

Inovação com dados em nuvem

OCI Data Flow

Guia para Laboratório Hands-On

Thais Henrique | Andrea Rigoni

Novembro 2021



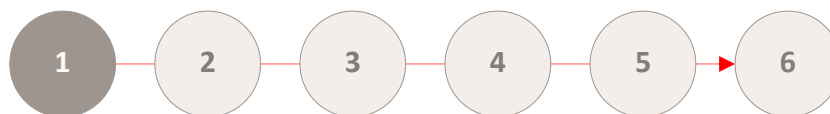
Este trabalho está licenciado sob uma Licença Creative Commons Atribuição-Compartilhagual 4.0 Internacional. Para ver uma cópia desta licença, visite <http://creativecommons.org/licenses/by-sa/4.0/>.

Sumário

1. Pré-requisitos	5
Verificação dos buckets necessários	5
Criação dos buckets no Object Storage	5
Identificação do Namespace do object storage	7
Download dos arquivos utilizados	8
Download do Script Python <i>csv_to_parquet</i> e do Dataset <i>iris.csv</i>	8
2. Configurando o script python	10
3. Transferindo arquivos utilizados para os buckets	12
Após a conclusão do upload clique no botão Close	13
Transferindo o script Python para o bucket	13
Criando a policy para visualização dos log	14
4. Criando sua primeira Dataflow Application	17
5. Executando sua primeira Application - “Run”	20
6. Verificando Logs e o resultado esperado	23

Lab 1.

Pré-requisitos



1. Pré-requisitos

Nesta etapa criar alguns recursos necessários para a criação do Data Flow application

Entre os principais pré-requisitos do Data Flow, e também recursos necessários para a execução desta atividade podemos destacar:

- Criação dos Buckets utilizados pelo Data Flow.
 - Buckets de input e output de dados
- Identificação do Namespace do object storage
- Download do script python de exemplo
- Download do Dataset IRIS em formato CSV
- Criar uma policy para visualizar os logs

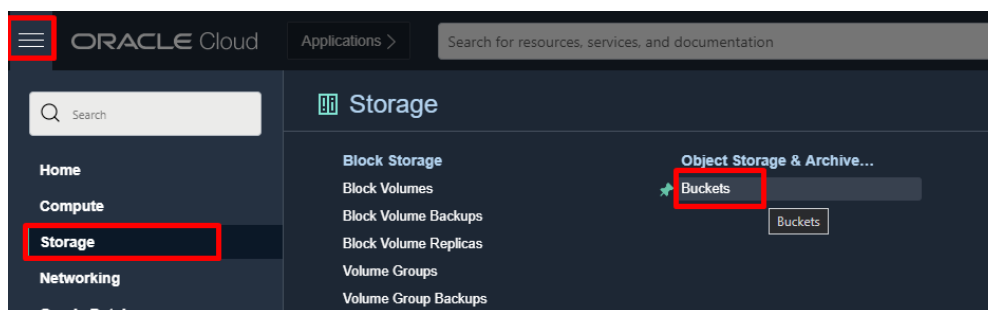
Verificação dos buckets necessários

Nesta etapa criar os seguintes buckets no object storage:

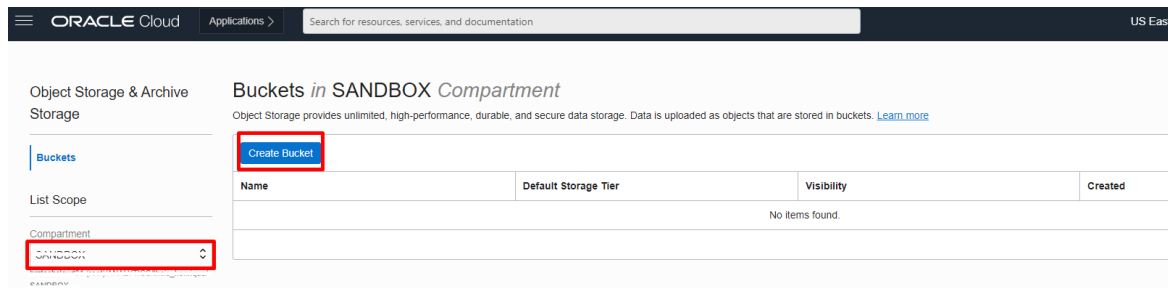
- dataflow-logs
- dataflow-app
- raw-data
- data-out

Criação dos buckets no Object Storage

Expanda a lista de serviços da OCI utilizando o menu de hambúrguer, no canto superior esquerdo. Em seguida, selecione Storage e clique em “Buckets”



Verifique e memorize o compartiment em que você está, lembre-se que você sempre pode alterar o compartiment onde você vai criar seus recursos. Clique em **Create Bucket**



Em Bucket Name escreva 'raw-data' e deixe a opção *Standard* selecionada em "Default Storage Tier". Em seguida clique em Create

Create Bucket

Bucket Name
raw-data

Default Storage Tier
☒ Standard
☐ Archive

The default storage tier for a bucket can only be specified during creation. Once set, you cannot change the storage tier in which a bucket resides. [Learn more about storage tiers](#)

☐ Enable Auto-Tiering
Automatically move infrequently accessed objects from the Standard tier to less expensive storage. [Learn more](#)

☐ Enable Object Versioning
Create an object version when a new object is uploaded, an existing object is overwritten, or when an object is deleted. [Learn more](#)

☐ Emit Object Events
Create automation based on object state changes using the [Events Service](#).

☐ Uncommitted Multipart Uploads Cleanup
Create a lifecycle rule to automatically delete uncommitted multipart uploads older than 7 days. [Learn more](#)

Encryption
☒ Encrypt using Oracle managed keys
Leaves all encryption-related matters to Oracle.
☐ Encrypt using customer-managed keys
Requires a valid key from a vault that you have access to. [Learn more](#)

Tags
Tagging is a metadata system that allows you to organize and track resources within your tenancy. Tags are composed of keys and values that can be attached to resources. [Learn more about tagging](#)

Tag Namespace	Tag Key	Value
---------------	---------	-------

Create Cancel

Repita os passos acima e crie os buckets restantes:

- data-out
- dataflow-logs
- dataflow-app

Após criar todos os bucket você verá todos eles listado como na imagem abaixo:

Object Storage & Archive Storage

Buckets in COMPARTMENT Compartment

Object Storage provides unlimited, high-performance, durable, and secure data storage. Data is uploaded as objects that are stored in buckets. [Learn more](#)

