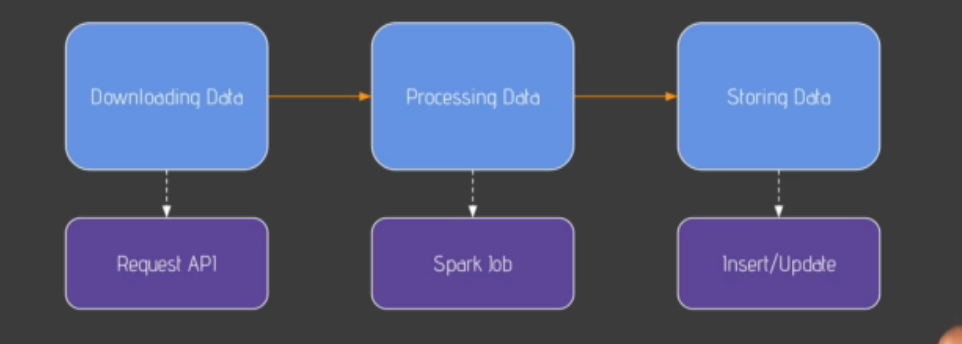
APACHE airflowu neden kullanmaya ihtiyaç duymuşuzdur

Her gün saat 10 da çalışan bir data pipeline ım olsun 3 taskden oluşan



Tüm tasklerimizde extarnool toollar la etkileim halinde

Downloading taski request api ile processing data taski spark ile Stroing data taski de db halinde

Peki bu tasklerde bi sorun çıktı apı çalışmadı yada db deki tabloya daha fazla kayıt ekleyemiyoruz

peki ben bunun gibi yüzlerce data pipeline ı yönetmeye çalışışırsam ne olur

işte bu noktada airflowa ihtiyaç duyulmuştur

data pipelineımızı yönetebilir tasklerimizi güvenilir bir şekilde çalıştırabiliriz

airflow data pıpelınımızı Schedule ve monitörüng edebileceğimiz açık kaynaklı bir platform

airflow ile tasklerimizi doğru zamanda doğru şekilde doğru sırayla çalıştırabiliriz

avantajları:

scabiltiy : istediğim kadar taski workerlara dağıtarak paralel bir şekilde çalıştırabilirim

tabikide bu mimariye ve kaynaklara bağlı celery executorda ne kadar çok worker demek o kadar çok task execute edebilirim demek

ui: data pipelinemızdaki tüm bileşenleri görüntüleyebiliriz

airflow bileşenleri:

webserver : kullanıcı arayüzüne ulaşmayı sağlar

Schedule:tasklerimizi ve data pipelinımızı zamanlamaktan sorumlu

Taskin durumuna bağlı olarak Hangi tasklerin ne zaman hangi sırayla execute edileceğinden sorumlu

Dag run ı başlatan bileşen scheduler dur

Metastore : metadatanın tutulduğu veri tabanı ,dagruns,task instance ,task status,connections variable,xcom bunların hepsi metadata da tutulur bunların hepsini arayüzden görüntüleyebilirim

Default olarak sqlite geliyor ama bu paralel olarak çalışmaya uygun değil o yüzden db yi değiştircez

Dagi Schedule veya mauel olarak trigger ettikçe dag run oluşur

Her task execute edildikten sonra task instance oluşur

Task\_status,taskin durumunu ifade eder arayüzden görüntüleyebilirsin

Succes ,failed,running ….. gibi

Conneciton : dag de external tool kullandıysam örnek postrseql bu db ye bağlanmak için gerekli parametreleri verdiğim kısım

Xcom: taskler arası veri paylaşımını sağlar

variable : örneğin birden fazla dag aynı hedef klasörüne dosya yazıyorsa ben bu hedef klasörünü değiştirmek istediğimde tüm daglere gidip değiştirmem lazım ama variable yaratsaydım ui dan tek lamleyle bu hedef klasörünü güncelleyebilirdim

diğer bileşen

Executor: tasklerin nasıl çalışacağını belirler

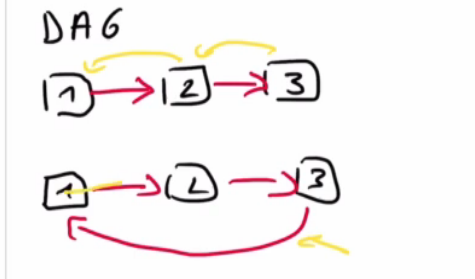
Varsayılan olarak SequentialExecutor olarak geliyor

Buda şu demek benim tasklerim sırayla çalışcak önce biri sonra diğeri sonra diğeri gibi

Tasklerimizi paralel çalıştıramayız bu executorde çünkü metadata db varsayılan olarak sqllite

Sqlite paralel çalışma imkanı sağlamıyor

Worker:tasklerin çalışıtığı yer



Tasklerimin bütünü diyebilriz data pipelina da diyebiliriz

Önce task1 çalışır sonra task2 sonra task3

Diğer bir deyişle task 2 task1 e bağlı task3 task2 ye bağlı eğer task 3 de task 1 e bağlı olursa o zman dag olmaz bir döngü olur benim dagim çalışmalı ve bitmeli

Operator

Operator tasklerde işi yapmamızı sağlar

Örneğin bir mysql db sinde tablo oluşturcam öncesinde mysql yüklü değil ise yüklemem lazım sonrasında connecitonu oluşturmam lazım sonrasında MysqlOperatoru kullanmam lazım

Bu sayede mysql query si yazabilirim

3 tip operator var

https://airflow.apache.org/docs/apache-airflow/stable/\_api/airflow/operators/index.html

1 -) Action Operator:

Örnek BashOperator : bash komutları yazmayı sağlar

PythonOperator : python fonksiyonu yazmamızı sağlar

2 -) transfer operator : kaynak ve hedef arasında data transferi sağlar :

