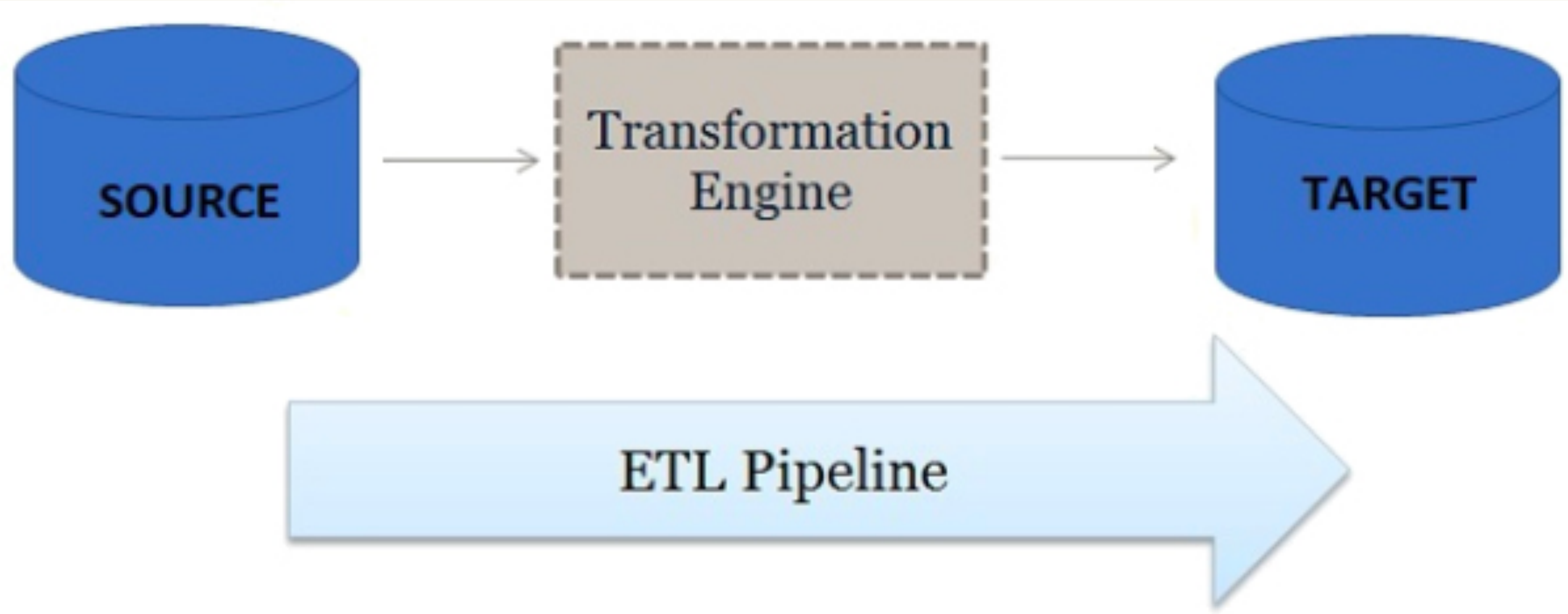
Airflow

Hari Devanathan

What happens if data source is updated frequently?

ETL Pipelines



Components

- Write jobs to extract, transform, or load
- Use a scheduler
- Popular tool: Cron
  - *Built into Linux*