Create Bucket

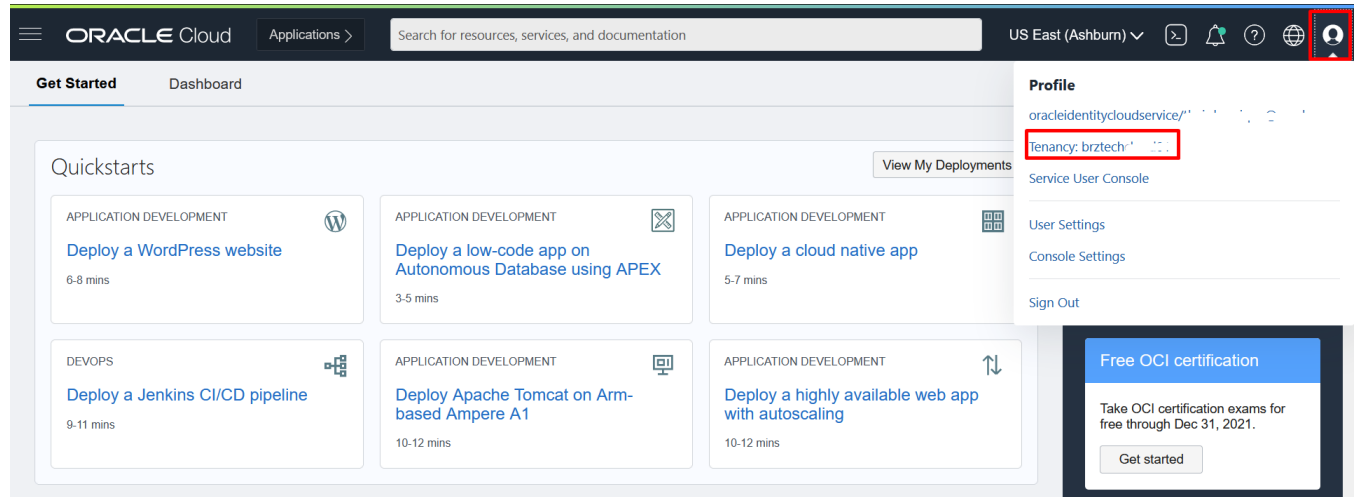
Name	Default Storage Tier	Visibility	Created
data-out	Standard	Private	Thu, Nov 18, 2021, 18:56:20 UTC
dataflow-app	Standard	Private	Thu, Nov 18, 2021, 18:55:25 UTC
dataflow-logs	Standard	Private	Thu, Nov 18, 2021, 18:55:10 UTC
raw-data	Standard	Private	Thu, Nov 18, 2021, 18:54:19 UTC

Showing 4 Items < 1 of 1 >

Identificação do Namespace do object storage

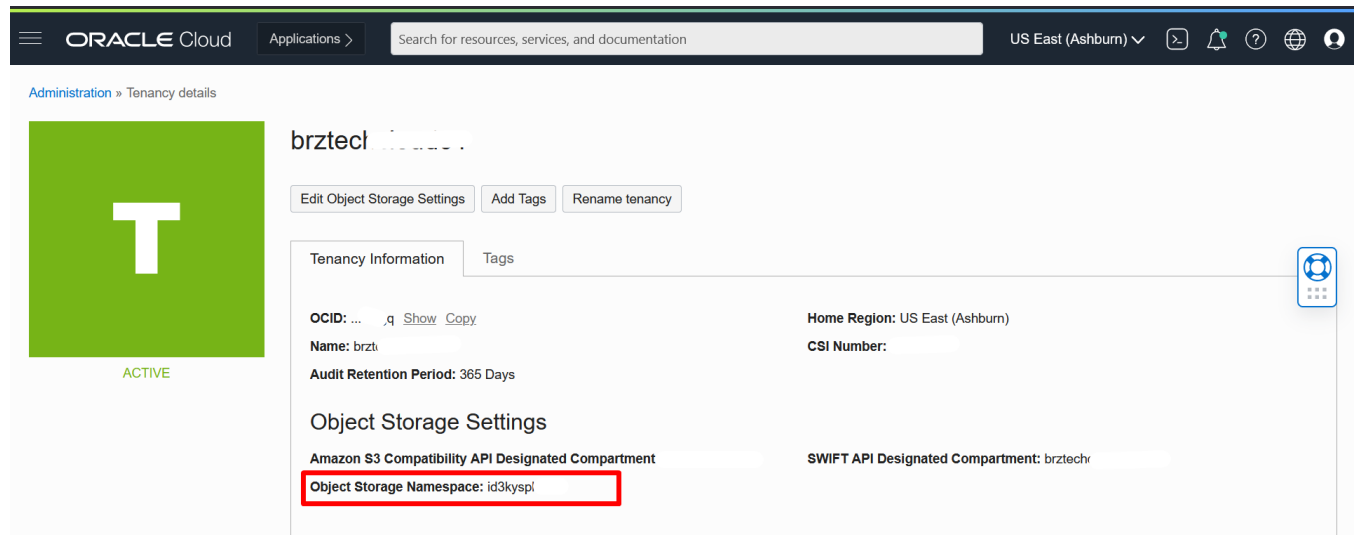
Nesta etapa, coletar o Namespace do object storage do seu ambiente. Esta informação é bastante relevante, pois será utilizada nas etapas de configuração do nosso script python.

Para visualizar e anotar o Namespace do seu ambiente, acesse o menu com seu avatar de usuário no canto superior direito, e clique no nome do seu Tenancy:



Agora nas informações do seu Tenancy, podemos encontrar e tomar nota do Object storage Namespace.

Guarde o nome do Namespace em um notepad ou editor de sua preferência.



Download dos arquivos utilizados

Durante este LAB iremos utilizar dos arquivos, o script python **csv_to_parquet.py** e o dataset **iris.csv** estes arquivos foram disponibilizados em conjunto com os demais arquivos disponíveis no evento, porém caso necessário, poderão também ser encontrados nos links citados abaixo.

Download do Script Python *csv_to_parquet* e do Dataset *iris.csv*

Acesse a seguinte URL e faça o download do script python **csv_to_parquet.py** e **iris.csv**

[https://securesites-prodapp.cec.ocp.oraclecloud.com/documents/link/LF22EA174D34A4760C11446037A2BF21739EBAADA95E/folder/F666AAEA7E31BE38E4FEE3FD8812A00EDB72C63140ED/ OCI Data Flow - Arquivos](https://securesites-prodapp.cec.ocp.oraclecloud.com/documents/link/LF22EA174D34A4760C11446037A2BF21739EBAADA95E/folder/F666AAEA7E31BE38E4FEE3FD8812A00EDB72C63140ED/OCI%20Data%20Flow%20-%20Arquivos)

Senha: **ft-cloud-girls**









OCI Data Flow - Arquivos

Compartilhar Link Membros Fazer Upload Criar ... □

Documentos > Compartilhar > OCI Data Flow - Arquivos

☐ Selecionar Tudo

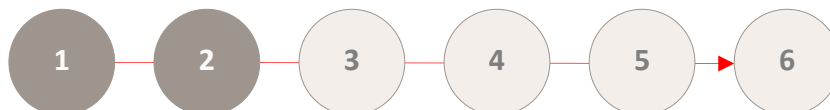
Nome ▾ □ ▾

	Nome ↑	Versão	Última Atualização ↕	Atualizado por	Tamanho	Tipo	
<input type="checkbox"/>	 csv_to_parquet.py	v1	Há 2 minutos	Você	3 KB	PY	  
<input type="checkbox"/>	 iris.csv	v1	Há 2 minutos	Você	5 KB	CSV	  

Salve os arquivos em um diretório de sua preferência.

