Time zones

Support for time zones is enabled by default. Airflow stores datetime information in UTC internally and in the database. It allows you to run your DAGs with time zone dependent schedules. At the moment Airflow does not convert them to the end user's time zone in the user interface. There it will always be displayed in UTC. Also templates used in Operators are not converted. Time zone information is exposed and it is up to the writer of DAG what do with it.

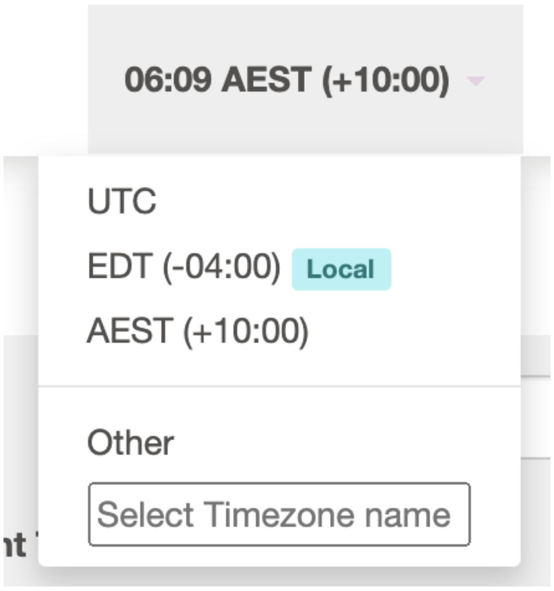
This is handy if your users live in more than one time zone and you want to display datetime information according to each user's wall clock.

Even if you are running Airflow in only one time zone it is still good practice to store data in UTC in your database (also before Airflow became time zone aware this was also to recommended or even required setup). The main reason is Daylight Saving Time (DST). Many countries have a system of DST, where clocks are moved forward in spring and backward in autumn. If you're working in local time, you're likely to encounter errors twice a year, when the transitions happen. (The pendulum and pytz documentation discusses these issues in greater detail.) This probably doesn't matter for a simple DAG, but it's a problem if you are in, for example, financial services where you have end of day deadlines to meet.

The time zone is set in `airflow.cfg`. By default it is set to utc, but you change it to use the system's settings or an arbitrary IANA time zone, e.g. `Europe/Amsterdam`. It is dependent on `pendulum`, which is more accurate than `pytz`. Pendulum is installed when you install Airflow.

Web UI

By default the Web UI will show times in UTC. It is possible to change the timezone shown by using the menu in the top right (click on the clock to activate it):



"Local" is detected from the browser's timezone. The "Server" value comes from the `default_timezone` setting in the `[core]` section.

The users' selected timezone is stored in LocalStorage so is a pre-browser setting.

Note

If you have configured your Airflow install to use a different default timezone and want the UI to use this same timezone, set `default_ui_timezone` in the `[webserver]` section to either an empty string, or the same value.

(It currently defaults to UTC to keep behaviour of the UI consistent by default between point-releases.)

Concepts

Naive and aware datetime objects

Python's datetime.datetime objects have a tzinfo attribute that can be used to store time zone information, represented as an instance of a subclass of datetime.tzinfo. When this attribute is set and describes an offset, a datetime object is aware. Otherwise, it's naive.

You can use `timezone.is_localized()` and `timezone.is_naive()` to determine whether datetimes are aware or naive.

Because Airflow uses time-zone-aware datetime objects. If your code creates datetime objects they need to be aware too.

```
from airflow.utils import timezone

now = timezone.utcnow()
a_date = timezone.datetime(2017,1,1)
```

Interpretation of naive datetime objects

Although Airflow operates fully time zone aware, it still accepts naive date time objects for `start_dates` and `end_dates` in your DAG definitions. This is mostly in order to preserve backwards compatibility. In case a naive `start_date` or `end_date` is encountered the default time zone is applied. It is applied in such a way that it is assumed that the naive date time is already in the default time zone. In other words if you have a default time zone setting of `Europe/Amsterdam` and create a naive datetime `start_date` of `datetime(2017,1,1)` it is assumed to be a `start_date` of Jan 1, 2017 Amsterdam time.

```
default_args=dict(
    start_date=datetime(2016, 1, 1),
    owner='airflow'
)

dag = DAG('my_dag', default_args=default_args)
op = DummyOperator(task_id='dummy', dag=dag)
print(op.owner) # Airflow
```

Unfortunately, during DST transitions, some datetimes don't exist or are ambiguous. In such situations, pendulum raises an exception. That's why you should always create aware datetime objects when time zone support is enabled.

In practice, this is rarely an issue. Airflow gives you aware datetime objects in the models and DAGs, and most often, new datetime objects are created from existing ones through timedelta arithmetic. The only datetime that's often created in application code is the current time, and `timezone.utcnow()` automatically does the right thing.

Default time zone

The default time zone is the time zone defined by the `default_timezone` setting under `[core]` . If you just installed Airflow it will be set to `utc` , which is recommended. You can also set it to `system` or an IANA time zone (e.g. `Europe/Amsterdam`). DAGs are also evaluated on Airflow workers, it is therefore important to make sure this setting is equal on all Airflow nodes.

```
[core]
default_timezone = utc
```

Note

For more information on setting the configuration, see Setting Configuration Options

Time zone aware DAGs

Creating a time zone aware DAG is quite simple. Just make sure to supply a time zone aware `start_date` using `pendulum` .

```
import pendulum

local_tz = pendulum.timezone("Europe/Amsterdam")

default_args=dict(
    start_date=datetime(2016, 1, 1, tzinfo=local_tz),
    owner='airflow'
)

dag = DAG('my_tz_dag', default_args=default_args)
op = DummyOperator(task_id='dummy', dag=dag)
print(dag.timezone) # <Timezone [Europe/Amsterdam]>
```

Please note that while it is possible to set a `start_date` and `end_date` for Tasks always the DAG timezone or global timezone (in that order) will be used to calculate the next execution date. Upon first encounter the start date or end date will be converted to UTC using the timezone associated with start_date or end_date, then for calculations this timezone information will be disregarded.

Templates

Airflow returns time zone aware datetimes in templates, but does not convert them to local time so they remain in UTC. It is left up to the DAG to handle this.

```
import pendulum

local_tz = pendulum.timezone("Europe/Amsterdam")
local_tz.convert(execution_date)
```

Cron schedules

Time zone aware DAGs that use cron schedules respect daylight savings time. For example, a DAG with a start date in the `US/Eastern` time zone with a schedule of `0 0 * * *` will run daily at 04:00 UTC during daylight savings time and at 05:00 otherwise.

Time deltas

Time zone aware DAGs that use `timedelta` or `relativedelta` schedules respect daylight savings time for the start date but do not adjust for daylight savings time when scheduling subsequent runs. For example, a DAG with a start date of `pendulum.create(2020, 1, 1, tz="US/Eastern")` and a schedule interval of `timedelta(days=1)` will run daily at 05:00 UTC regardless of daylight savings time.

Previous Next

Was this entry helpful?

☆ ☆ ☆ ☆ ☆

Want to be a part of Apache Airflow? Join community