Downside of Cron

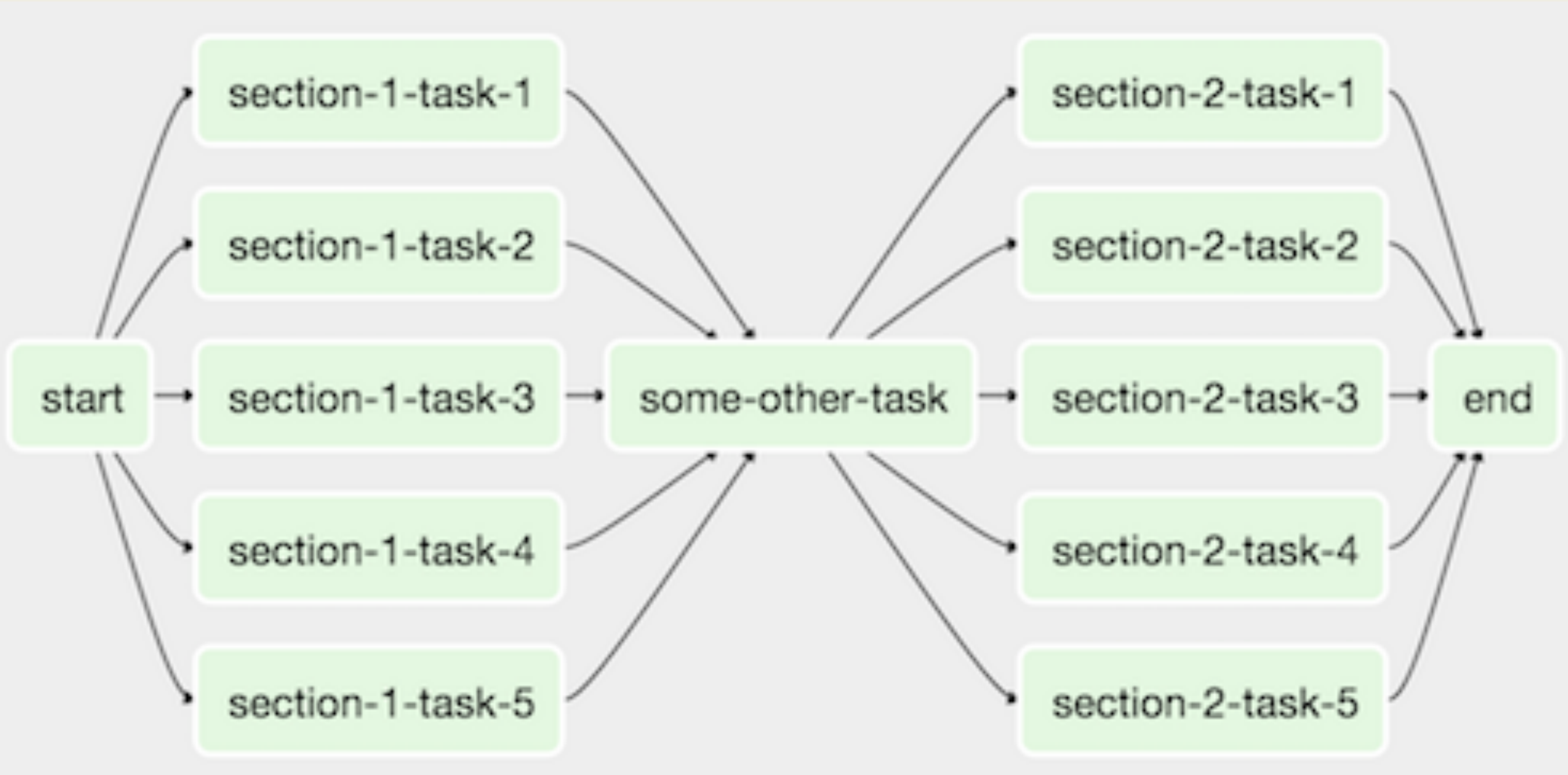
- Managing dependencies between jobs was difficult
- Logs placed where cron job was run
- Rerunning jobs that failed were difficult