Lab 2.

Configurando o script python



2. Configurando o script python

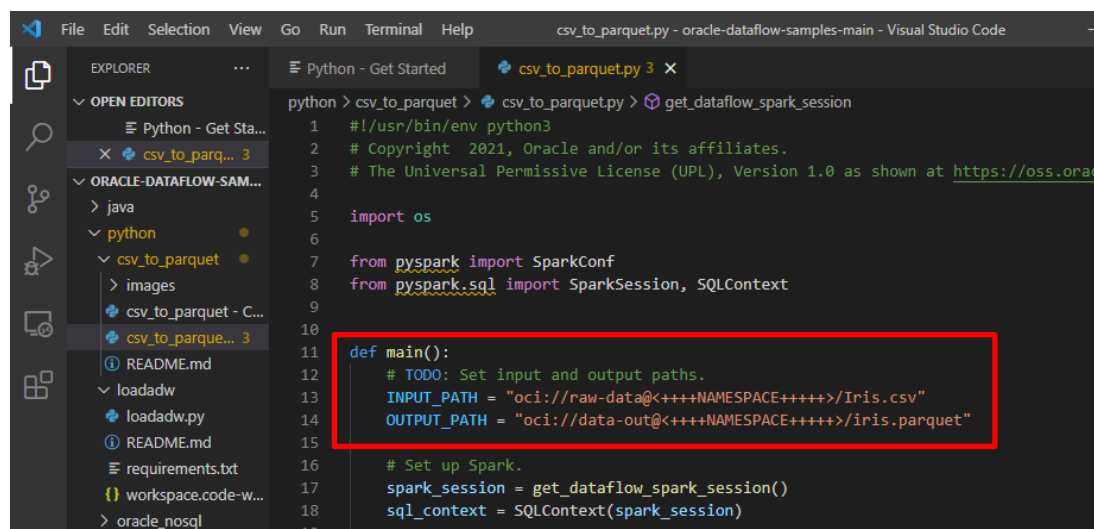
Abra o arquivo **csv_to_parquet.py** com o editor de texto de sua preferência.

Dentro do script, localize as entradas **INPUT_PATH** e **OUTPUT_PATH**, como podemos observar, neste ponto estamos atribuindo a estas variáveis o local de origem do dataset utilizado (**iris.csv**) e também o bucket onde iremos gravar nosso arquivo parquet (**data-out**).

Notem que o formato utilizado para especificarmos um bucket no OCI deverá seguir o seguinte padrão:

oci://<NOME DO BUCKET>@<NAMESPACE>/<NOME DO ARQUIVO OU DIRETORIO>

Exemplo: **"oci://raw-data@id3kyspxxxxx/iris.csv"**



```
1  #!/usr/bin/env python3
2  # Copyright 2021, Oracle and/or its affiliates.
3  # The Universal Permissive License (UPL), Version 1.0 as shown at https://oss.oracle.com/licenses/upl
4
5  import os
6
7  from pyspark import SparkConf
8  from pyspark.sql import SparkSession, SQLContext
9
10
11 def main():
12     # TODO: Set input and output paths.
13     INPUT_PATH = "oci://raw-data@<++++NAMESPACE++++>/Iris.csv"
14     OUTPUT_PATH = "oci://data-out@<++++NAMESPACE++++>/Iris.parquet"
15
16     # Set up Spark.
17     spark_session = get_dataflow_spark_session()
18     sql_context = SQLContext(spark_session)
```

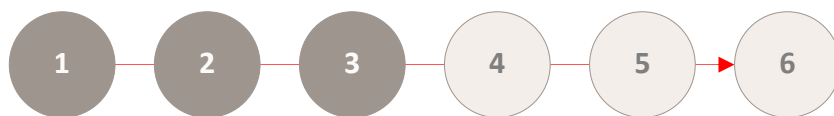
*****importante: Os nomes dos buckets e do dataset iris.csv devem estar escritos corretamente***

Agora utilizaremos o namespace já identificado nas sessões anteriores para ajustar cada um dos apontamentos acima.

Após alterar salve as alterações realizadas.

Lab 3.

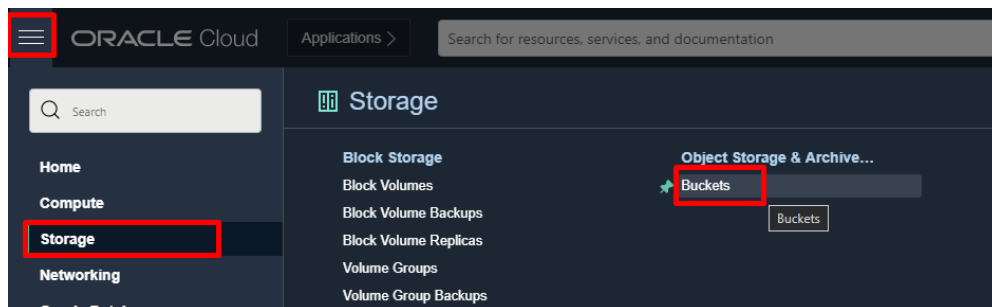
Transferindo arquivos utilizados para os buckets



3. Transferindo arquivos utilizados para os buckets

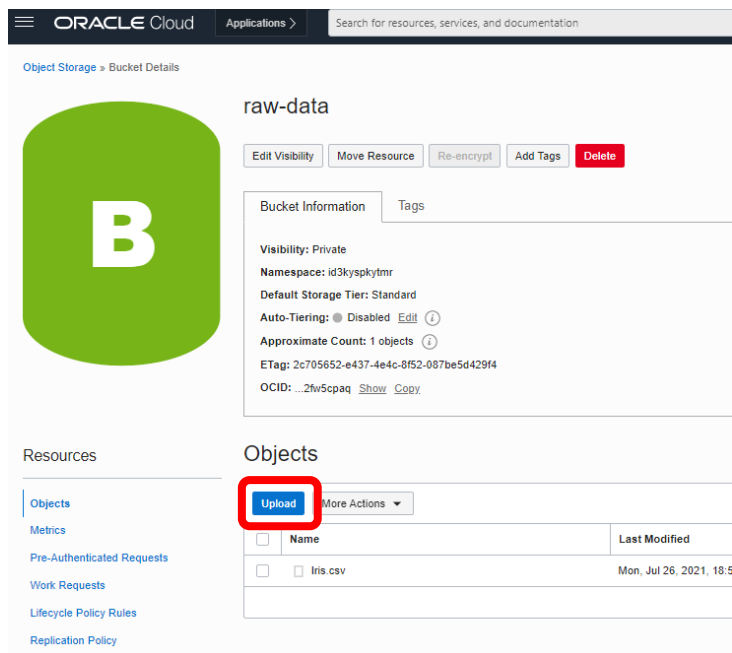
Nesta atividade utilizaremos a própria UI do OCI para fazer o upload dos arquivos para os buckets corretos no object storage.

Para realizar esta transferência devemos acessar o serviço de object storage a partir do menu hamburguer no canto superior esquerdo.