Örneğin prestodan mysql e data transferi sağlamak için

3-)Sensor Operator

Bir sonraki taske geçmeden önce bişey olmasını bekler örneğin

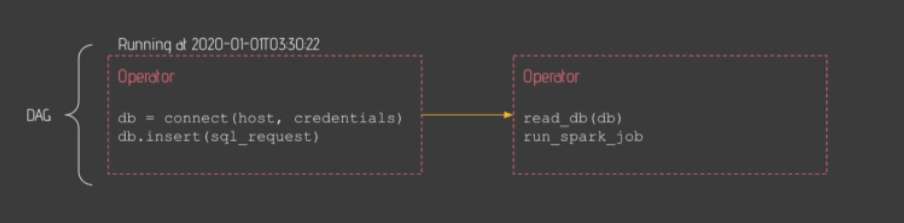
Bir dosyanın belirli bir konuma inmesini bekliyoruz ve indikten sonra da diğer task e geçicez

Bunun içi FileSensor ü kullancaz

Task = data pipeline daki bir operator

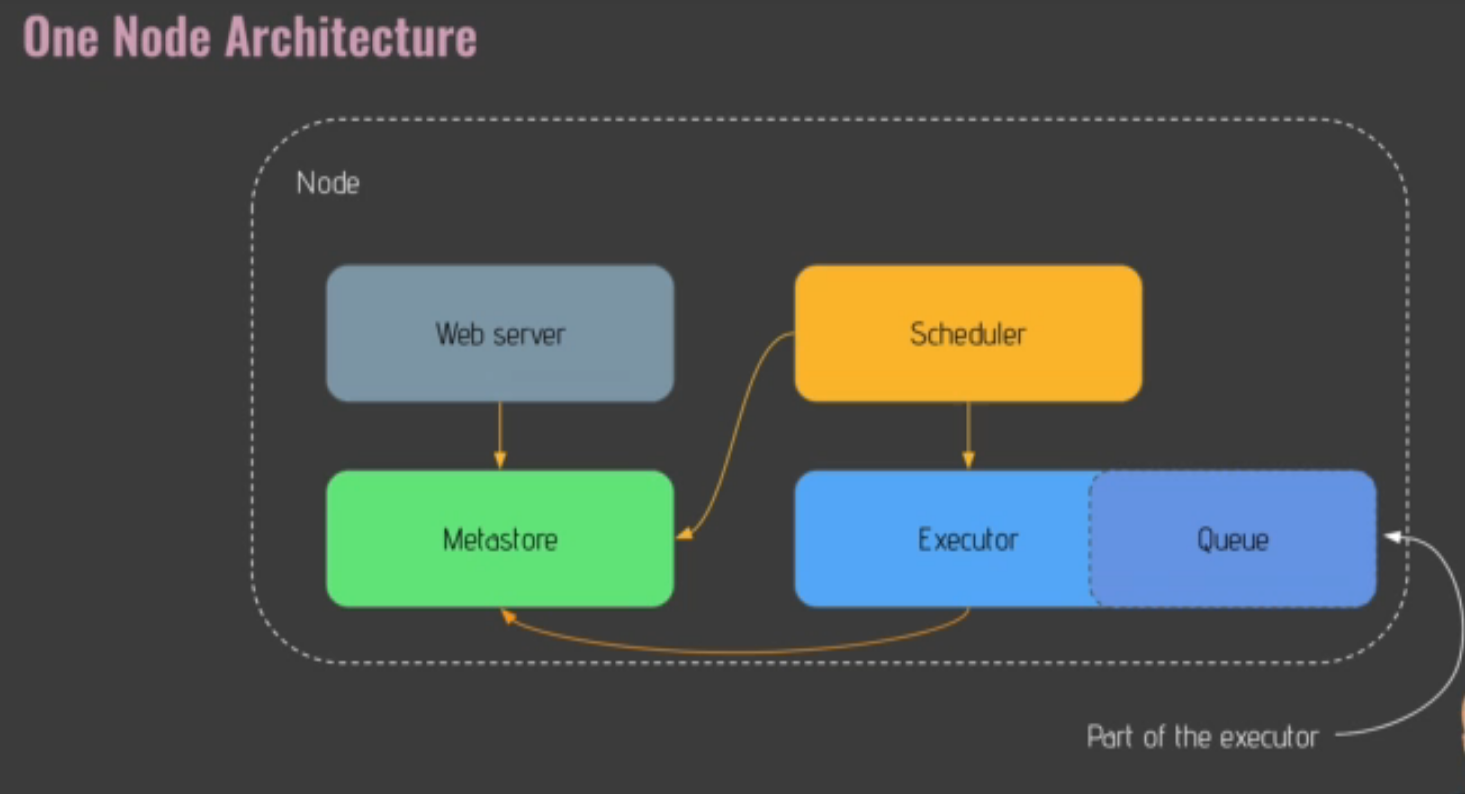
Task instance : operator çalışınca oluşur

Workflow



Tüm conceptlerin kombinasyonuna da worflow diyoruz trigger olmuş dag , operatörler ve operatörler arası bağlılıklar

Airflow nasıl çalışır



Tüm bileşenlerim tek makinede çalışıyor

Webserver metastore dan bazı metadata ları alıyor(task istance dag runs connections vs)

Sheculer da metastore ile iletişim halinde çalıştırmak istediğim tasklerin bilgisini göndermek için

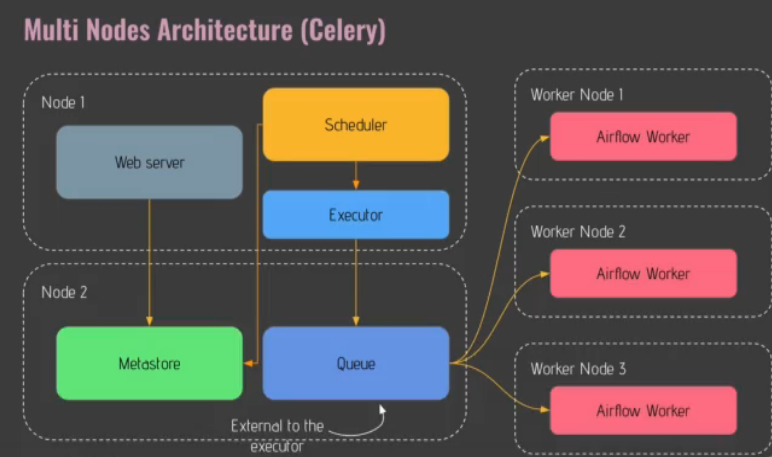
Executor de metastore daki taşkın statusunu güncelliyor

