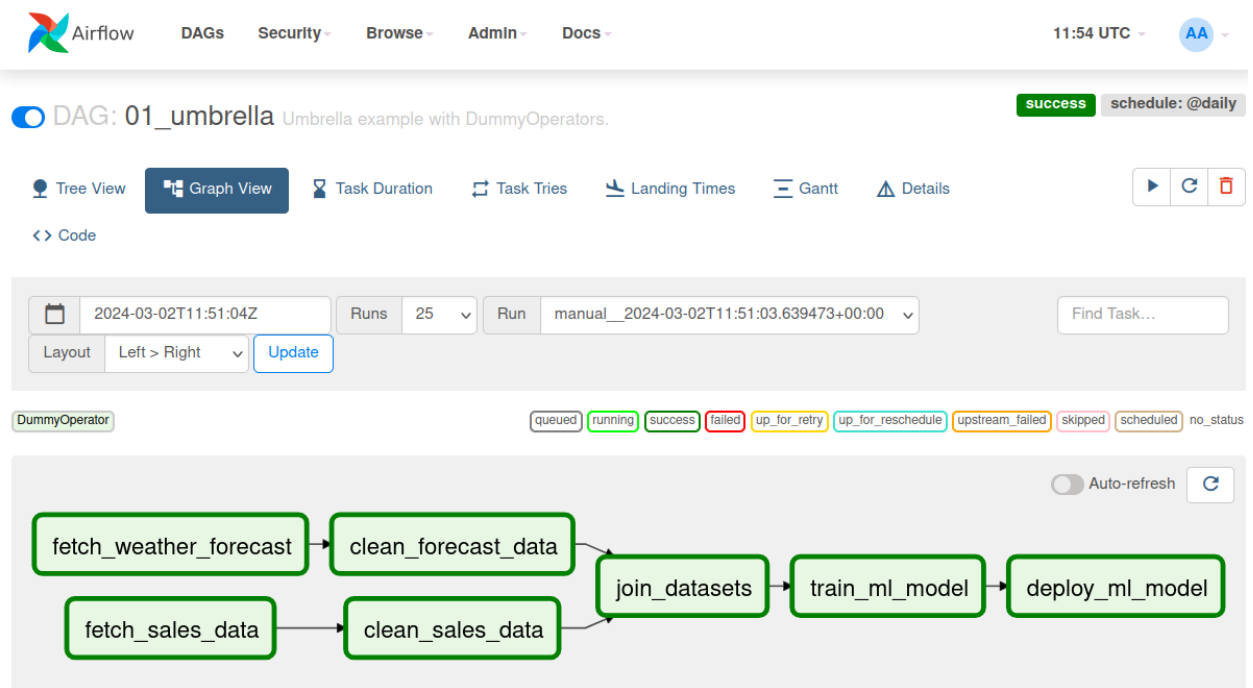
Update

DummyOperator

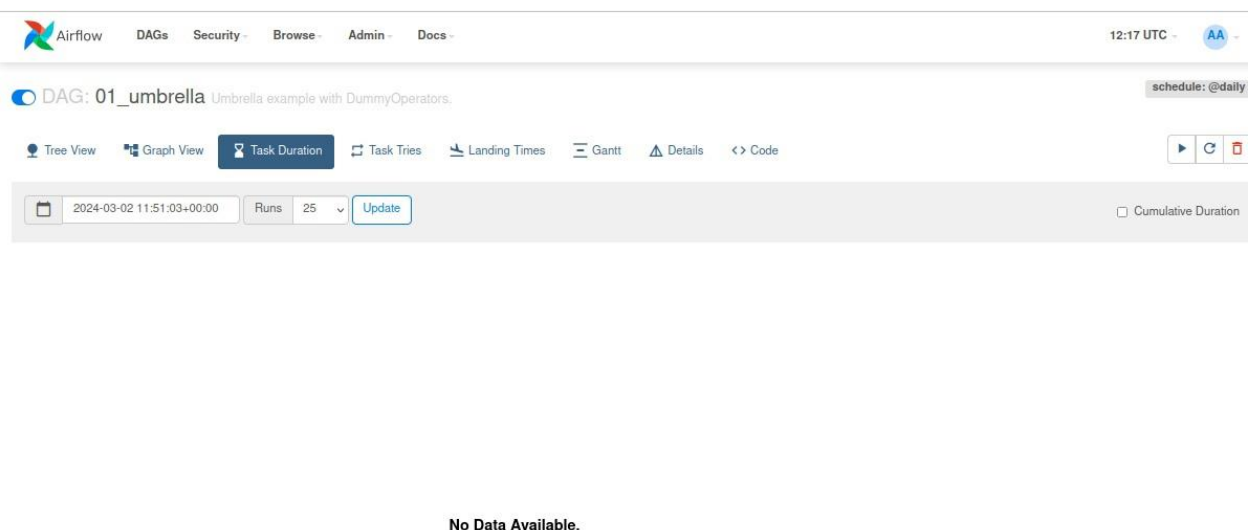
 queued  running  success  failed  up_for_retry  up_for_reschedule  upstream_failed  skipped  scheduled  no_status