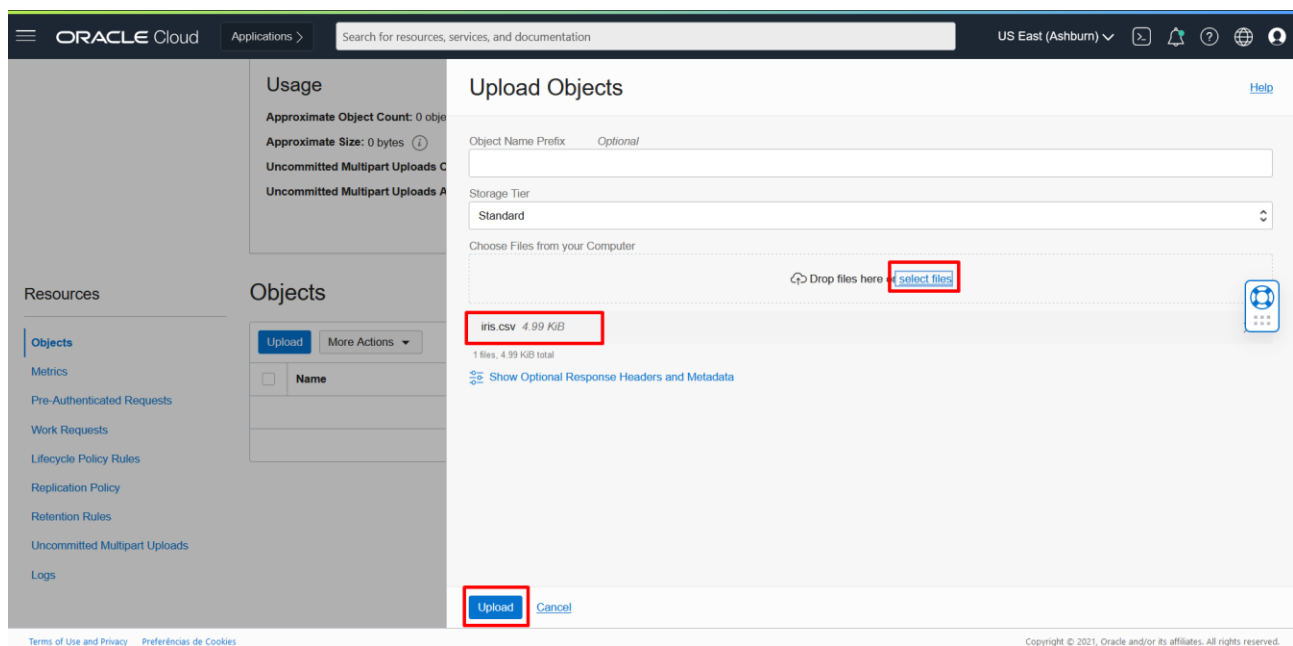


Para transformação, iremos utilizar o dataset **iris.csv** no **bucket raw-data**, conforme indicamos na variável **INPUT_PATH** em nosso script python.

Acesse o **bucket raw-data**, e clique no botão **UPLOAD**



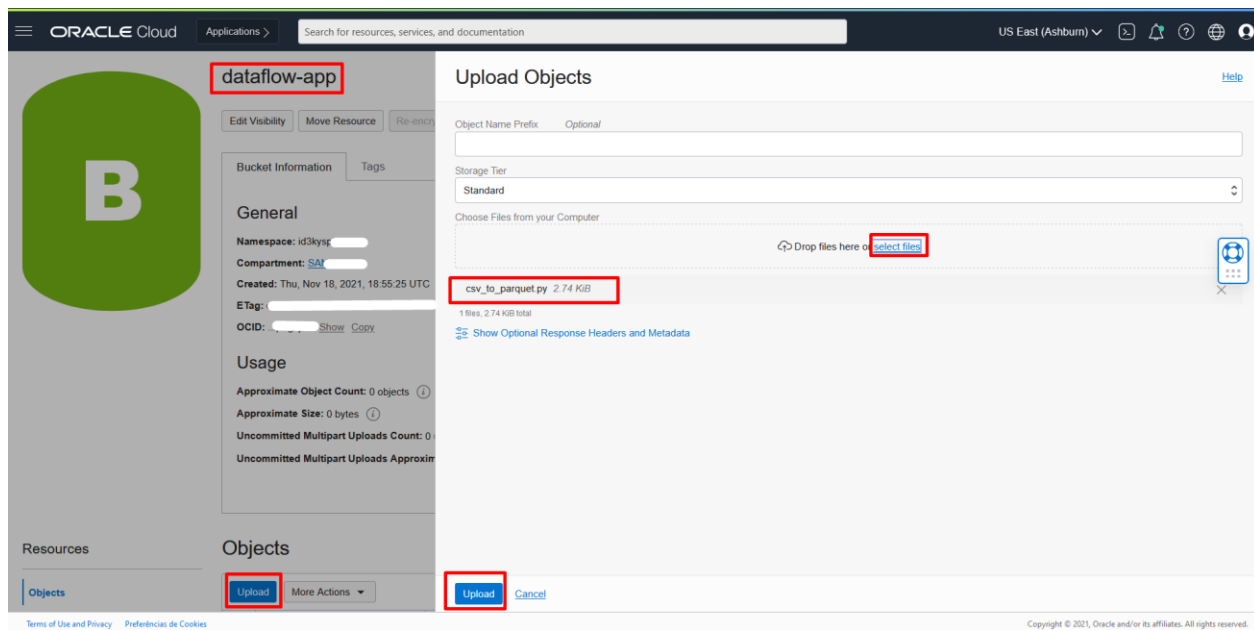
Clique em select e selecione o dataset do iris.csv que você baixou na etapa anterior, depois clique no botão Upload



Após a conclusão do upload clique no botão Close

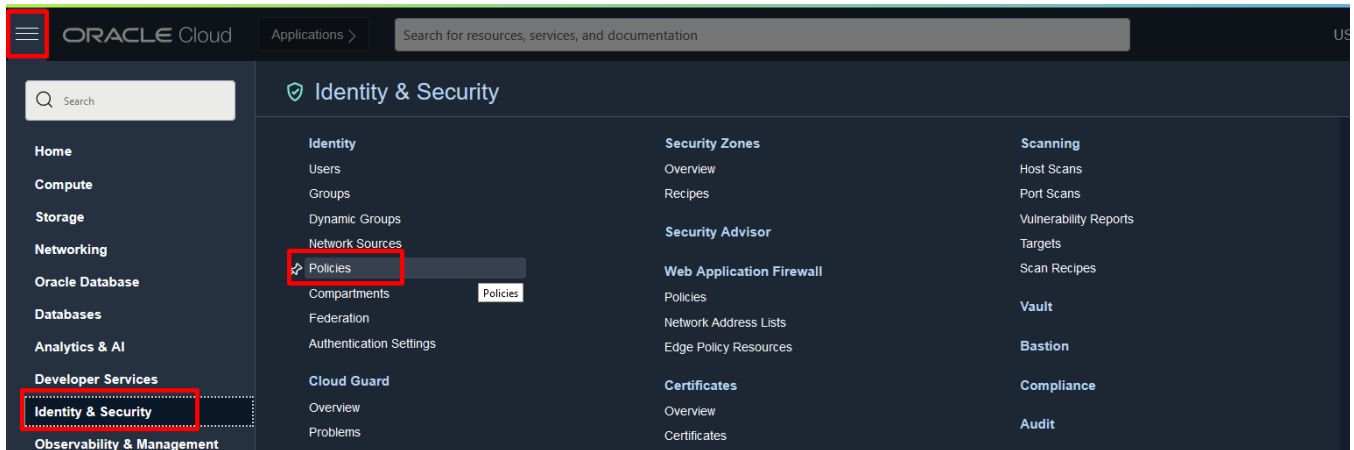
Tranferindo o script Python para o bucket

