



PI Data Engineer Challenge

Aclaración inicial

- Para la resolución del caso se puede utilizar cualquier lenguaje de programación, herramienta o servicio que considere necesario y pertinente. Nos gustaría ver tu código o implementación.
- Esto NO es un examen de ingreso a la empresa, la resolución de este caso nos permite conocer tus habilidades de una forma más práctica.

El problema

Un proceso ETL toma datos de un archivo y lo deposita en la tabla dbo.Unificado.

Por algún error en estos archivos, aparecieron registros duplicados en la tabla.

Consultando con el cliente, nos cuenta que es posible que estas cosas sucedan como consecuencia de errores en el sistema que genera estos archivos, pero que siempre tomemos el ultimo registro que fue copiado, considerando que un registro será duplicado si los campos [ID], [MUESTRA] y [RESULTADO] son iguales en dos filas distintas.

Consignas

- (a) Montar el backup de la base de datos (SQL Server - .bak)
- (b) Descargar programáticamente el archivo csv con el link de mas abajo. Hacelo teniendo en cuenta que este archivo cambia semana a semana con datos nuevos para integrar a la base de datos.
- (c) Desarrollar un proceso que inserte las filas del archivo .csv en la tabla Unificado. Tener en cuenta que la columna FECHA_COPIA esta vacía en el archivo, y hay que agregarle la fecha en la cual estas insertando las nuevas filas a la base de datos.
- (d) Testearlo y verificar que no haya perdida de información. Documentar.
- (e) Dejar de algún modo programado ese proceso para que se ejecute los lunes de cada semana, a las 5:00 AM.
- (f) Mejorar el proceso para que guarde logs con la información que crea necesaria (cantidad de filas afectadas, fecha del proceso, instancia de base de datos, etc).



PI DATA STRATEGY & CONSULTING

Restricciones

- No se dispone del SQL Agent para hacer el scheduling, hay que buscar otra forma.
- La implementación de la base de datos no soporta cursores.

Link al archivo csv

https://gen2cluster.blob.core.windows.net/challenge/csv/nuevas_filas.csv?sv=2019-12-12&ss=bf&srt=co&sp=rwdlacx&se=2020-10-12T21:37:11Z&st=2020-10-01T13:37:11Z&spr=https&sig=JleilT6Tnh3XTPkaucXdzG7rnKX5gjeLEqwG2BGQeo%3D

Presentación de resultados

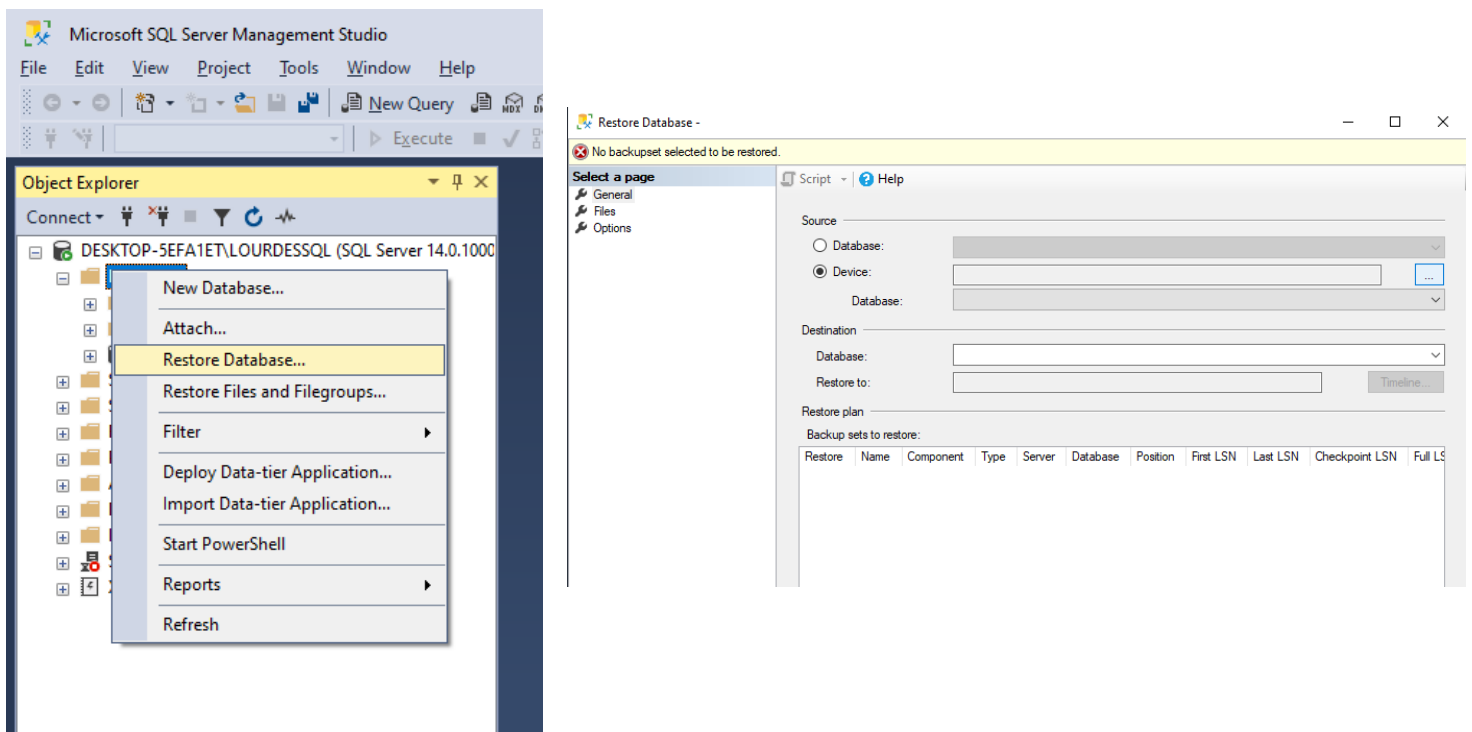
- El entregable principal es un Informe con tus respuestas: explicación de la solución y scripts utilizados.
- Queremos conocer detalles como los criterios utilizados, dificultades encontradas en el camino y resultados parciales.
- ¿Cuántas horas pensaste que te llevaría? ¿Cuántas te llevo realmente?
- ¿Qué sitios web de consulta utilizaste como ayuda?
- Tendrás una presentación de 30 minutos para mostrar los resultados.
- Esperamos recibir tu trabajo en alrededor de 1 semana posterior al envío del challenge.

