How to Share Data Between Your Tasks With XCOM?

Let’s your tasks communicate

Use Case

- Let's say we fetched data from a table using PostgresHook and PythonOperator but now we would like to pass one record (composed of a name and a value) of those data to another task. How could we do that?
- One solution could be to save those data into a network-attached storage (NAS). Yes, you can do that but since the data we want to share is very small, it's a bit overkill to do so.
- Another solution could be to use a database where we would simply store the output into a temporary table and fetch the results back to the destination task. Again, it's a valuable solution if you have a large set of data.
- Actually, Apache Airflow introduced XCOMs to share key-value information between tasks.

Definition

XCOM stands for “cross-communication” and allows multiple tasks to exchange messages (data) between them.

XCOMs are principally defined by a key, value and a timestamp.

They are stored into the Airflow’s metadatabase with an associated `execution_date`, `task_id` and `dag_id`.

How Do They Work?

- XCOMs (data) can be “pushed” (sent) or “pulled” (received).
- When we push a message from a task using `xcom_push()`, this message becomes available to other tasks.
  - If a task returns a value (either from its Operator’s execute() method, or from a PythonOperator’s python_callable function), a XCOM containing that value is automatically pushed.
- When we pull a message from a task using `xcom_pull()`, the task gets the message based on parameters such as `key`, `task_ids` (the “s” is not a mistyping) and `dag_id`.
  - By default, xcom_pull() for the keys that are automatically given to XCOMs when they are pushed by being returned from execute functions (as opposed to XCOMs that are pushed manually).

Important Notes

- XCOMs can be used to share any object that can be serializable (pickled) but be careful about the size of this object. Airflow is not a data streaming solution!
- Some operators such as BashOperator or SimpleHttpOperator have a parameter called `xcom_push=False` by default. If you set `xcom_push=True` the last output will be pushed to an XCOM for the BashOperator or if you use SimpleHttpOperator, the response of the HTTP request will also be pushed to an XCOM.
- Be careful, `execution_date` has not the same meaning in the context of a DagRun and a XCOM. `execution_date` in XCOMs is used to hide a XCOM until this date. For example, if we have two XCOMs with the same key value, dag id and task id, the XCOM having the most recent `execution_date` will be pulled out by default. If you didn't set an `execution_date`, this date will be equal to the `execution_date` of the DagRun.

Coding Time!

Let’s explore this!