Queue da tasklerimin hangi sırayla çalışacağını belirtir

Bu architectur=localexecutoru airflowu öğrenmek amacıyla kullanılabilir az sayıda task çalıştıracaksak

Peki istediğimiz sayıda task çalıştırmak için hangi mimariyi executoru kullancaz

MultiNodeCluster



Görüldüğü gibi tüm airflow bileşenleri tek makinede değil

Böylece tasklerimiz tek bir makinede yürütülmücek makeniler arasında dağıtılacak

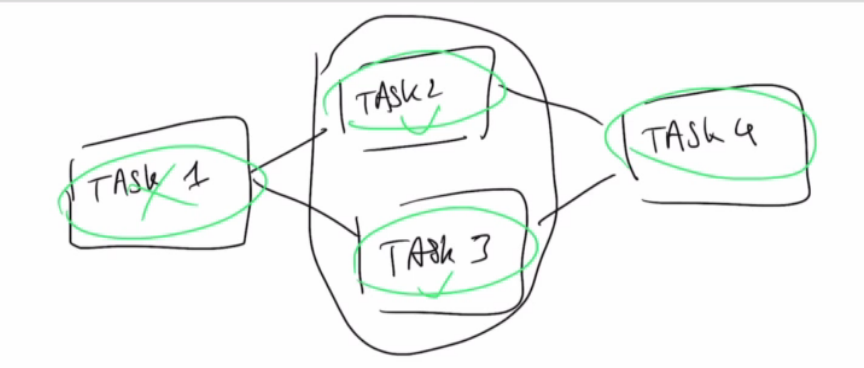
Executor yürütmek istediğim taskleri queue ya gönderir

Queue olarak extarnal tool kullancaz örnek olarak redis

Taskler kuyruğa girdikten sonra airflow worker lar bu taski alçak ve kendi makinesinde çalıştıracak

Bu mimaride istediğim kadar taski çalıştırabilirim daha fazla worker ekleyerek daha fazla worker demek daha fazla task execute edebilirsin demek

Executor



Böyle bir dağim olsun normalde task1 in önce sonra tas2 ve task 3 ün paralel çalışmasını sonrasonda task 4 un çalışmasını beklerim ama executorum SequentialExecutor oludğu için hepsi tek tek çalışcak

Task2 ve task3 ün paralel çalışmasını istiyorsam db mi ve executorumu değiştircem

Executorum LocalExecutor olcak db m airflowun önerdiği db postgres olcak

Önce task1 çalışcak Task1 den sonra hangi taskin çalışcağı garantiside yok task 2 de çalışabilir taks3 de en son task 4 çalışcak

Local executore çevirmek için config dosyasında executoru LocalExecutor yapcam sonrasında postgresi yüklücem posgres için kullanıcı oluşturcam ve config dosyasındaki sql\_alchemy\_conn

Executor: tasklerimizin naıl çalışacağını belirliyordu

Airflowdaki default executor SequentialExecutor bu şu demek aynı anda sadece tek task çalışabilir

Paralel çalışma yok

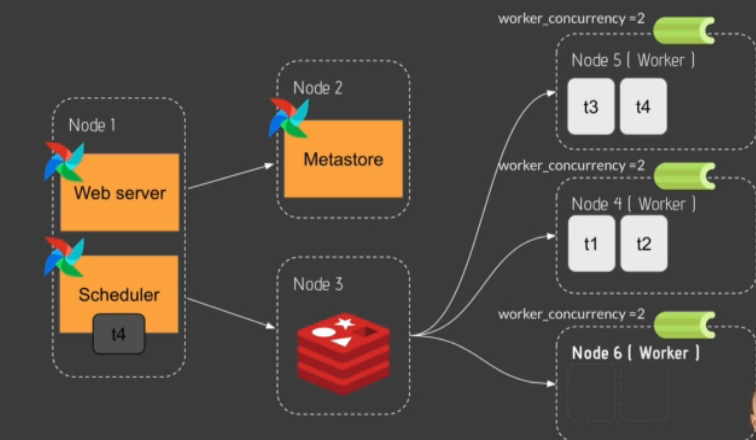
LocalExecutor:

Tek bir makinede taskleri paralel çalıştırmayı sağlar

Bu executoru kulanıyorsak mysql postresql gibi dbleri yüklemek lazım

Airflowu öğrenmek amıcyla kullanılabilir ama dezavantajları var tek makinede kaynak sınırlı olduğıu için buda sınırlı sayıda taskleri çalıştırabilirsin demek tek makine olduğu için o makineye bişey olursa tüm çalışan bileşenleri etkilier

CeleryExecutor



Tasklerimizi bir çok makinede paralel olarak çalıştırmamızı sağlar

Taskleri bir çok makineye dağıttığı için bu da daha fazla task execute edebilmek demek

Queue sistemi için extarnal tool kullanmak gerek redis gibi

Local executorda taskler çalışmadan önce kuyruğa alınıyordu sonrasında çalışmaya hazır olduğunda kuyruktan çıkıp çalışıyordu burda onun yerine redis var

Taskler rediste kuyruğa alınıyor sonrasında workerlera dağıtılıp workerlarda çalışıyor

Celery Executoru kullanırken dikkat etmem gerekenler

Her task için import etttiğim providerlar operatörler modüllerin hepsi tüm workerlarda kurulu olmalı