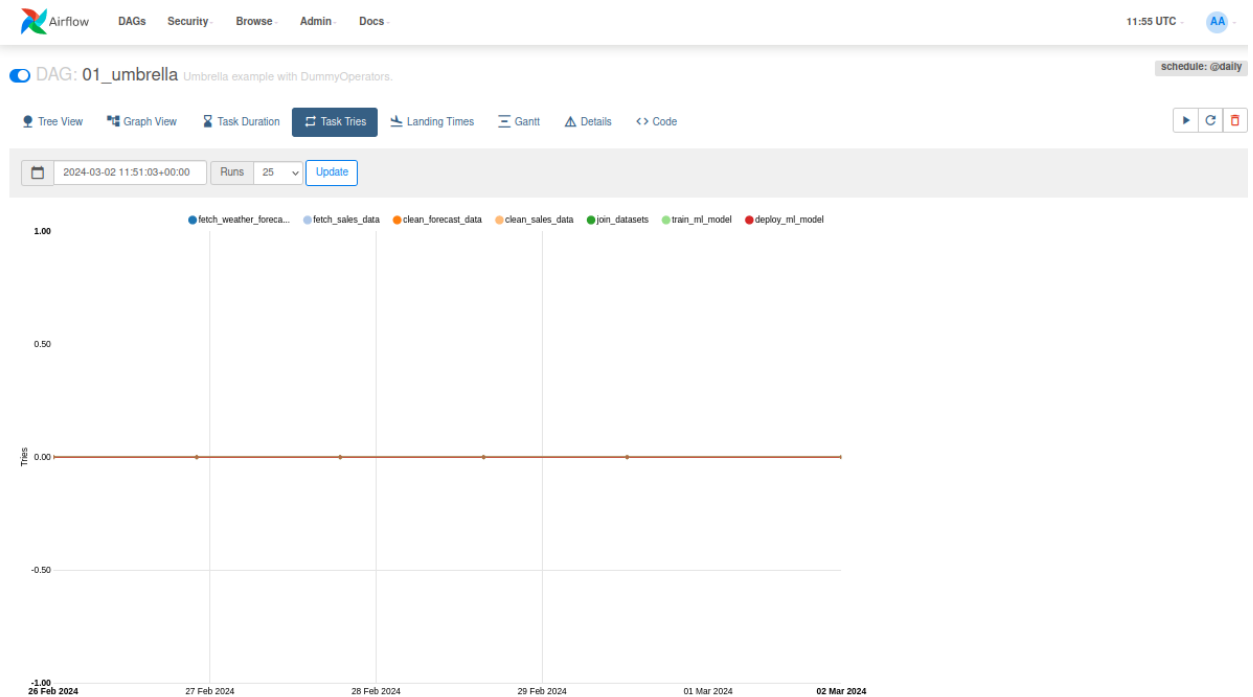
На вкладке "Graph View" в Apache Airflow DAG показана визуализация графа выполнения задач. Это позволяет увидеть структуру DAG, отображая задачи в виде узлов и их зависимости в виде стрелок между узлами. Это очень удобно для визуального анализа порядка выполнения задач в процессе работы.



