Paralelismo

Anteriormente, ao trabalharmos com o executor Local, conhecemos o parâmetro "max_active_tasks_per_dag", responsável por definir a quantidade máxima de tarefas que podem ser executadas paralelamente em um mesmo DAG. Lembre-se que o definimos como "2".

Agora que estamos trabalhando com o executor Celery e utilizando workers, existe outro parâmetro importante, o "worker_concurrency", responsável por definir a quantidade máxima de tarefas que podem ser executadas simultaneamente por um mesmo worker.

Precisamos entender a forma como "worker_concurrency" se relaciona com os demais parâmetros. Por exemplo, se definirmos o "max_active_tasks_per_dag" como "2" e o "work_concurrency" como "3", mas tivermos somente um DAG ativo, o nosso worker só poderá executar 2 tarefas em paralelo, pois definimos que um DAG só realiza duas tasks simultaneamente, embora o nosso worker pudesse executar até 3 tarefas.

Para que possamos melhor visualizar esse exemplo, vamos redefinir esses parâmetros.

Abra o arquivo de configuração "airflow.cfg", da pasta "airflow", no VS Code. Agora, tecle "Ctrl + F" para abrir a barra de pesquisa, busque por "worker_concurrency" e vá até a variável de mesmo nome, que deve estar definida como "16"; altere-a para "3".

Para usar o arquivo "airflow.cfg" no **docker** tire o comentário da tag AIRFLOW_CONFIG no arquivo docker-compose.yaml e informe o caminho do arquivo.

Alteramos o valor do parâmetro worker_concurrency para 3, enquanto deixamos o max_active_tasks_per_dag como 2. Com isso, nós percebemos que mesmo com o nosso worker conseguindo executar até 3 tarefas ao mesmo tempo, apenas duas estavam sendo executadas, pois tínhamos apenas um DAG ativo que poderia enviar pra fila apenas duas tarefas ao mesmo tempo.

E se deixarmos os nossos parâmetros da seguinte maneira:

```
max_active_tasks_per_dag  = 2
worker_concurrency  = 1
```

O nosso worker vai conseguir executar apenas uma tarefa. Mas nosso DAG vai sempre agendar duas. Nesse caso, será que a segunda tarefa agendada pelo DAG fica na fila de tarefas aguardando o worker executar a primeira?

Vamos testar isso!

- Altere o parâmetro worker_concurrency = 1 no arquivo airflow.cfg;
- Execute o Airflow e um Worker;
- Execute o Flower e veja como as tarefas serão executadas

Pools

Os parâmetros "max_active_runs_per_dag" e "max_active_tasks_per_dag" funcionam no escopo do DAG, limitando, respectivamente, a quantidade de DAG runs executados simultaneamente e a quantidade de tarefas executadas em paralelo. Aprendemos, também, que o "worker_concurrency" define o número de tarefas simultâneas que cada worker pode executar.

Além desses ajustes, há, ainda, outro parâmetro muito importante: os pools, algo como "reservatórios", em tradução livre. Vamos a um exemplo para que possamos melhor compreender do que se trata essa configuração!

No nosso projeto, estamos extraindo dados da API do Yahoo Finance utilizando a biblioteca YFinance. Temos dois DAGs, com diferentes tarefas, responsáveis por extrair simultaneamente esses dados. No entanto, precisamos nos ater ao fato que esta API, assim como as demais, possui um limite de requisições que suporta receber em paralelo. Portanto, se ultrapassarmos este limite durante a execução das tasks, obteremos falhas.

Cada um dos nossos DAGs possui uma quantidade de tarefas, mas nosso worker consegue executar, no máximo, 3 delas simultaneamente, o que indica que a API receberá sempre 3 requisições em paralelo. Suponhamos, no entanto, que a API suportasse somente 2 requisições ao mesmo tempo, isso geraria uma falha porque extrapolamos o número de requisições que ela suporta. É aí que entra a configuração dos pools!