Si tenés alguna duda, contactanos en recursos.humanos@piconsulting.com.ar

Resolución:

CARGAR EL BACKUP (archivo .bak) A UNA BD

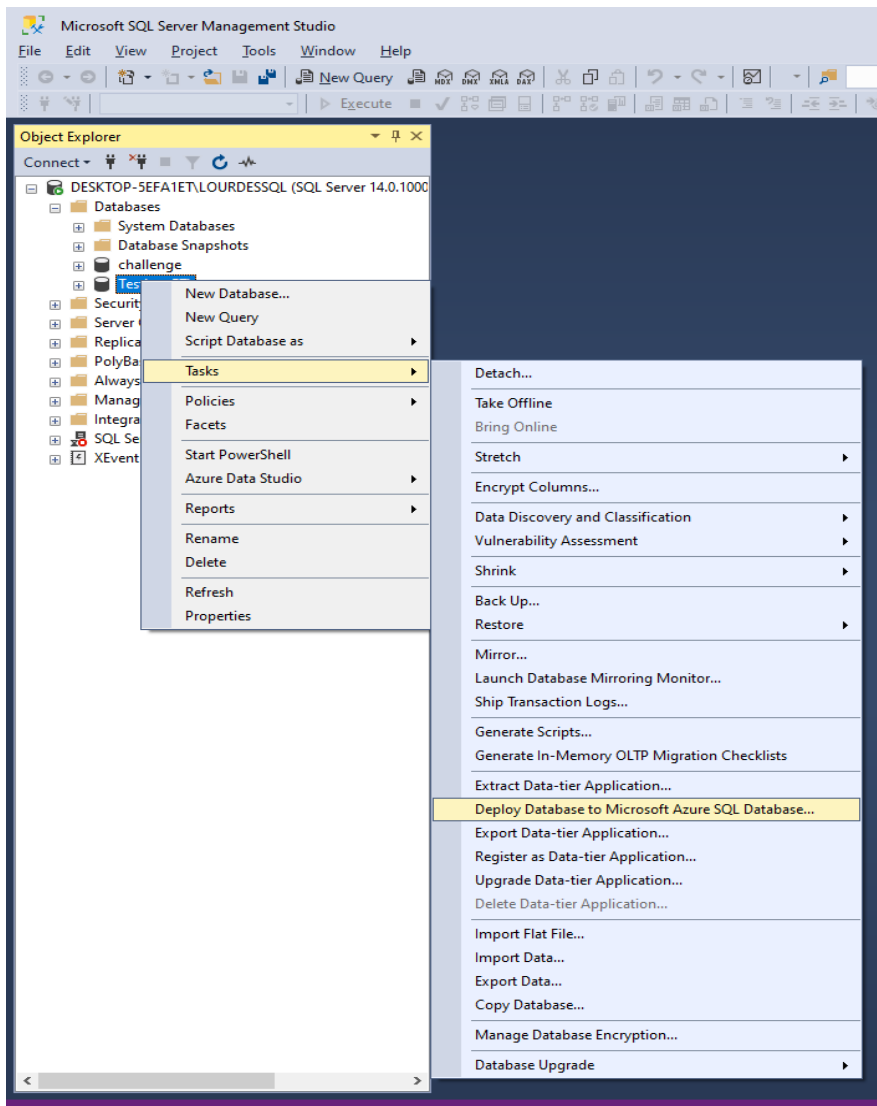
- 1) Entrar a SQL Server
- 2) Click derecho en Databases → Restore database
- 3) Seleccione la fuente de donde restauraré mi base de datos, en este caso Device
- 4) Seleccione el archivo .bak (tener en cuenta que se debo copiar el archivo .bak en la carpeta MSSQL, que la voy a encontrar en Disco C → archivos de programa → Microsoft SQL Server → MSQL11... → MSSQL)
- 5) Aceptar