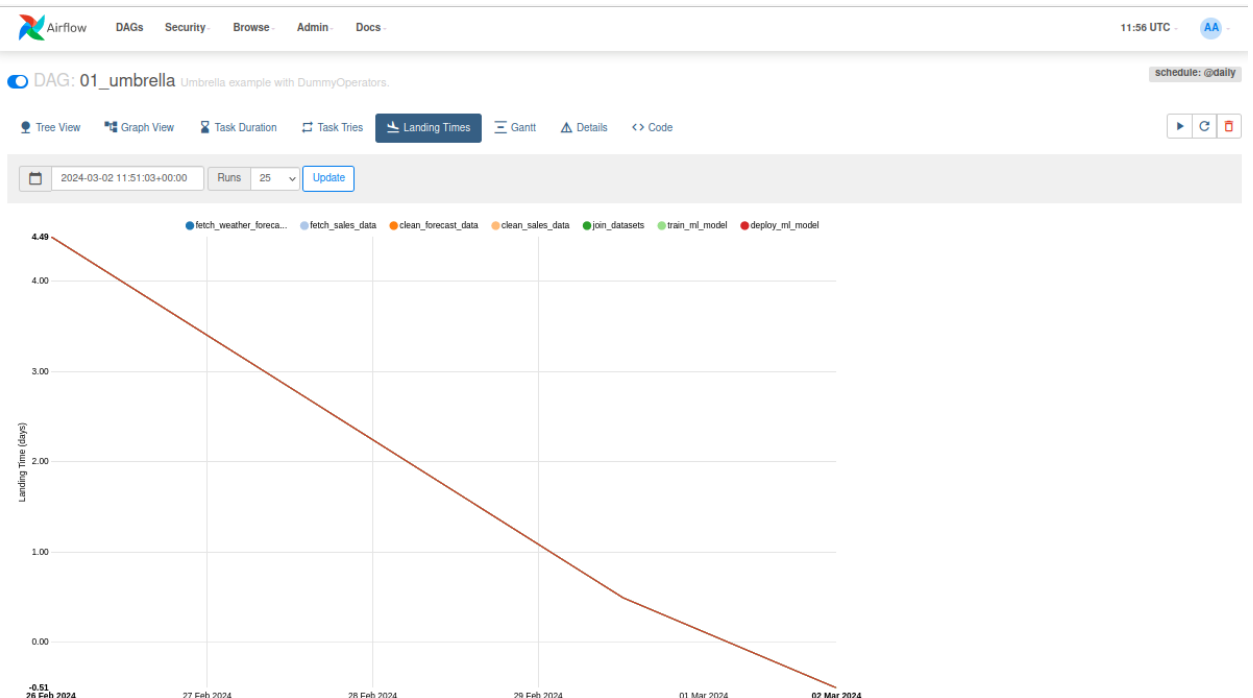
На вкладке "Task Duration" в Apache Airflow DAG показана информация о времени выполнения каждой задачи в режиме реального времени или в прошлом выполнении. Это позволяет оценить, сколько времени занимает выполнение каждой конкретной задачи в рамках DAG. Просматривая эту информацию, можно выявить задачи, которые занимают больше времени, и оптимизировать процессы для улучшения времени выполнения DAG в целом.



На вкладке "Task Trees" в Apache Airflow DAG показана иерархия выполнения задач. Здесь можно увидеть отношения между различными задачами в рамках DAG. Это помогает понять порядок выполнения задач, их зависимости и взаимодействие друг с другом. Это дает более подробное представление о структуре и логике выполнения задач.

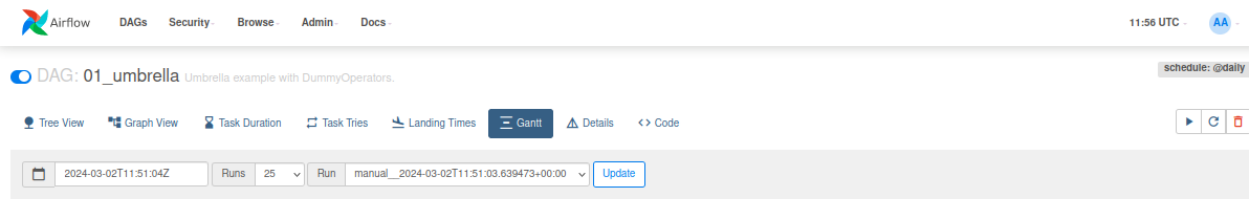


На вкладке "Landing Times" в Apache Airflow DAG показано время “посадки” (landing time) каждой задачи в DAG на отметке времени построения DAG. Это время показывает, когда задача была добавлена в очередь на выполнение. Просматривая информацию на вкладке "Landing Times", можно оценить, какие задачи были добавлены в очередь с определенным временным отступом и какие задачи могут быть более задержаны по сравнению с другими. Это может помочь в оптимизации расписания выполнения задач и общей производительности DAG.