Seguindo o mesmo procedimento executado para o upload do arquivo iris.csv, agora iremos transferir o script python (csv_to_parquet.py) **já alterado** para o bucket **dataflow-app**

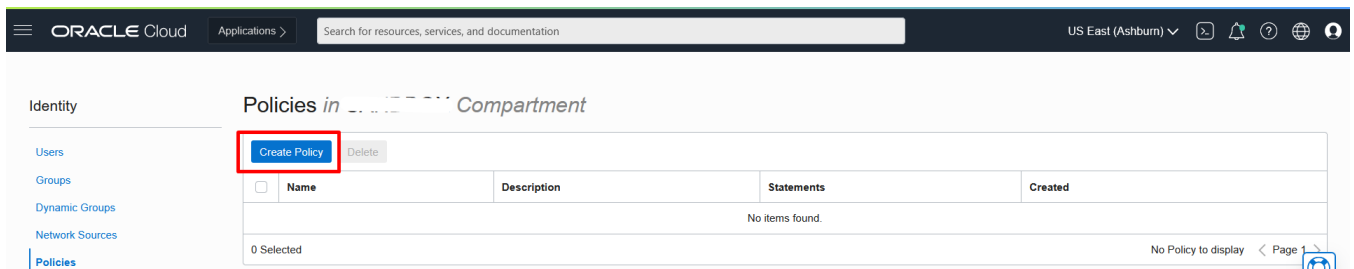


Criando a policy para visualização dos log

Expanda a lista de serviços da OCI utilizando o menu de hambúrguer, no canto superior esquerdo. Em seguida, selecione Identity & Security e clique em “Policies”



Em seguida clique em “Create Policy” (Você pode adicionar esse regra em uma policy já existente)



Preencha o Nome, a Descrição, Selecione um compartment. Em seguida Clique no botão ao lado de “Show manual editor” e cole a policy no campo indicado na imagem. Após preencher todos campos clique em Create.

ORACLE Cloud Applications > Search for resources, services, and documentation US East (Ashburn) Help

Create Policy

Name: **policy-data-flow**
No spaces. Only letters, numerals, hyphens, periods, or underscores.

Description: **Policies para o OCI Data Flow**

Compartment: **<compartment>**
dataflow-logs (target.bucket.name='dataflow-logs', any {request.permission='OBJECT_CREATE', request.permission='OBJECT_INSPECT'})

Policy Builder ☒ Show manual editor

allow any-user to manage objects in tenancy WHERE ALL {target.bucket.name='dataflow-logs', any {request.permission='OBJECT_CREATE', request.permission='OBJECT_INSPECT'}}
 allow any-user to manage dataflow-run in tenancy

Example: Allow group [group_name] to [verb] [resource-type] in compartment [compartment_name] where [condition]

☐ Create Another Policy

allow any-user to manage objects in compartment <compartment> WHERE ALL {target.bucket.name='dataflow-logs', any {request.permission='OBJECT_CREATE', request.permission='OBJECT_INSPECT'}}

allow any-user to manage dataflow-run in compartment <compartment>

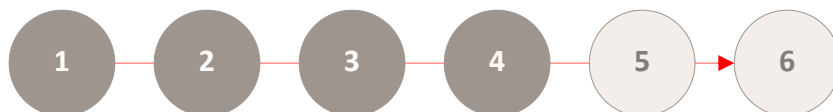
*Se você estiver utilizando o compartment ROOT substitua o text <compartment> por TENANCY

*Você pode configurar as policies com mais níveis de Granularização:

<https://docs.oracle.com/en-us/iaas/data-flow/using/policies.htm#policies>

Lab 4.

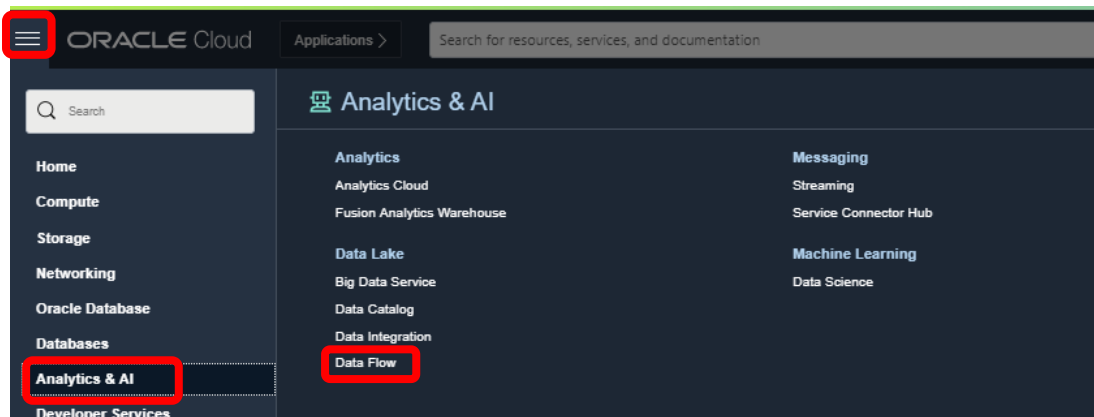
Criando sua primeira Dataflow Application



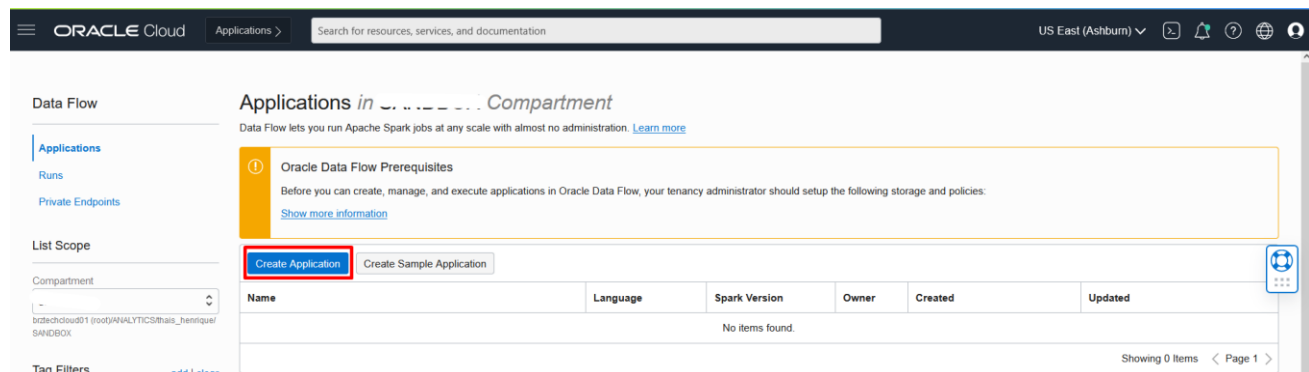
4. Criando sua primeira Dataflow Application

Neste passo iremos criar nossa primeira Dataflow application. O objetivo desta application será transformar o dataset iris.csv em um arquivo no formato parquet, que poderá ser consumido por outras ferramentas.

