Apache Airflow

Uvod

- Apache Airflow je open-source platforma za razvoj, zakazivanje i praćenje batch orjentisanih workflow- a
- Workflow može biti napravljen sa skoro bilo kojom tehnologijom što je moguće zbog Airflow- ovog proširivog python radnog okvira
- Stanjima workflow- a se upravlja pomoću web interfejsa
- Jedna od glavnih karakteristika Airflow workflow- a je da se svi pišu u python- u

Uvod

- Primer koda

```
from datetime import datetime
from airflow import DAG
from airflow.decorators import task
from airflow.operators.bash import BashOperator

# A DAG represents a workflow, a collection of tasks
with DAG(dag_id="demo", start_date=datetime(2022, 1, 1), schedule="0 0 * * *") as dag:

    # Tasks are represented as operators
    hello = BashOperator( task_id="hello", bash_command="echo hello")

    @task()
    def airflow():
        print("airflow")

    # Set dependencies between tasks
    hello >> airflow()
```

Apache Airflow – Instalacija

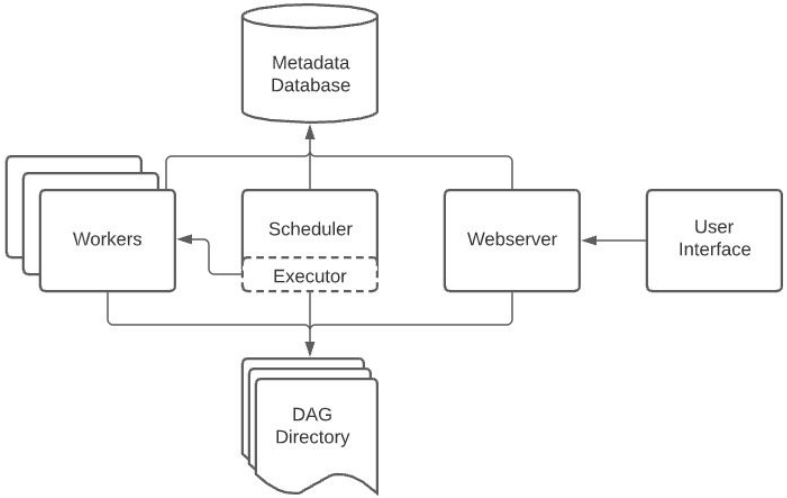
- Moguće je instalirati Airflow na više načina
  - upotrebom izvornog koda i kreiranih paketa koji se mogu pokretati/izvršavati direktno na lokalnoj mašini
  - upotrebom PyPi
  - upotrebom docker slika
  - upotrebom oficijalnog Airflow Helm Chart
  - upotrebom 3rd-party slika…

Apache Airflow – Instalacija

- Preduslovi
  - Python: 3.8, 3.9, 3.10, 3.11, 3.12, 3.13
  - Kompatibilne baze podataka
    - PostgreSQL: 11, 12, 13, 14, 15, 16, 17
    - MySQL: 5.7, 8.0, 8.1
    - SQLite: 3.15.0+ - parallelization not possible
    - MSSQL(Experimental): 2017, 2019 - deprecated
  - Kubernetes: 1.23, 1.24, 1.25, 1.26, 1.27, 1.30, 1.31, 1.32, 1.33
    - If used with kubernetes

Apache Airflow – Pregled Arhitekture

- Workflow je predstavljen kao DAG (Directed Acyclic Graph) i sadrži individualne delove workflow- a koji se nazivaju zadacima (Task- ovima)
- DAG specificira zavisnosti među zadacima i uređenje u kom treba da budu izvršeni
- Airflow se sastoji od sledećih komponenti
  - scheduler
  - executor
  - webserver
  - dag files
  - metadata database



Apache Airflow – Pregled Arhitekture

- Workload
  - DAG se pokreće kroz seriju zadataka (Task- ova)
- Kontrolni tok
  - DAG- ovi su dizajnirani da se izvršavaju više puta i više njih može biti pokrenuto u paraleli
  - DAG- ovi su parametrizovani
- Korisnički interfejs
  - Airflow dolazi sa korisničkih interfejsom putem kog može da se prati stanje DAG- ova i zadataka, okidanje DAG- ova, pregled logova i da se uradi ograničeno debug- ovanje DAG- a

Apache Airflow – DAGs

- DAG (Directed Acyclic Graph) je osnovni koncept Airflow- a, koji predstavlja kolekciju zadataka, organizovan sa zavisnostima i vezama kojima se iskazuje kako zadaci trebaju biti pokrenuti
- Deklaracija DAG- a
  - context manager
  - standard constructor
  - dag decorator
- DAG argumenti
  - https://airflow.apache.org/docs/apache-airflow/2.7.2/_api/airflow/models/dag/index.html#airflow.models.dag.DAG
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/dags.html#declaring-a-dag

Apache Airflow – Tasks

- Zadatak (Task) je osnova jedinica izvršavanja u Airflow- u
- Zadaci su raspoređeni u DAG- ove i imaju postavljene zavisnosti između njih kako bi se zadao redosled kojim bi trebalo da budu izvršavani
- Postoje tri osnovna tipa zadataka
  - Operatori
  - Senzori
  - TaskFlow- dekorisani zadaci
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/tasks.html