Os pools podem ser utilizados para limitar o paralelismo de execução em conjuntos específicos de tarefas. Basicamente, estes reservatórios informam quantas tarefas podem ser executadas simultaneamente considerando todos os DAGs ativos.

Acesse a interface do Airflow no navegador. Perceba que há quatro opções na parte superior: "DAGs", "Security", "Browse", "Admin" e "Docs". Em "Admin", selecione a 6º subopção, "Pools". Ela o levará ao reservatório padrão "default_pool" e à quantidade de slots que ele disponibiliza para a execução de tarefas, que está definida como "128".

Cada tarefa utiliza 1 slot enquanto está em execução, então a quantidade de slots disponibilizadas pelo pool condiz com o número de tarefas que podem ser executadas simultaneamente, considerando todos os DAGs ativos que utilizam o mesmo reservatório. Sendo assim, quando não temos nenhum outro pool, todos os DAGs criados utilizam o reservatório "default_pool" por padrão.

Vamos criar nosso próprio reservatório! Para isso, clique no símbolo de "+", chamado "Add a new record", no canto superior esquerdo, e o nomeie de "small_pool". Defina o valor de slots como "2" e clique em "Save". Nosso pool foi criado!

Como o Apache Airflow é uma ferramenta escalável, pode acontecer de chegarmos em um ponto no qual múltiplas tarefas estão sendo executadas em paralelo.

No entanto, em casos como esse é necessário ter um pouco de cautela. Isso porque, se você tiver muitas tarefas que interagem com o mesmo sistema de origem, como um banco de dados ou API, você certamente não deseja sobrecarregá-los com requisições. É justamente em situações como essas que podemos utilizar os pools

Os pools ou reservatórios, é uma configuração que pode ser usada para limitar o paralelismo de execução em conjuntos específicos de tarefas. Basicamente, esses reservatórios servem para informar quantas tarefas podem ser executadas em paralelo considerando todos os DAGs que estão ativos ao mesmo tempo.

Eles são frequentemente usados nos casos em que você deseja limitar o número de tarefas paralelas que fazem uma determinada coisa.

Por padrão, todas as tarefas do Airflow são atribuídas ao default_pool que é o reservatório padrão. Ele possui 128 slots disponíveis, mas você pode alterar esse valor como também pode criar seu próprio reservatório. Quando tarefas são atribuídas a um pool específico, elas são agendadas até os slots daquele pool serem todos preenchidos.

À medida que os slots ficam disponíveis novamente, as tarefas que restaram na fila de execução começam a ser executadas.

Para mais informações sobre essa configuração consulte a documentação:

- Pools

Ajuste de Parâmetros

O Airflow é um orquestrador utilizado para uma ampla variedade de casos de uso e por isso possui tantos parâmetros que podem ser ajustados. Com esses inúmeros parâmetros, o Airflow permite que a gente realize:

- Configurações do ambiente;
- Configurações do DAG;
- Configurações da tarefa.

Por isso é importante que o administrador do Airflow conheça bem os requisitos do seu caso de uso antes de passar para o dimensionamento do Airflow e conseguir escolher quais parâmetros precisam de modificação.

Aqui no curso nós conhecemos os seguintes parâmetros:

- **max_active_runs_per_dag**: determina o número máximo de execuções de DAG (DAG runs) ativas por DAG que o Scheduler pode criar simultaneamente;
- **max_active_tasks_per_dag**: determina o número máximo de tarefas que podem ser agendadas/executadas de forma simultânea em um mesmo DAG;
- **pool**: determina a quantidade de slots disponíveis para a execução de um conjunto de tarefas;
- **worker_concurrency**: determina quantas tarefas cada worker do Celery pode executar simultaneamente.

Além desses, existem outros parâmetros que alteram diferentes configurações, caso queira se aprofundar um pouco mais e conhecer eles, sugiro a leitura do artigo:

- Scaling Out Airflow