Solution: Airflow

- Automate scripts to perform tasks
- Nice UI to monitor and schedule jobs

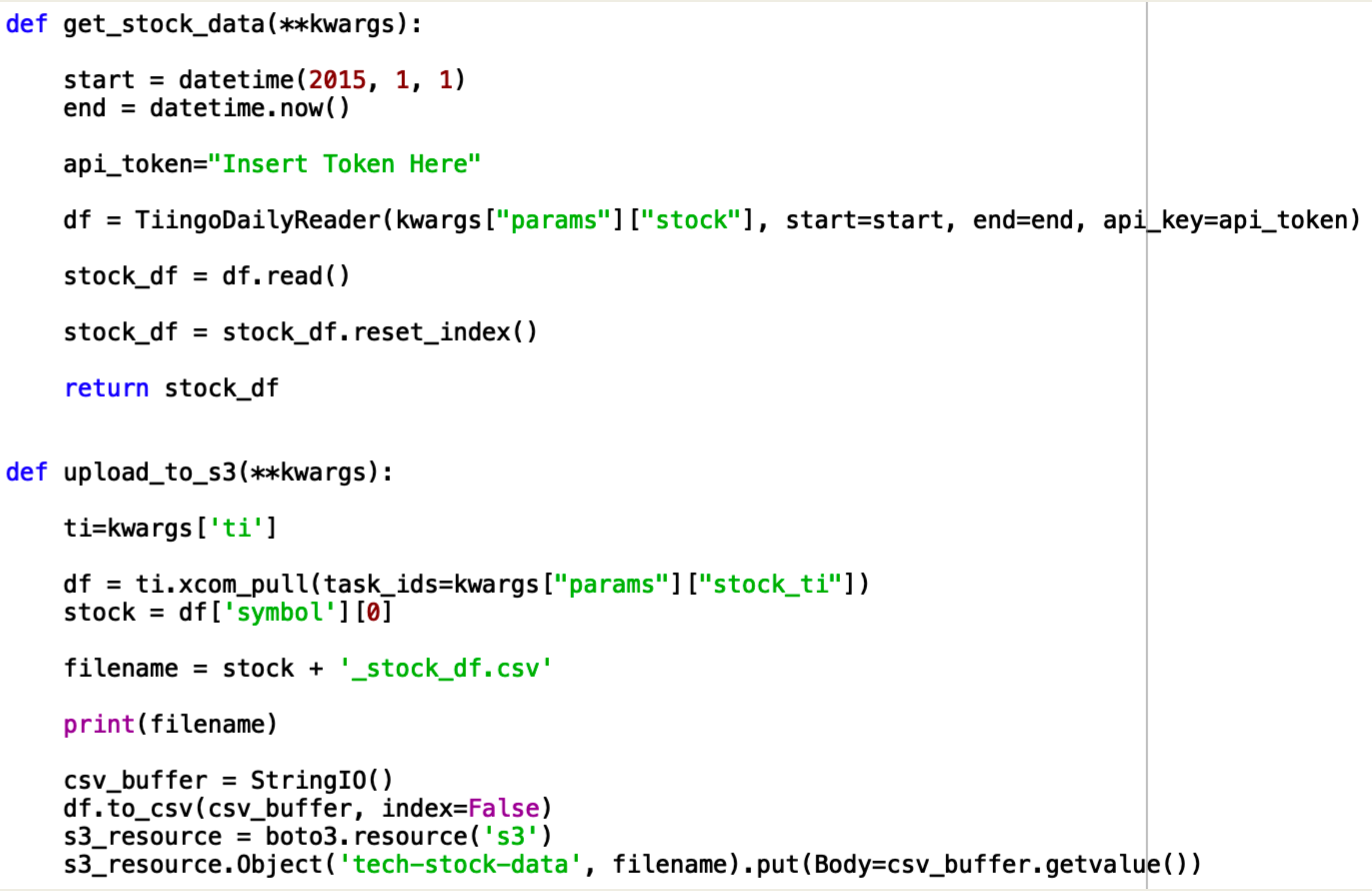
Basic Components

- Workflow/DAG
  - *Acyclic graph where jobs are executed in a sequence*
- Operator
  - *Defines a task that needs to be performed*
  - *PythonOperator, BashOperator, MySQLOperator*
- Task


start
section-1-task-1
section-1-task-2
section-1-task-3
section-1-task-4
section-1-task-5
some-other-task
section-2-task-1
section-2-task-2
section-2-task-3
section-2-task-4
section-2-task-5
end