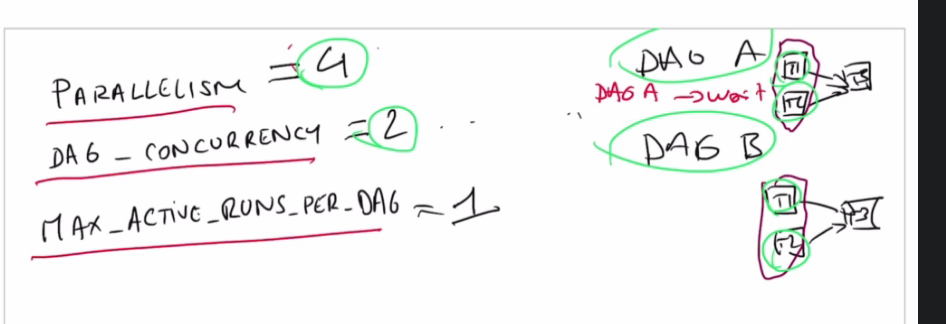
Örneğin task1 de mysqloperatorunu kullandım bu task worker 1 de

Tek worker 1 de mysql provider ın yüklü olması yeterli değil tüm workerlarda yüklü olmalı

Worker concurrency paramatresinin değerini worker ın kaynaklarına göre oluşturmam lazım

Bu workerı kullanıyorsam queue sistemi için harici bir tool yüklemeye ihtiyaç var redis gibi

Config dosyasında bilmem gereken paramatreler



base\_log\_folder:log dosyanın tutludğu klasör cloud da tutulabilir

Parallelism : airflowda paralel olarak çalıştırabileceğim task sayısı

Sadece bir dag de değil o an çalışan tüm daglerde paralel çalışcak task sayısı varsayılan olarak 32

Dag\_concurrency : yalnızca tek dag de çalışacak paralel task sayısı

Yani bir dagde max 16 tane task paralel çalışabilir

Varsayılan olarak 16

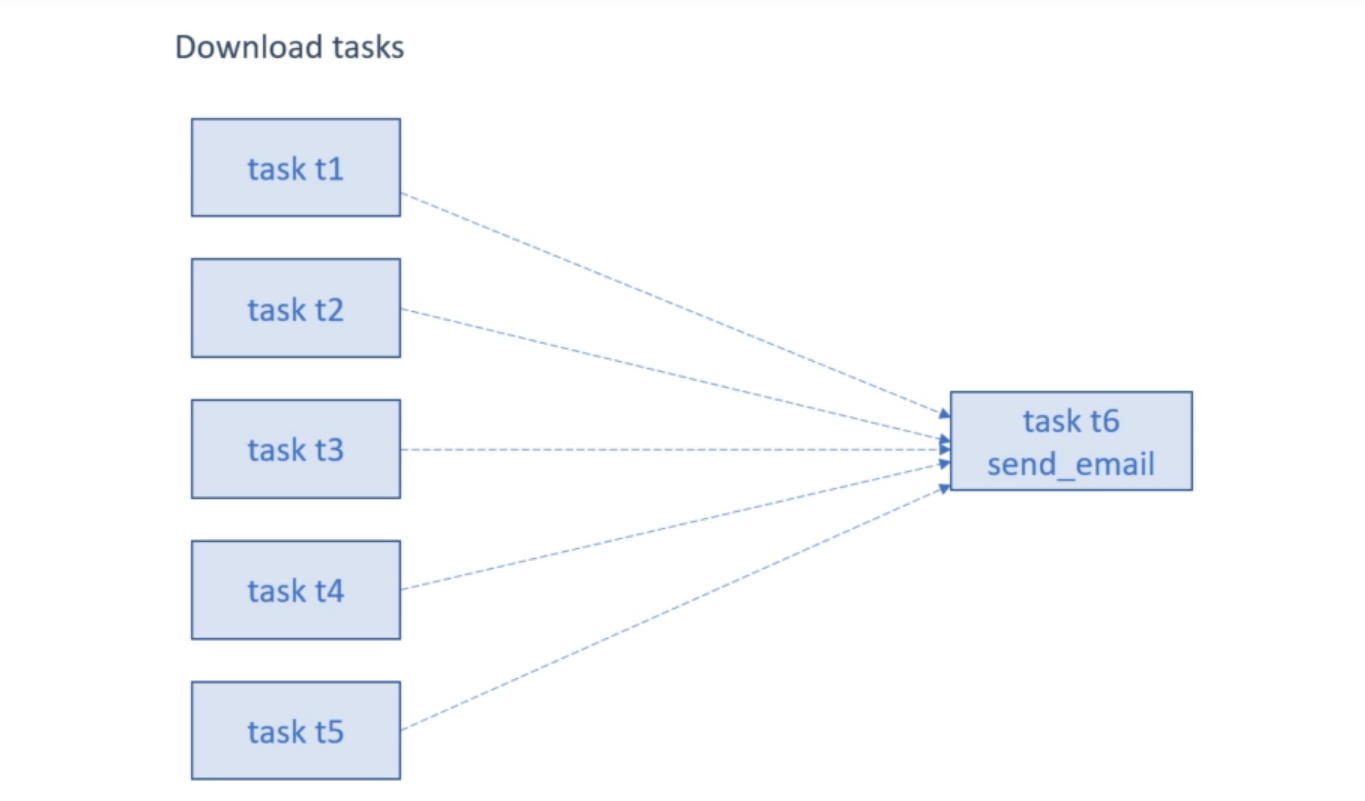
Max\_active\_dag\_runs\_per\_dag:aynı anda çalışacak dag run sayısı varsayılan olarak 16

result\_backend : workerlar tarafından tamamlanan tasklerin raporu nerede tutulmalı

dag\_dir\_list\_interval :varsayılan 300 saniye

dag klasörünü kontrol ettiği süre yeni dag veya güncellenen dag var mı diye

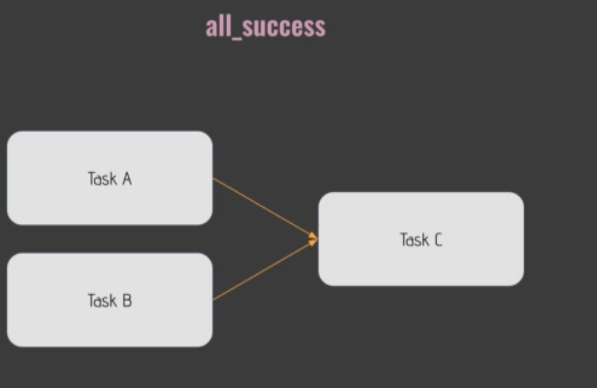
Trigger Rules



Task t6 nın success olması için t1 t2 t3 t4 t5 in success olması lazım varsayılan olarak böyle