A console do serviço Data Flow pode ser acessada através do menu Hamburguer no canto superior esquerdo da tela, e depois acessando o item. Analytics & IA -> Data Flow:



Dentro da console do Dataflow, vamos clicar em “Create Application”



Após clicar no botão, iremos fornecer as informações básicas para criação de nossa primeira aplicação

Create Application

General Information

Name ⓘ

csv2p

Description *Optional* ⓘ

Converte Iris csv para Parquet

Vamos definir um nome para nossa application. Neste exemplo estamos utilizando: **“csv2p”**

Atualmente podemos executar nossas apps utilizando SPARK nas versões 3.0.2 ou 2.4.4, neste exemplo

Resource Configuration

Spark Version

Spark 3.0.2 -> Scala 2.12

Driver Shape ⓘ

VM.Standard2.1 (15 GB Memory, 1 OCPU, 175 GB Block Volume)

Executor Shape ⓘ

VM.Standard2.1 (15 GB Memory, 1 OCPU, 175 GB Block Volume)

Number Of Executors ⓘ

1

Agora precisamos definir a capacidade de processamento disponibilizada para nossa application, esta definição é feita em 3 etapas: **Driver Shape, Executor Shape e Number of Executors**, neste exemplo

Application Configuration

☐ Use Spark-Submit Options ⓘ

Language

☐ Java ☒ Python ☐ SQL ☐ Scala

Select a File ⓘ

☐ Enter the file URL manually

Object Storage File Name in OSALAB [\(Change Compartment\)](#)

dataflow-app

csv_to_parquet.py

Como ultimo passo, devemos especificar qual é a linguagem utilizada em nossa APP, e também informar o bucket e o script que contém o código. **Selecionar o bucket dataflow-app e o script csv_to_parquet.py**

Arguments *Optional* ⓘ

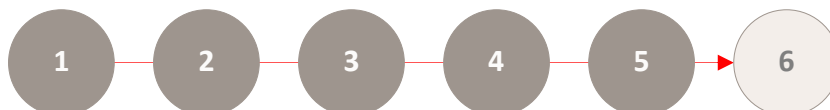
Submitter ID ⓘ

Create

Cancel

Lab 5.

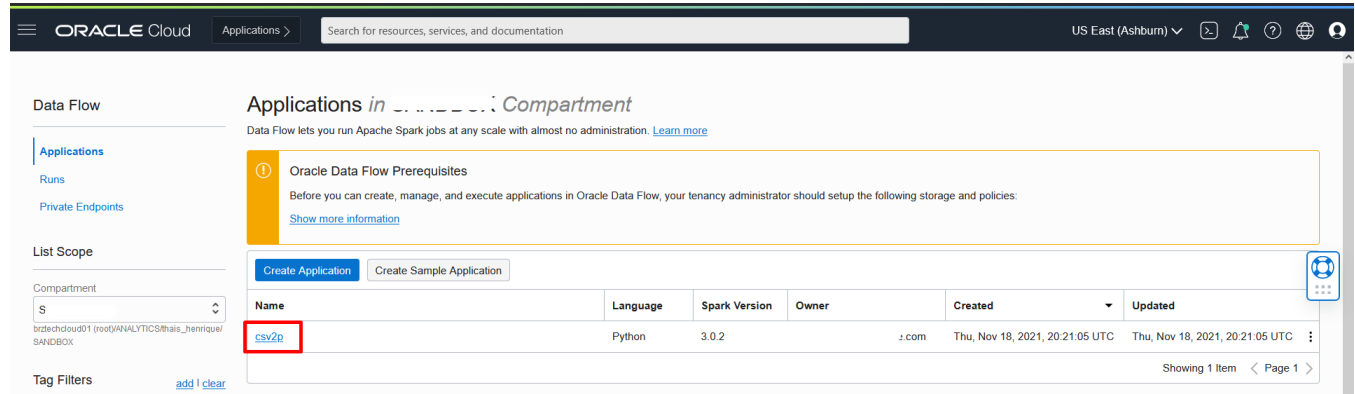
Executando sua primeira Application - “Run”



5. Executando sua primeira Application - “Run”

Após criação de sua “Dataflow App”, agora podemos executar o código quantas vezes necessário através do botão RUN na console ou através de linha de comando com OCI CLI / API

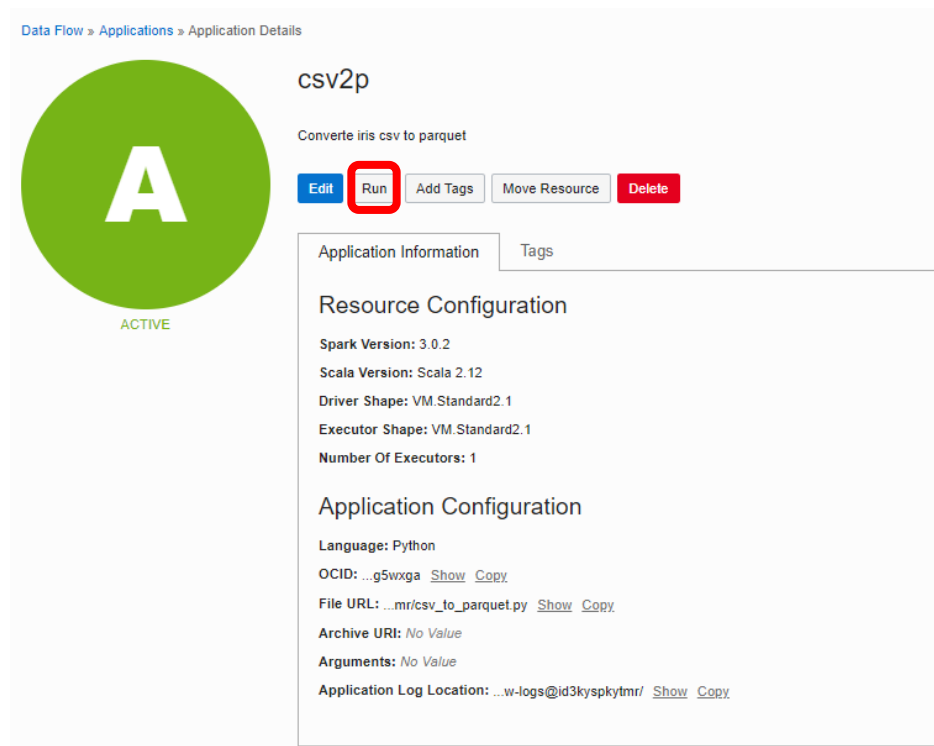
Neste exemplo iremos executar o script Application csv2p através da console:



The screenshot shows the Oracle Cloud Applications console. On the left, there's a sidebar with 'Data Flow' and 'Applications' sections. The main area displays 'Applications in ... Compartment'. A yellow warning box at the top states 'Oracle Data Flow Prerequisites'. Below this, there are buttons for 'Create Application' and 'Create Sample Application'. A table lists applications, with 'csv2p' highlighted by a red box. The table columns are Name, Language, Spark Version, Owner, Created, and Updated.