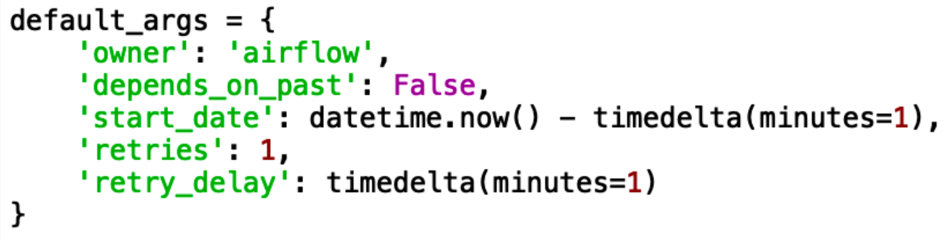
EXAMPLE

TASKS

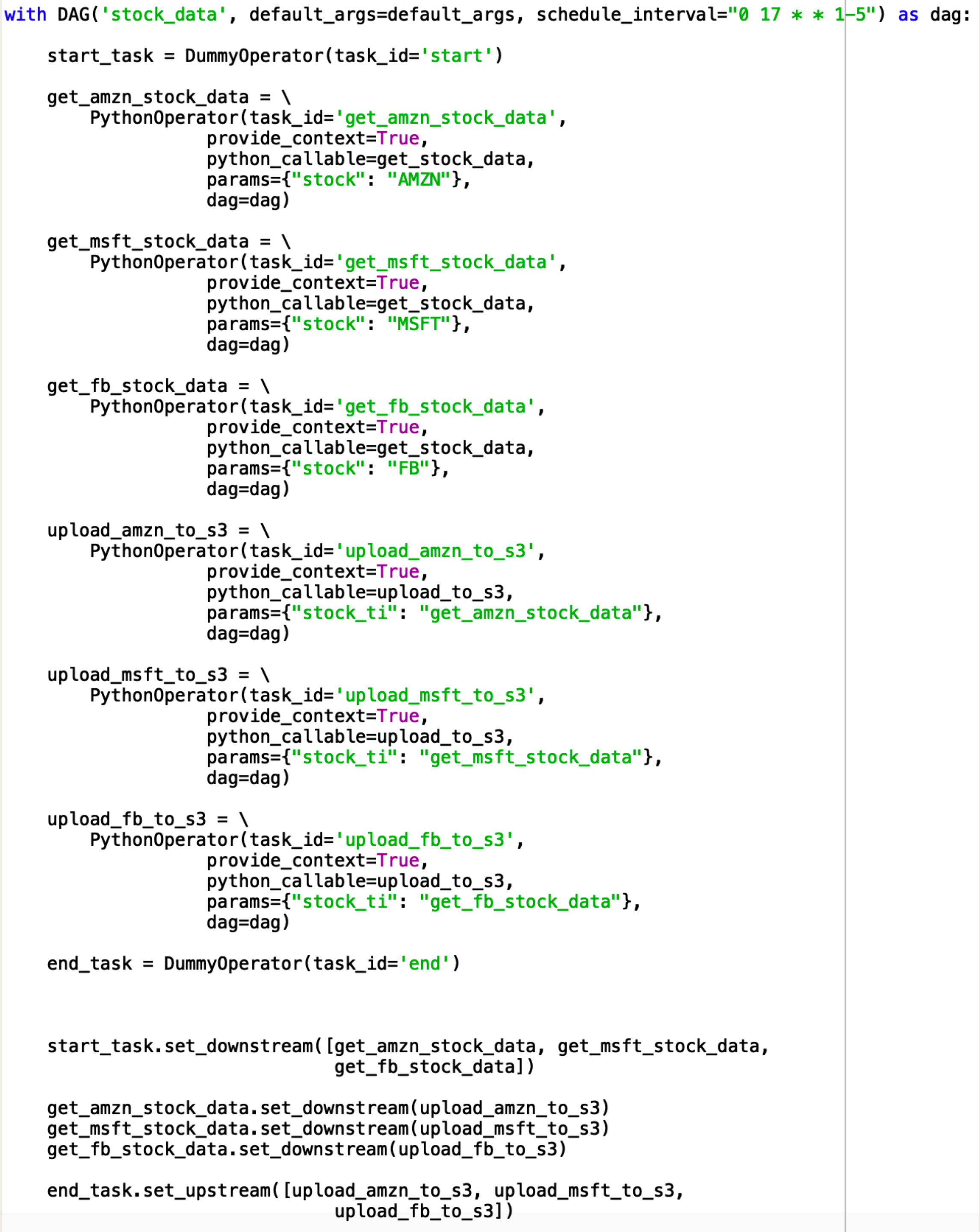


```
def get_stock_data(**kwargs):

    start = datetime(2015, 1, 1)
    end = datetime.now()

    api_token="Insert Token Here"

    df = TiingoDailyReader(kwargs["params"]["stock"], start=start, end=end, api_key=api_token)

    stock_df = df.read()

    stock_df = stock_df.reset_index()

    return stock_df


def upload_to_s3(**kwargs):

    ti=kwargs['ti']

    df = ti.xcom_pull(task_ids=kwargs["params"]["stock_ti"])
    stock = df['symbol'][0]

    filename = stock + '_stock_df.csv'

    print(filename)

    csv_buffer = StringIO()
    df.to_csv(csv_buffer, index=False)
    s3_resource = boto3.resource('s3')
    s3_resource.Object('tech-stock-data', filename).put(Body=csv_buffer.getvalue())
```