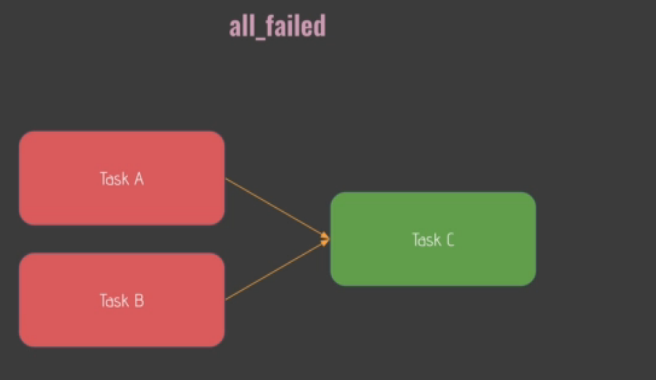
Ama ben t11 t2 t3 t4 t5 tasklarından biri bile succes olsa t6 nın success olmasını istiyorum

İşte trigger kuralları bu yüzden var hangisi işimiz yarıyorsa onu kullancaz



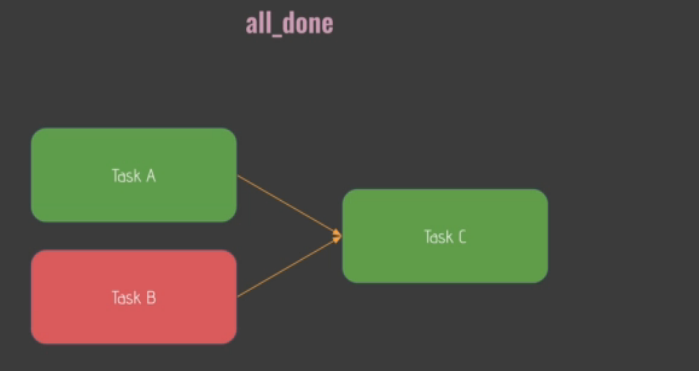
Varsayılan olarak bu şekilde task c nin success olması için Taska ve task b nin succes olması lazım

Eğer task a taskb den biri failed olursa task c nin statusu up\_stream\_failed

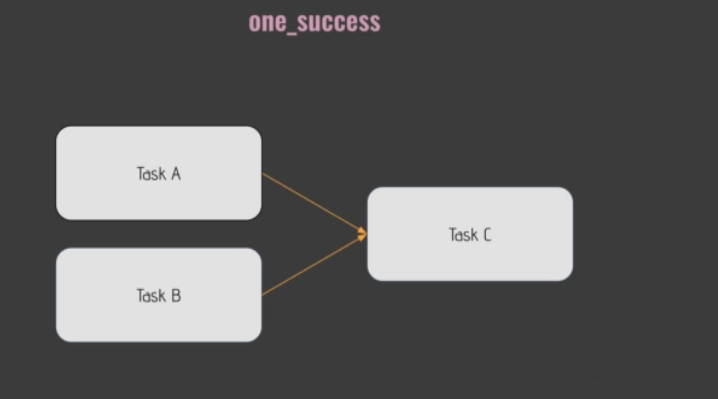


Eğer task cnin success olmasını istiyorsam task a ve b fail olmalı

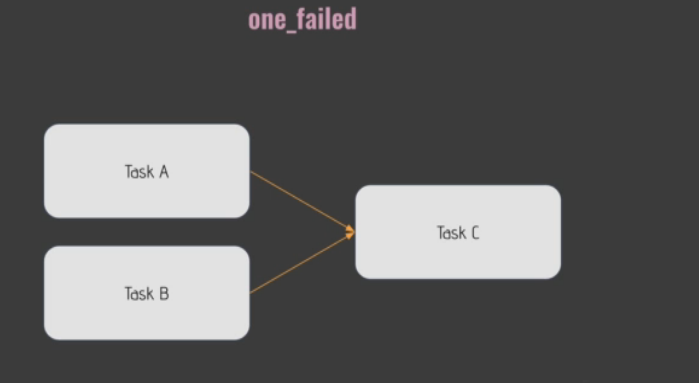
Eğer iksinden biri fail ise task c skip statüsünde olur



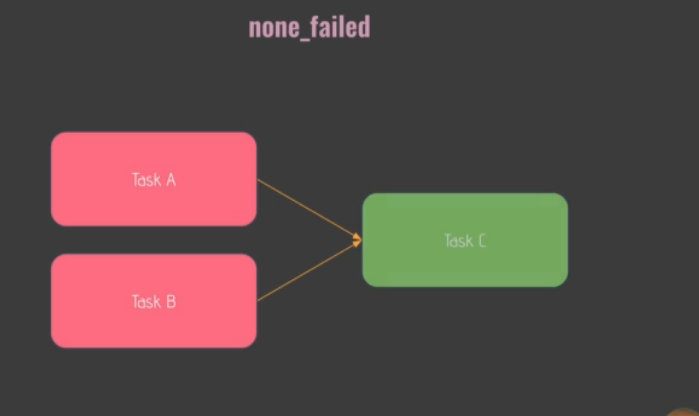
Task A ve Task B nin statusu ne olursa olusn task c sucess olur