MIGRAR MI BD DESDE SQL SERVER A AZURE:


1)

- Hacer clic derecho en la base de datos que deseo migrar
- Tasks
- Deploy Database to Microsoft Azure SQL Database



2)

- Next
- Connect, y elijo la BD de destino
- Next

 Help

Specify Target Connection

Specify the name of the instance of SQL Server or the Microsoft Azure SQL Database server that will host the deployed database, name the new database, and then click Connect to login to the target server.

Server connection:

Connect...

New database name:

Microsoft Azure SQL Database settings

Edition of Microsoft Azure SQL Database:

Basic ▼

Maximum database size (GB):

2 ▼

DATABRICKS

En mi notebook voy a pegar el siguiente código en SCALA para vincular el cluster a Data Lake, y además para montarlo como una unidad local.

```
Cmd 1
1  val appID = "ab583e87-5ccb-417d-872b-";
2  val password = "Zs62auQdNS.T~o~dhcx_";
3  val tenantID = "ca1d08b0-9543-4bd9-7786";
4  val fileName = "challenge";
5  val storageAccountName = "storagechallenge";
6
7  val configs = Map("fs.azure.account.auth.type" -> "OAuth",
8                    "fs.azure.account.oauth.provider.type" -> "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
9                    "fs.azure.account.oauth2.client.id" -> appID,
10                   "fs.azure.account.oauth2.client.secret" -> password,
11                   "fs.azure.account.oauth2.client.endpoint" -> ("https://login.microsoftonline.com/" + tenantID + "/oauth2/token"),
12                   "fs.azure.createRemoteFileSystemDuringInitialization" -> "true")
13
14  dbutils.fs.mount(
15    source = "abfss://" + fileName + "@" + storageAccountName + ".dfs.core.windows.net/",
16    mountPoint = "/mnt/data",
17    extraConfigs = configs)
```