Name	Language	Spark Version	Owner	Created	Updated
csv2p	Python	3.0.2	...com	Thu, Nov 18, 2021, 20:21:05 UTC	Thu, Nov 18, 2021, 20:21:05 UTC

Dentro da Application criada csv2p, podemos encontrar diversas informações relacionadas a aplicação, além do botão “RUN” que iremos acionar para uma nova execução:



The screenshot shows the 'Application Details' page for 'csv2p'. On the left, there's a green circle with a white 'A' and the word 'ACTIVE' below it. The main area has a title 'csv2p' and a description 'Converte iris csv to parquet'. Below this are buttons for 'Edit', 'Run' (highlighted with a red box), 'Add Tags', 'Move Resource', and 'Delete'. The page is divided into two tabs: 'Application Information' and 'Tags'. The 'Application Information' tab is active, showing 'Resource Configuration' and 'Application Configuration' sections.

Resource Configuration

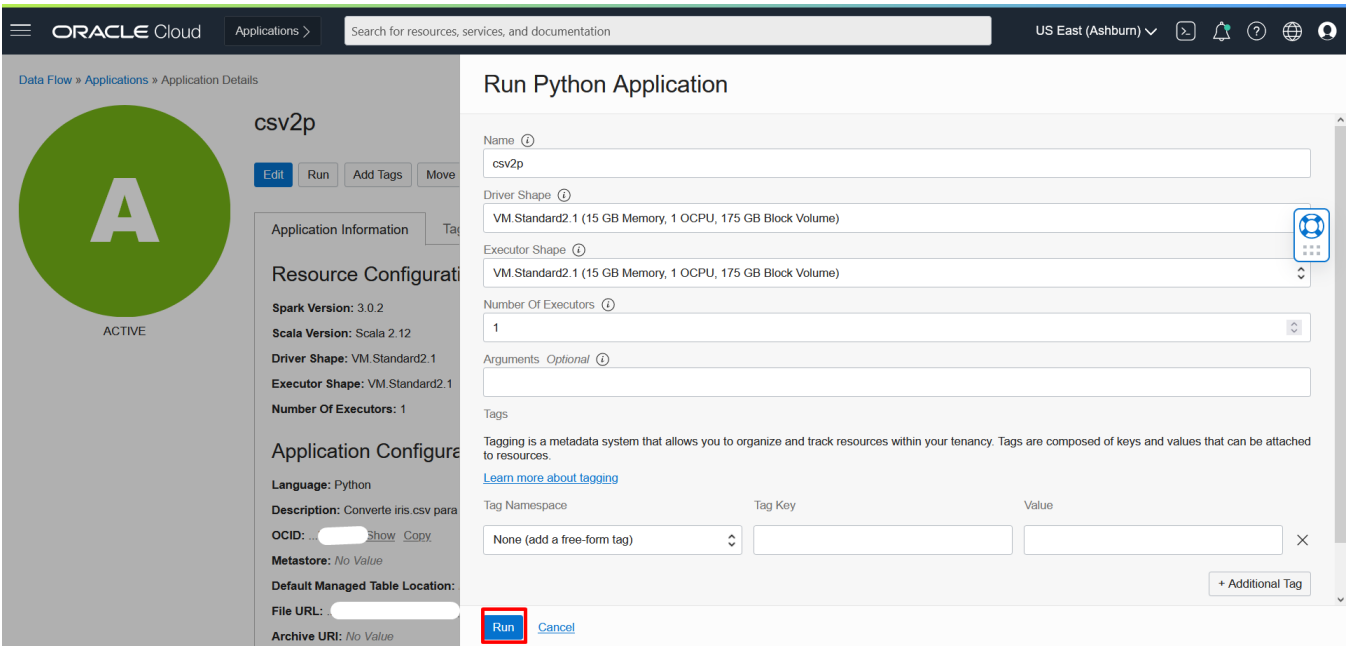
- Spark Version: 3.0.2
- Scala Version: Scala 2.12
- Driver Shape: VM.Standard2.1
- Executor Shape: VM.Standard2.1
- Number Of Executors: 1

Application Configuration

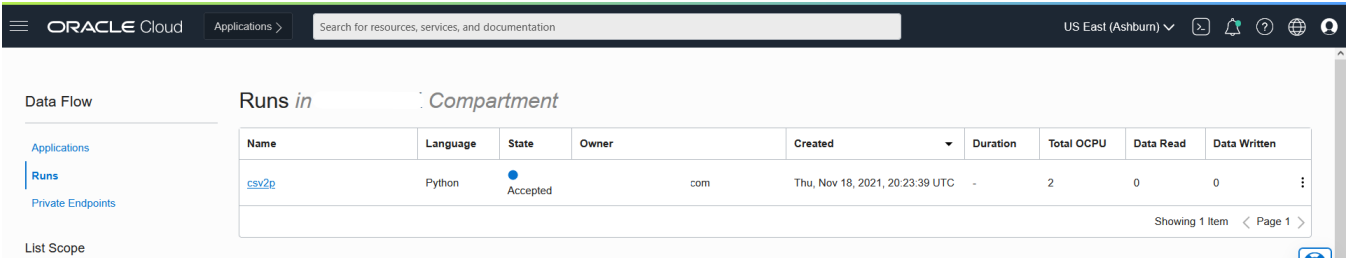
- Language: Python
- OCID: ...g5wxga [Show](#) [Copy](#)
- File URL: ...mr/csv_to_parquet.py [Show](#) [Copy](#)
- Archive URI: No Value
- Arguments: No Value
- Application Log Location: ...w-logs@id3kyspkymr/ [Show](#) [Copy](#)

Para cada execução podemos definir individualmente os parâmetros relacionados a infraestrutura alocada ou argumentos.

Para nossa execução, não será necessário nenhum tipo de alteração, somente clicar em “Run”

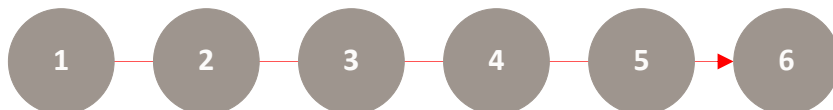


Em seguida você será direcionado a página de Runs, como no imagem abaixo



Lab 6.