Tasklerden biri sucess olursa diğer taskin statusunu beklemeden task c de succes olur



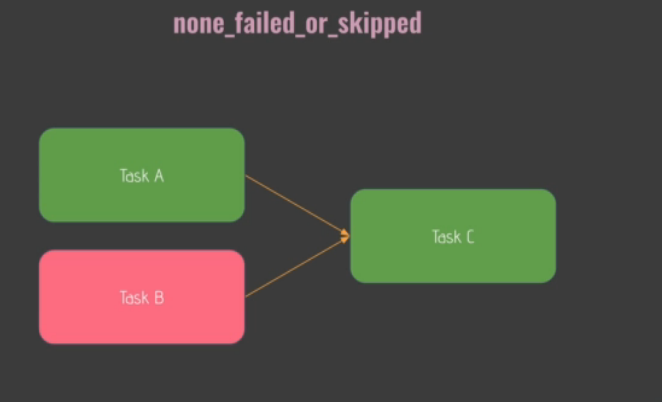
Tasklardan biri fail ise diğer taskin durumunu beklemeden success olur



Task c nin success olması için 2 taskında fail olmaması gerekiyor

Task a ve task b succes ise task c de success olur

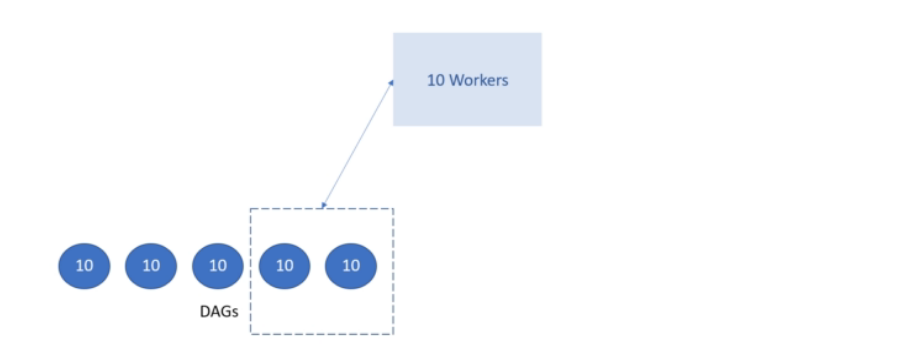
Task a ve task b de skip ise task c success olur



None failed den farkı olarak task cnin success olamsı için tasklardan birinin sucess olması lazım

Task cnin success olması için task a ve task b fail olmamalı yada task a ve b skip olmamalı

Pools:



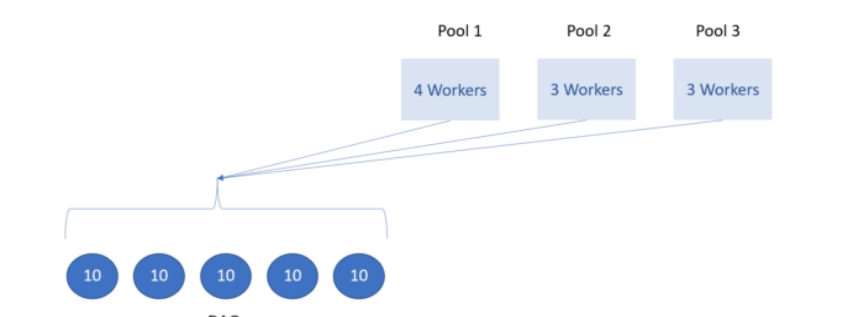
5 dagin ve her dag de 10 tane paralel çalışan taskin var ve 10 workerın var

Diyelim ki 2 dag deki tüm taskler tüm workerlara dağıtıldı

Diğer 3 dag boşta workerlardaki taskler serbest kalana kadar

Bu tür olaylardan kaçınmak Pool bu noktada devreye giriyor

Workerlara eşit sayıda task dağıtmak için pool kullancaz



Taskler sadece execute ediliği workerdaki kaynakaları kullanabilir

Arayüzden admin den oluşturabilirsin

Slots kısmına kaç tane worker olcak onu yazcaz pool 1 de 4 workers var bu da şu demek

4 tane paralel task çalıştırabilirim

Pool kullandıktan sonra şöyle olur dag1 dag2 dag 3 dag 4 dag 5 deki 4 task pool 1 de çalışır

3 task pool 2 de çalışır son 3 task da pool 3 de çalışır bu sayede tüm workerlarıma eşit sayıda task

paylaştırabilirim

Sensor opearatörleri:

Diğer taska geçmesi için bişeyin olmasını bekler

Örnek bir klasöre dosyanın gelmesini bir tabloya datanın gelmesini gibi

Aşağıda sensor operatörleri yer alıyor



Bir olayın gerçekleşmesi için beklemeye devam eden peratörlerdir eğer beklediği şey gerçekleşirse task success olur diğer türlü beklemeye devam eder belirli bir zaman aralığından sonra yeniden denicdek ve olayın olup olmaduğını kontrol edicek

Bir dosyanın bir klasöre ulaşmasını beklemek için FileSensoru kullancaz

Task in bir süre beklemesi gerekiyorsa timeDeltaSensoru var

Sql ile uğraşıyorsak Sqlsensoru

Sensor operatörleri için paramatreler

Poke\_interval: sensor operatörünün beklediği olay gerçekleşmediği zaman tekrar denemede bekleyeceği süre saniye cinsinden tanımlanır

Örneğin 5 saniye verirsek beklediğimiz olay ilk denemede gerçekleşmediyse 5 saniyede 1 denemeye devam edicektir

Sensor operatorunun denemeye ne kadar devam etmesi gerektiğini timeout paramatresi belirler

Sensor\_operatorunu kullanan task in statususu failed olur timeout dan sonra hala olay gerçekleşmediyse

Bir diğer paramatre soft\_fail paramatresi

Eğer True değeri verilirse beklediğimiz olay gerçekleşmediyse o task skip statüsünde olur