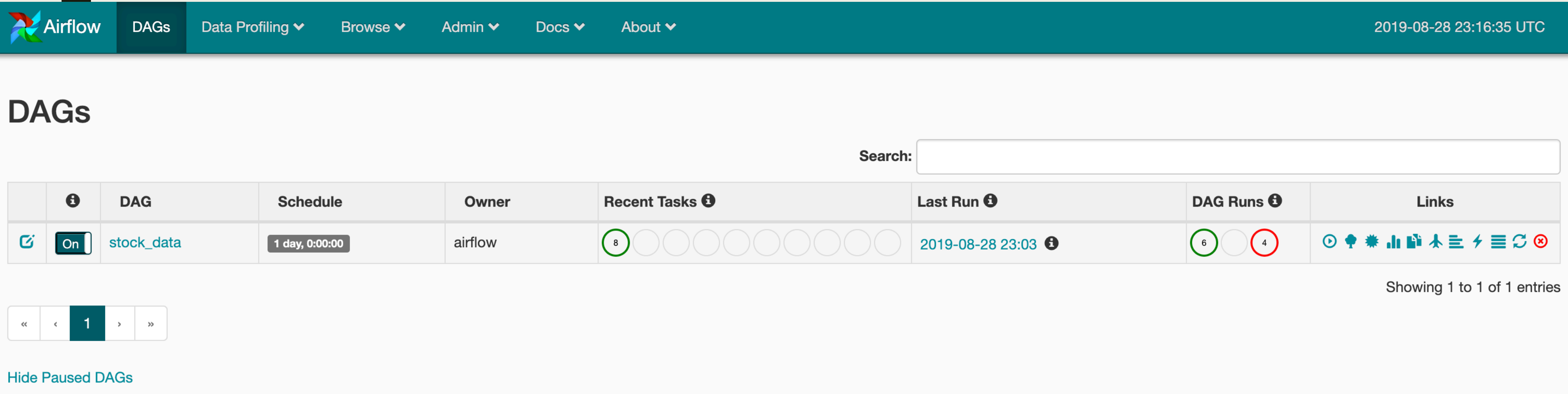
DAG/OPERATORS



```
default_args = {
    'owner': 'airflow',
    'depends_on_past': False,
    'start_date': datetime.now() - timedelta(minutes=1),
    'retries': 1,
    'retry_delay': timedelta(minutes=1)
}
```



```
with DAG('stock_data', default_args=default_args, schedule_interval="0 17 * * 1-5") as dag:

    start_task = DummyOperator(task_id='start')

    get_amzn_stock_data = \
        PythonOperator(task_id='get_amzn_stock_data',
                       provide_context=True,
                       python_callable=get_stock_data,
                       params={"stock": "AMZN"},
                       dag=dag)

    get_msft_stock_data = \
        PythonOperator(task_id='get_msft_stock_data',
                       provide_context=True,
                       python_callable=get_stock_data,
                       params={"stock": "MSFT"},
                       dag=dag)

    get_fb_stock_data = \
        PythonOperator(task_id='get_fb_stock_data',
                       provide_context=True,
                       python_callable=get_stock_data,
                       params={"stock": "FB"},
                       dag=dag)

    upload_amzn_to_s3 = \
        PythonOperator(task_id='upload_amzn_to_s3',
                       provide_context=True,
                       python_callable=upload_to_s3,
                       params={"stock_ti": "get_amzn_stock_data"},
                       dag=dag)

    upload_msft_to_s3 = \
        PythonOperator(task_id='upload_msft_to_s3',
                       provide_context=True,
                       python_callable=upload_to_s3,
                       params={"stock_ti": "get_msft_stock_data"},
                       dag=dag)

    upload_fb_to_s3 = \
        PythonOperator(task_id='upload_fb_to_s3',
                       provide_context=True,
                       python_callable=upload_to_s3,
                       params={"stock_ti": "get_fb_stock_data"},
                       dag=dag)

    end_task = DummyOperator(task_id='end')



    start_task.set_downstream([get_amzn_stock_data, get_msft_stock_data,
                               get_fb_stock_data])

    get_amzn_stock_data.set_downstream(upload_amzn_to_s3)
    get_msft_stock_data.set_downstream(upload_msft_to_s3)
    get_fb_stock_data.set_downstream(upload_fb_to_s3)

    end_task.set_upstream([upload_amzn_to_s3, upload_msft_to_s3,
                           upload_fb_to_s3])
```