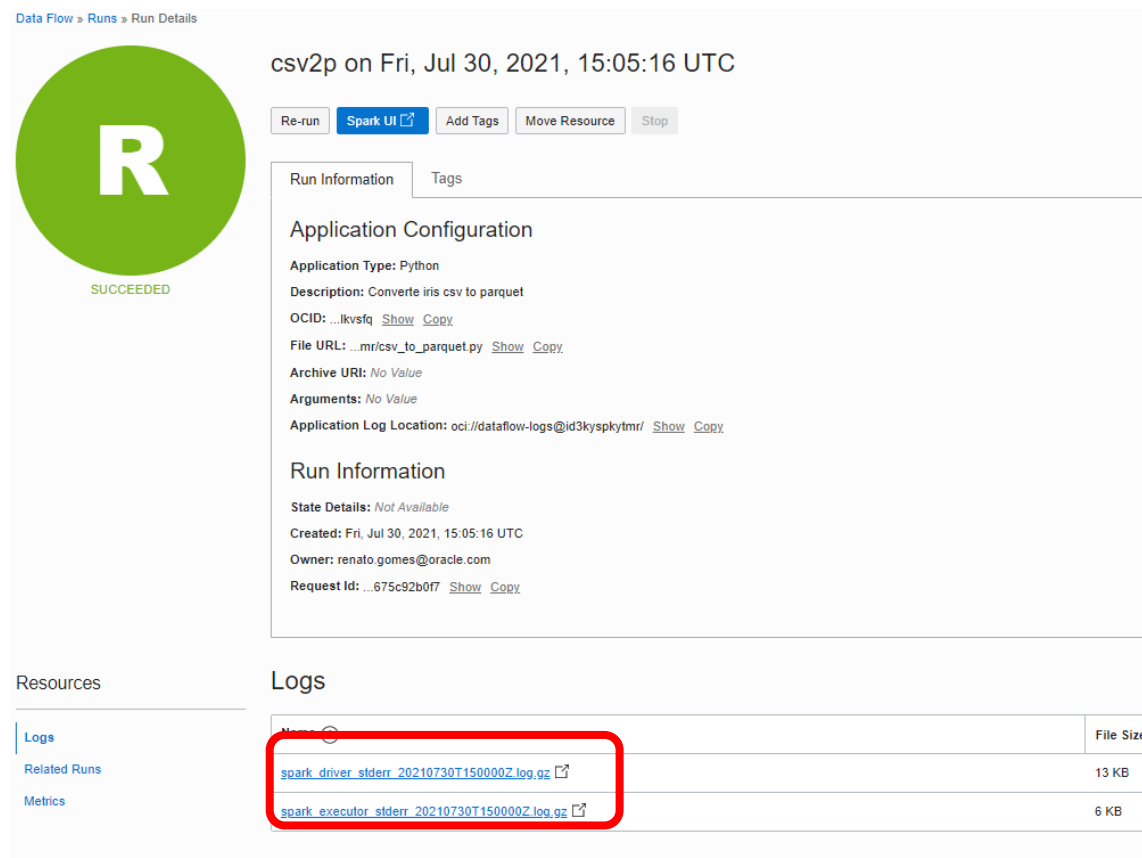
Verificando Logs e o resultado esperado



6. Verificando Logs e o resultado esperado

O dataflow registra automaticamente os logs de erro dos nodes driver e executores no bucket dataflow-logs, os logs de stderr e stdout são gerados a cada execução em dois arquivos .gz diferentes.

As logs ficam listadas a cada “Run” das Applications, porém esta exibição não ocorre de imediato. Caso necessário verificar uma log que ainda não esteja sendo exibida na **console de execução**, podemos também verificar esses log diretamente no **bucket dataflow-logs**.



Data Flow » Runs » Run Details

csv2p on Fri, Jul 30, 2021, 15:05:16 UTC

Re-run Spark UI Add Tags Move Resource Stop

Run Information Tags

Application Configuration

Application Type: Python
Description: Converte iris csv to parquet
OCID: ...lkvsfq [Show](#) [Copy](#)
File URL: ...mr/csv_to_parquet.py [Show](#) [Copy](#)
Archive URI: No Value
Arguments: No Value
Application Log Location: oci://dataflow-logs@id3kyspkytmr/ [Show](#) [Copy](#)

Run Information

State Details: Not Available
Created: Fri, Jul 30, 2021, 15:05:16 UTC
Owner: renato.gomes@oracle.com
Request Id: ...675c92b0f7 [Show](#) [Copy](#)

Resources

Logs

Related Runs

Metrics


Name	File Size
spark_driver_stderr_20210730T150000Z.log.gz	13 KB
spark_executor_stderr_20210730T150000Z.log.gz	6 KB

Após a execução com sucesso, podemos então verificar o bucket **dataflow-out**, e confirmar a geração do arquivo .parquet conforme esperado:

Object Storage » Bucket Details

data-out

Edit Visibility Move Resource Re-encrypt Add Tags Delete



Bucket Information

Tags

Visibility: Private

Namespace: id3kyspkytmr

Default Storage Tier: Standard

Auto-Tiering: Disabled [Edit](#) ⓘ

Approximate Count: 3 objects ⓘ

Etag: a9e1189a-6e1d-4e38-96ba-658755a0f4c4

OCID: ...xhdt4t4g [Show](#) [Copy](#)

Resources

- Objects
- Metrics
- Pre-Authenticated Requests
- Work Requests
- Lifecycle Policy Rules
- Replication Policy
- Retention Rules
- Logs

Objects

Upload More Actions


<input type="checkbox"/>	Name
<input checked="" type="checkbox"/>	iris.parquet
<input type="checkbox"/>	._SUCCESS
<input type="checkbox"/>	part-00000-b47f7f41d-2142-4f0b-add5-19f54495d3e2-c000.snappy.parquet

Você também pode verificar os logs direto no paginá de runs, onde você poderá fazer o download como indicado na imagem abaixo:

ORACLE Cloud Applications Search for resources, services, and documentation US East (Ashburn)

csv2p on Thu, Nov 18, 2021, 20:23:39 UTC

Re-run Spark UI Add Tags Move Resource Stop



SUCCEEDED

Run Information

Tags

Application Configuration

Language: Python

Description: Converte iris.csv para parquet

OCID: ... Show Copy

File URL: ... Show Copy

Archive URI: No Value

Metastore: No Value

Default Managed Table Location: No Value

Arguments: No Value

Application Log Location: oci://dataflow-logs@... Show Copy

Run Information

State Details: Not Available

Created: Thu, Nov 18, 2021, 20:23:39 UTC

Owner: ...

Request Id: ...b6131e76a7 Show Copy

Resource Configuration

Spark Version: 3.0.2

Scala Version: Scala 2.12

Driver Shape: VM Standard2.1

Executor Shape: VM Standard2.1

Number of Executors: 1

Network

Access Type: Internet Access (No Subnet)

Resources

- Logs
- Related Runs
- Metrics

Logs

Archive Logs

Name ⓘ	File Size	Source	Type	Created	
spark_application_stdout.log.gz	20 bytes	APPLICATION	STDOUT	Thu, Nov 18, 2021, 20:29:36 UTC	Open Log File Download
spark_application_stderr.log.gz	20 bytes	APPLICATION	STDERR	Thu, Nov 18, 2021, 20:29:36 UTC	Open Log File Download