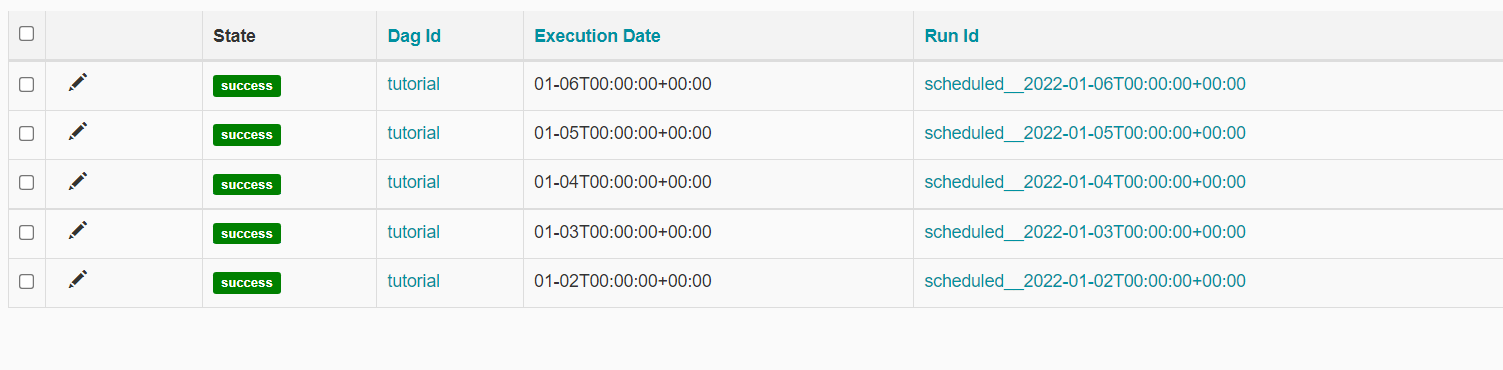
Catchup Paramatresi:

Varsayılan olarak True geliyor bu da şu demek

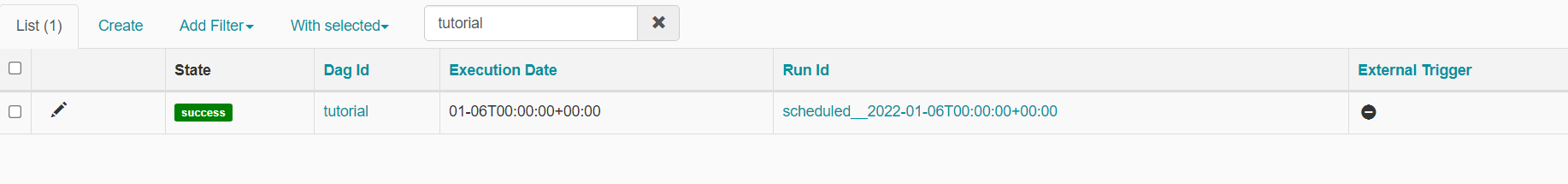
Benim dağimin start date i 2 ocak 2022 olsun ve bu dag günlük çalışsın

Ben bu daği ilk defa 7 ocak da çalıştırırsam 7 ocakta çalıştırdığım saat 14.08

Toplamda 6 tane dag run oluşcak demek daha önce çalışması gereken ama çalışmayan dag run larda oluşcak bu dag runlar 2 ocak 3 ocak 4 ocak 5 ocak 6 ocağın dag runları 7 ocaktaki oluşma çünkü dag 23.59.59 dan sonra çalışıyor



Eğer false olsaydı o zaman 6 ocaktaki dag run oluşurdu yani daği başlattığım tarhten önceki oluşması gereken dag run



Diyelim ki benim dagimin start date i 7 ocak olsun Schedule interval ı da 6 saat de bir olsun

Benim ilk dag run ım ın oluşacağı tarih 01-07T06.00 da yani 7 ocak sabah 6 da oluşcak ben sabah 6 dan önce diyelim saat 5 te arayüzde dagi başlatma tuşuna bastım ozman dag çalışmaz saaatin 6 olmasını bekler saat 6 olunca çalışır

2.dag run 01-07T12.00 da oluşur bu tarihe scheduled\_time yani dag run un oluştuğu tarih

3.dag run 01-07T18.00 de oluşur

4.dag saat 01-07T00.00 da oluşur

5.dag saat 01-07T06.00 da oluşur bu şekilde

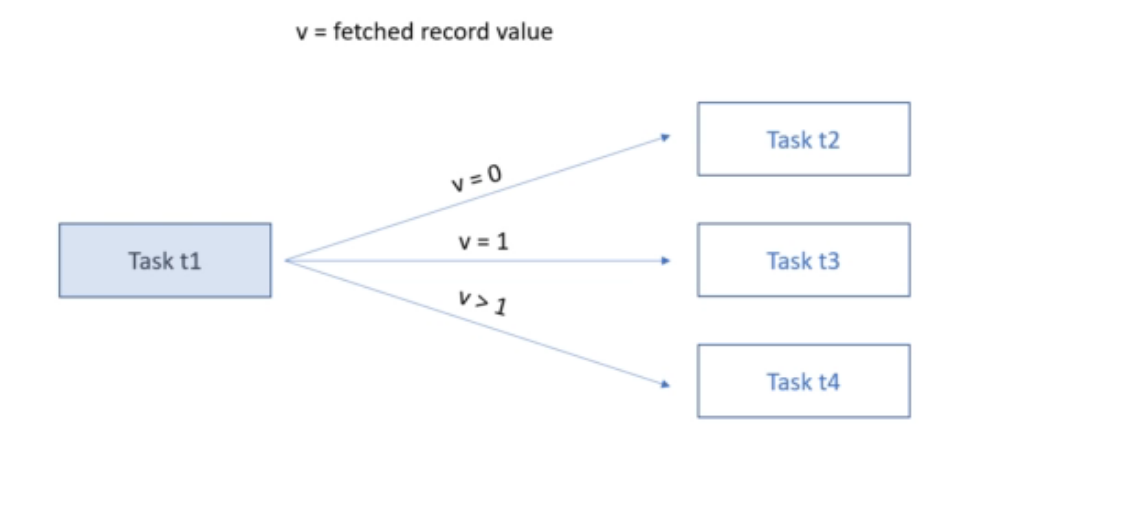
Ozman benim 90 dakkada bir çalışan bir dagim var start date i 11 ocak olsun