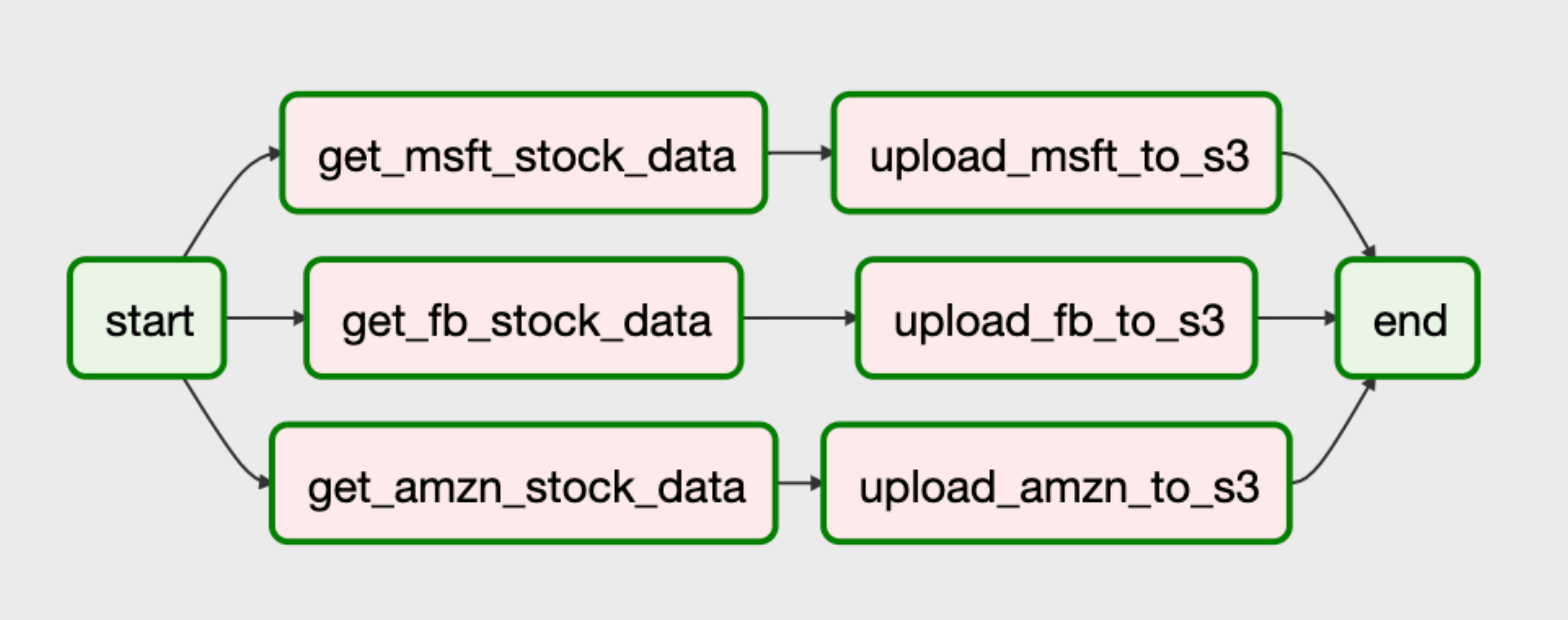
DAGs

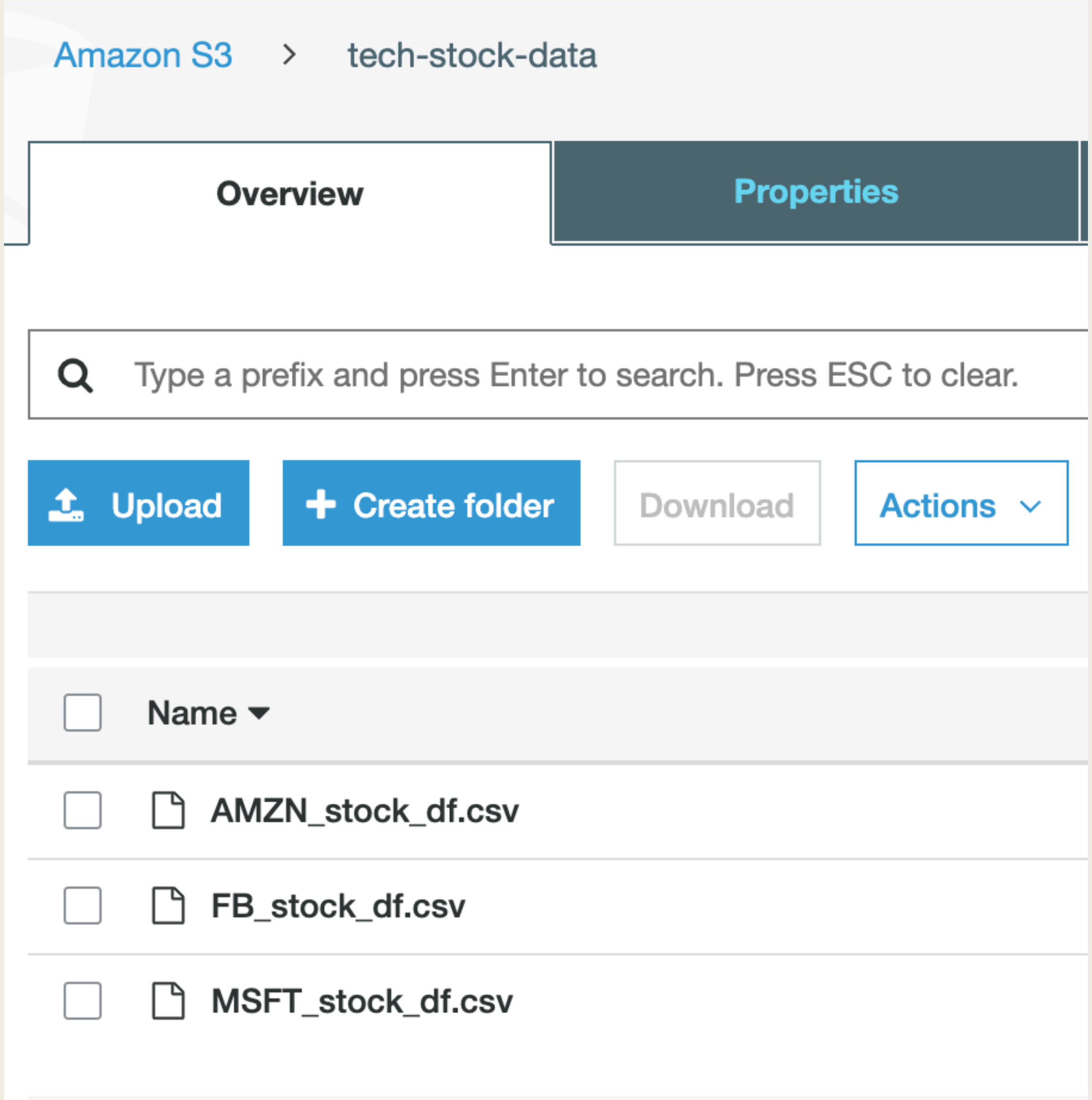
Search:

	ℹ	DAG	Schedule	Owner	Recent Tasks ℹ	Last Run ℹ	DAG Runs ℹ	Links
	On	stock_data	1 day, 0:00:00	airflow	8	2019-08-28 23:03 ℹ	6 4	

Showing 1 to 1 of 1 entries

Hide Paused DAGs


start
get_msft_stock_data
get_fb_stock_data
get_amzn_stock_data
upload_msft_to_s3
upload_fb_to_s3
upload_amzn_to_s3
end


Amazon S3 > tech-stock-data
Overview
Properties
Type a prefix and press Enter to search. Press ESC to clear.
Upload
Create folder
Download
Actions
Name

□ Name

□ AMZN_stock_df.csv

□ FB_stock_df.csv

□ MSFT_stock_df.csv