Apache Airflow – Operators

- Operator je konceptualno šablon za predefinisani zadatak (Task)
- Operatori mogu biti definisani deklarativno unutar DAG- a
- Airflow ima veoma proširiv skup operatora, ali najpopularniji su
  - BashOperator
  - PythonOperator
  - EmailOperator
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/operators.html

Apache Airflow – Sensors

- Senzori su poseban tip operatora koji su dizanirani da čekaju na događaj
- Mogu biti vremenski orijentisani, mogu čekati na pojavu datoteke ili na neki eksterni događaj
- Senzori su primarno u idle stanju i imaju dva različita režima rada
  - poke (default)
  - reschedule
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/sensors.html

Apache Airflow – TaskFlow

- Ako je kod najčešće pisan upotrebom python- a umesto operatora, upotrebom TaskFlow API- a će biti olakšano održavanje "čistih" DAG- ova bez suvišnog ponovljenog koda uz pomoć @task dekoratora
- TaskFlow vodi računa o ulazima i izlazima između zadataka upotrebom XComs
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/taskflow.html

Apache Airflow – Executor

- Izvršivači (executors) su mehanizam putem kog se instanca zadatka pokreće
- Imaju unificiran API i mogu biti "uključivani" odnosno zamenjeni (swapped) spram potrebe
- Da bi se proverio koji izvršivač je trenutno postavljen može se izvršiti
  - airflow config get-value core executor
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/executor/index.html

Apache Airflow – XComs

- XComs (skraćeno od "cross-communications") su mehanizam koji dozvoljava zadacima da komuniciraju međusobno
- Podrazumevano zadaci su u potpunosti izolovani i mogu biti izvršavani na različitim mašinama
- XCom je identifikovan putem:
  - key - njegov naziv
  - task_id
  - dag_id
- XComs mogu imati bilo koju serijalizabilnu vrednost, ali su dizajnirani samo za male količine podataka
- Ne bi trebali biti korišćeni da se prenesu velike vrednosti puput dataframe-ova

Apache Airflow – XComs

- XComs su eksplicitno "pushed" and "pulled" prema/iz njihovog skladišta koristeći xcom_push i xcom_pull metode nad instancama zadataka
- XComs mogu biti korišćeni u šablonima
  - SELECT * FROM {{ task_instance.xcom_pull(task_ids='foo', key='table_name') }}
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/xcoms.html

Apache Airflow – Variables

- Varijable su Airflow runtime konfiguracioni koncept, ključ/vrednost skladište koje je globalno i može biti dobavljano iz zadataka
- Mogu biti jednostavno postavljane sa Airflow korisničkim interfejsom ili bulk-postavljane kao JSON fajl
- Mogu biti upotrebljavane u
  - kodu - pomoću modela Variable upotrebom get metode nad njim
  - šablonima
- Varijable su globalne i one ne bi trebale biti korišćene da bi se prosledili podaci između zadataka nego za konfiguraciju

Apache Airflow – Variables

- Prema Airflow dokumentaciji preporučeno je da se podešavanja i konfiguracija čuvaju u DAG fajlovima da bi mogli biti verzionisani
- **Varijable se koriste samo za vrednosti koje su zaista runtime zavisne!**
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/variables.html

Apache Airflow – Params

- Parametri su korišćeni da bi se obezbedila runtime konfiguracija zadataka
- Podrazumevani parametri mogu biti konfigurisani u DAG kodu, a i dodatni parametri mogu biti dodati kroz kod
- Moguće je da se prepišu vrednosti Parametara tokom runtime- a kada je DAG okinut
- Parametri se validiraju koristeći JSON šemu koja treba biti unapred zadata
- Za zakazano pokretanje DAG- a koriste se podrazumevane vrednosti Param-a
- https://airflow.apache.org/docs/apache-airflow/2.7.2/core-concepts/params.html

Okruženje

- Putem dokera se simulira realni scenario
- U okviru Dockerfile se nalazi instalacija neophodnih dodatnih paketa
- U docker compose fajlu se nalazi specifikacija svih servisa koji su neophodni
  - preuzeta je sa oficijalnog sajta i delimično adaptirana
- potrebno je preuzeti apache spark binaries (pod nazivom spark-3.0.1-bin-hadoop3.2.tgz) i postaviti ga direktorijum Airflow
  - https://archive.apache.org/dist/spark/spark-3.0.1/spark-3.0.1-bin-hadoop3.2.tgz

Zadatak

- Napisati spark program putem python- a koji prikazuje frekvenciju pojavljivanja reči u Danteovoj božanskoj komediji. Koristiti apache airflow za pokretanje spark programa.

Reference

- https://airflow.apache.org/docs/apache-airflow/2.7.2/index.html
- https://github.com/Microsoft/sql-server-samples/tree/master/samples/databases/adventure-works
- https://learn.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver16&tabs=ssms

Kraj!