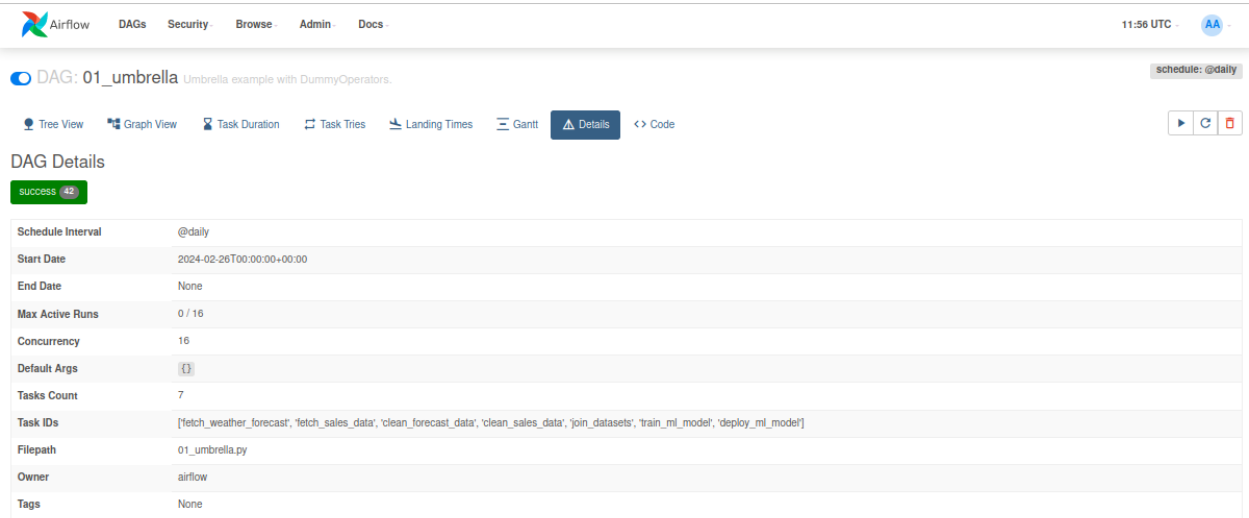
На вкладке "Gantt" в Apache Airflow DAG показана диаграмма Ганта (Gantt chart), которая визуализирует расписание выполнения задач DAG на временной шкале. Каждая задача представлена как полоса на временной оси, показывающая ее запланированное начало и завершение. Это обеспечивает наглядное представление о временном распределении выполнения задач в DAG, и помогает определить зависимости между задачами, оценить

продолжительность выполнения и выявить потенциальные проблемы в расписании. Использование диаграммы Ганта на вкладке "Gantt" может значительно облегчить мониторинг и анализ выполнения DAG.



На вкладке "Details" в Apache Airflow DAG показаны подробности о самом DAG, такие как:

- Информация о DAG: название, описание, состояние выполнения.
- Список всех задач в DAG с их текущим статусом выполнения.
- Время запуска и окончания DAG и каждой отдельной задачи.
- Логи выполнения задач, которые могут помочь в отладке.
- Зависимости между задачами в виде графа, отображающего порядок выполнения задач.



На вкладке "Code" в Apache Airflow DAG отображается исходный код Python, который определяет конфигурацию и структуру DAG. Этот код включает в себя следующие основные элементы:

- Определение DAG с указанием названия, расписания запуска, параметров и описания.
- Определение задач (Task) внутри DAG, каждая задача представляет собой отдельную операцию или шаг в пайплайне.
- Определение зависимостей между задачами, указывая порядок их выполнения.
- Логика выполнения каждой задачи, включая вызов соответствующих операторов (Operator) для выполнения определенных действий.

На вкладке "Code" можно просмотреть и отредактировать исходный код DAG, чтобы добавлять новые задачи, изменять параметры выполнения или вносить другие изменения в пайплайн данных.

← → ↺

localhost:8080/code?dag_id=01_umbrella&root=

80%

🔖

🔔

👤

☰

Airflow

DAGs

Security

Browse

Admin

Docs

11:57 UTC

AA

DAG: 01_umbrella

Umbrella example with DummyOperators.

schedule: @daily

🔍 Tree View

📊 Graph View

🕒 Task Duration

📋 Task Tries

📅 Landing Times

📊 Gantt

⚙️ Details

🔗 Code

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

"""DAG demonstrating the umbrella use case with dummy operators."""

import airflow.utils.dates

from airflow import DAG

from airflow.operators.dummy import DummyOperator

dag = DAG(

dag_id="01_umbrella",

description="Umbrella example with DummyOperators.",

start_date=airflow.utils.dates.days_ago(5),

schedule_interval="@daily",

)

fetch_weather_forecast = DummyOperator(task_id="fetch_weather_forecast", dag=dag)

fetch_sales_data = DummyOperator(task_id="fetch_sales_data", dag=dag)

clean_forecast_data = DummyOperator(task_id="clean_forecast_data", dag=dag)

clean_sales_data = DummyOperator(task_id="clean_sales_data", dag=dag)

join_datasets = DummyOperator(task_id="join_datasets", dag=dag)

train_ml_model = DummyOperator(task_id="train_ml_model", dag=dag)

deploy_ml_model = DummyOperator(task_id="deploy_ml_model", dag=dag)

Set dependencies between all tasks

fetch_weather_forecast >> clean_forecast_data

fetch_sales_data >> clean_sales_data

[clean_forecast_data, clean_sales_data] >> join_datasets

join_datasets >> train_ml_model >> deploy_ml_model

Toggle Wrap