11 ocak günü saat 15 .42 de çalıştırdım toplam kaç tane dag oluşur

ilk dag 01.30 da sonraki 3.00 de sonraki 4.30 sonraki 6 da sonraki 7.30 da sonraki 9 da sonraki 10.30 da sonraki 12 de sonraki 1.30 da sonraki 3 de oluşur bu da demek ki benim 10 tane dag run ım oluşur

sonraki 10 sonraki 11 sonraki 12 de sonraki 13 de sonraki 14 de sonraki 15 de sonraki 16 da oluşur bu demek ki 15 tane dag run um oluşcak

BranchpythonOperator

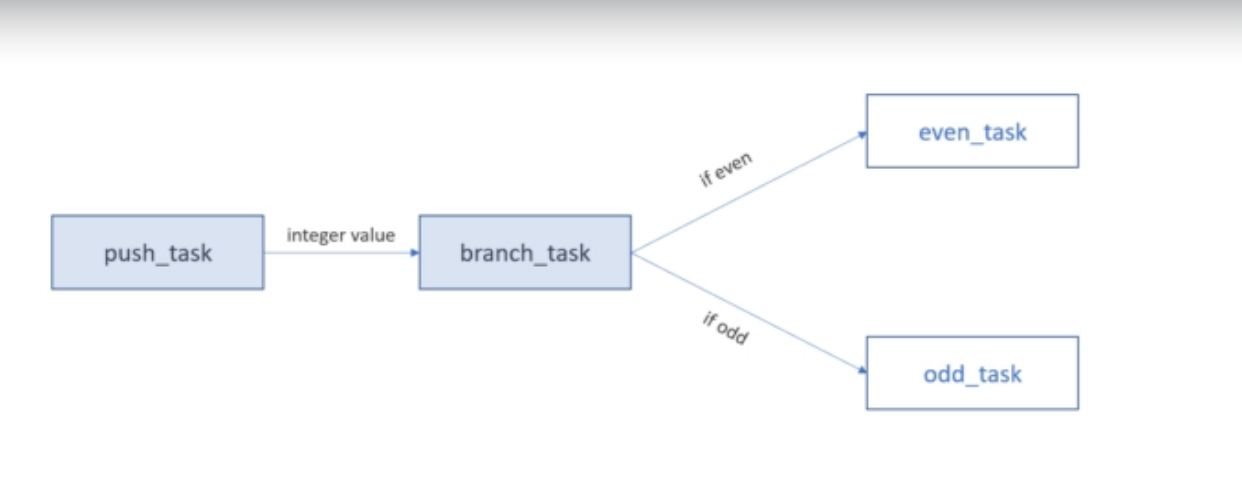


Bir task den sonra belirli bir şarta göre istediğim taski çalıştırma

Pyhton operatöründen farkı

Bir task id yi veya birden fazla task id yi return eder bu sayede hangi şart sağlanırsa hangi task çalışcak

O bilgiyi vermiş oluyoruz



Örneğin sayı çift se even \_task tekse odd\_task çalışssın istiyoruz burada xcom da kullanıcaz push task de xcomu oluşturcaz yani push etcez branch\_task de de xcomu pull etcez yani push task de oluşturduğum integer değeri alcam

Value çift se even\_task değilse odd\_task çalışcak

Branching kullanırken depends\_on paramatresi kullanılması önerilmez

Eğer bu paramatre true olarak ayarlanırsa örneğin bugün even task çalıştı odd\_task taskı skip oldu

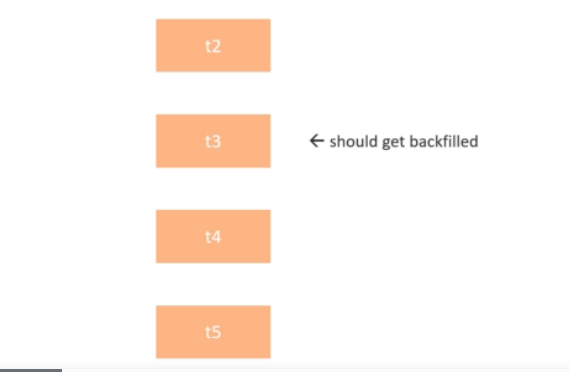
Yarın gelen değer tek olduğu için odd\_task çalışması lazım ama dünkü dag runda odd\_even taski sip olduğu için bu task çalışmaz buda istediğimiz bişey değil o yüzden bu paramatreyi kullanmamak gerekir

LatestOnlyOperator

Catchup paramatresi true olduğunda geçmişte çalışması gereken dag run larda çalışıyordu

Ama dag deki tüm taskler ben sadece istediğim taskler çalışsın istiyorsam ne yapcam

Bu operatörü kullancam



Dyelimki böyle bir dag im var ben bu daği buün çalıştırdım durdurdum sonra 3 gün sonra yimne başlattım catch up true olduğu için o çalışmayan 3 gündeki dag run lar da oluşçak ama işte ben o dag runlar oluşurken sadece task 3 ün çalışmasını istiyorum

LatestOnly operatörü kendinden sonraki tasklerin skip olmasını sağlar eğer şimdiki zaman scheduled time ile bir sonraki scheduled time arasında değilse



Stard date 24 ocak 2020 ydi Schedule interval de 6 saat de bir di

Şimdi buna bağlı olarak ilk dag run 2020-01-24 T 06.00 da oluşcak

