

## Practice: Backfill your DAG without the CLI

### Introduction

- If you don't have access to the CLI, there is no direct backfill method.
- However it is possible by triggering a `backfill_trigger_dag` via the API and using `dag_run.conf` to pass in parameters such as `start_date`, `end_date`, and `dag_id` for the required backfill.
- the `backfill_trigger_dag` will then trigger the backfill for the desired *target* DAG
- ensure the `backfill_trigger_dag` is unpaused, with a schedule of `None` so that it can be triggered as needed but does not run on an interval

### Prerequisites

- Airflow CLI installed


### Instructions

→ In the below example, we will use two DAGs:

1. The `backfill_trigger_dag` - this DAG uses a `BashOperator` to run the Airflow CLI command `airflow dags backfill` and pulls in the `dag_run.conf` values that we pass while manually triggering on the UI i.e `start_date`, `end_date`, `dag_id`, etc.
2. The `example_dag_basic` - this is the DAG the backfill is being performed on (i.e. the *target* DAG). You can backfill any DAG of your choice.

→ The DAG responsible for triggering the backfill:

`backfill_trigger_dag`:

 **Note:** We will use Jinja templates to pull in the `dag_run.conf` values from the API call:

```

from airflow import DAG
from airflow.operators.bash import BashOperator
from datetime import datetime

with DAG(dag_id='backfill_trigger_dag',
        schedule_interval=None,
        start_date=datetime(2022, 1, 1),
        tags=['backfill-trigger-cli'],
        catchup=False) as dag:

    # Use the UI to trigger a DAG run with conf to trigger a backfill, passing in start/end
    trigger_backfill = BashOperator(
        task_id='trigger_backfill',
        bash_command="airflow dags backfill --reset-dagruns -y -s {{ dag_run.conf['date_start'] }} -e {{ dag_run.conf['date_end'] }}"
    )

    trigger_backfill

```

→ The execution will be as follows

1. First we shall manually trigger the **backfill\_trigger\_dag** with a configuration json.

The screenshot shows the Airflow DAGs interface. At the top, there are tabs for 'All' (19), 'Active' (2), and 'Paused' (17). Below the tabs is a search bar and an 'Auto-refresh' button. The main table lists DAGs with columns: DAG, Owner, Runs, Schedule, Last Run, Next Run, Recent Tasks, Actions, and Links. The first DAG is 'backfill\_trigger\_dag' with owner 'airflow' and no schedule. The 'Actions' column for this DAG has a dropdown menu with two options: 'Trigger DAG' and 'Trigger DAG w/ config'. Red arrows are numbered 1 through 5, indicating the steps to trigger the DAG with configuration.

2. Now we shall pass the necessary parameters for performing the backfill including

→ **dag\_id**

→ **date\_start**

→ **date\_end**

i.e {"dag\_id": "example\_dag\_basic", "date\_start": 20230401, "date\_end": 20230405}

# Trigger DAG: backfill\_trigger\_dag

Logical date



2023-04-06T07:28:01+0

Run id (Optional)

Run ID

Configuration JSON (Optional, must be a dict object)

```
1 {"dag_id": "example_dag_basic", "date_start": 20230401, "date_end": 20230405}
```

1

To access configuration in your DAG use `{{ dag_run.conf }}`. As `core.dag_run_conf_overrides_params` is set to `True`,

☒ Unpause DAG when triggered

Trigger

Cancel

3. And we get the 5 backfills successfully

## DAGs

All 19

Active 2

Paused 17

Filter DAGs by tag

Search

DAG	Owner	Runs	Schedule	Last Run	Next Run
<input checked="" type="checkbox"/> backfill_trigger_dag backfill-trigger-cli	airflow	1	None	2023-04-06, 08:41:23	
<input type="checkbox"/> circular2	airflow		1 day, 0:00:00		2023-04-05, 08:42:12
<input type="checkbox"/> example_dag_advanced example	community		@daily		2023-04-05, 00:00:00
<input type="checkbox"/> example_dag_basic example	airflow	5	@daily	2023-04-05, 00:00:00	2023-04-05, 00:00:00

4. We can verify the same from the Browse → DAG Runs tab as well

List Dag Run

Search

Actions

Record Count: 6

	State	Dag Id	Logical Date	Run Id	Run Type	Queued At	Start Date	End Date	Note	External Trigger	Conf
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> success	backfill_trigger_dag	2023-04-06, 08:41:23	manual__2023-04-06T08:41:23+00:00	manual	2023-04-06, 08:42:21	2023-04-06, 08:42:21	2023-04-06, 08:42:55		True	{"dag_id": "example_dag_basic", "date_s
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> success	example_dag_basic	2023-04-05, 00:00:00	backfill__2023-04-05T00:00:00+00:00	backfill		2023-04-06, 08:42:23	2023-04-06, 08:42:53		False	{}
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> success	example_dag_basic	2023-04-04, 00:00:00	backfill__2023-04-04T00:00:00+00:00	backfill		2023-04-06, 08:42:23	2023-04-06, 08:42:53		False	{}
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> success	example_dag_basic	2023-04-03, 00:00:00	backfill__2023-04-03T00:00:00+00:00	backfill		2023-04-06, 08:42:23	2023-04-06, 08:42:53		False	{}
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> success	example_dag_basic	2023-04-02, 00:00:00	backfill__2023-04-02T00:00:00+00:00	backfill		2023-04-06, 08:42:23	2023-04-06, 08:42:53		False	{}
<input type="checkbox"/>	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> success	example_dag_basic	2023-04-01, 00:00:00	backfill__2023-04-01T00:00:00+00:00	backfill		2023-04-06, 08:42:23	2023-04-06, 08:42:53		False	{}

Great! Now we can backfill a DAG even if we don’t have access to the Airflow CLI.

Was this page helpful?



Want feedback like this? [Try Hotjar](#)