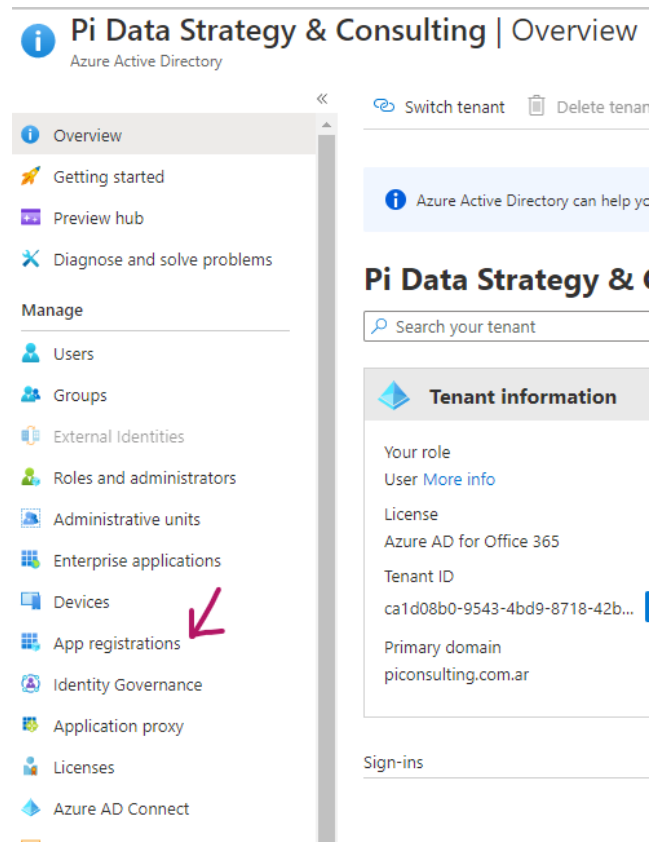
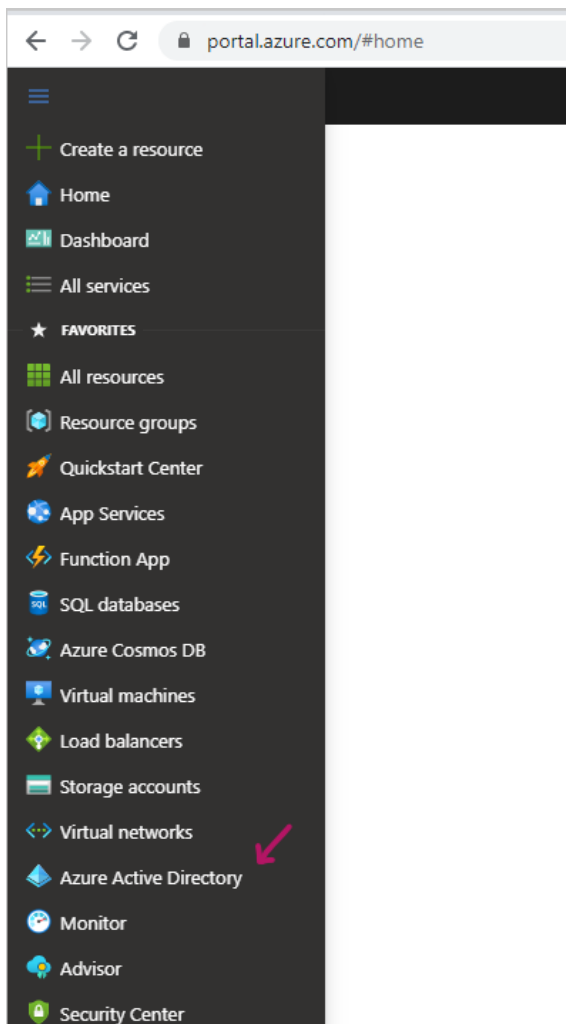
```
val appID = ""
val password = ""
val tenantID = ""
val fileName = "challenge";
val storageAccountName = "storagechallenge";
```

```
val configs = Map("fs.azure.account.auth.type" -> "OAuth",
                  "fs.azure.account.oauth.provider.type" -> "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
                  "fs.azure.account.oauth2.client.id" -> appID,
                  "fs.azure.account.oauth2.client.secret" -> password,
                  "fs.azure.account.oauth2.client.endpoint" -> ("https://login.microsoftonline.com/" + tenantID + "/oauth2/token"),
                  "fs.azure.createRemoteFileSystemDuringInitialization" -> "true")
```

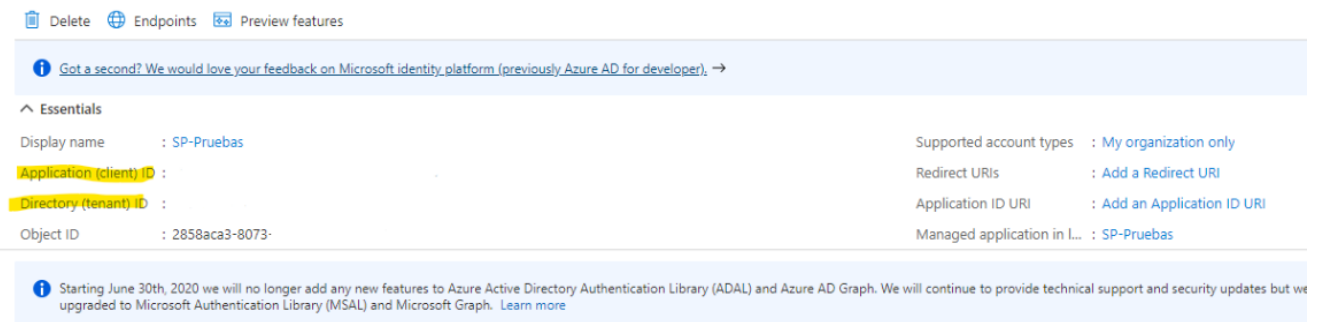
```
dbutils.fs.mount(
  source = "abfss://" + fileName + "@" + storageAccountName + ".dfs.core.windows.net/",
  mountPoint = "/mnt/data",
  extraConfigs = configs)
```

- Para obtener el appID y tenantID:

→ Azure Portal
→ Azure Active Directory
→ App Registrations



En mi caso yo ya tengo una creada, pero en el caso de que no, crearla haciendo click en New Registration (solo hay que asignarle un nombre y luego registrarla). Entro a la app y obtengo los id de los datos resaltados.



- Para obtener password:

- Certificates and Secrets
- New Client Secret
- Agregar una descripción y seleccionar el tiempo de expiración
- Add
- Copiar la contraseña del campo Value

Manage

- Branding
- Authentication
- Certificates & secrets**
- Token configuration
- API permissions
- Expose an API
- Owners
- Roles and administrators | Preview
- Manifest

Support + Troubleshooting

- Troubleshooting

Certificates

Certificates can be used as secrets to prove the application's identity.

[Upload certificate](#)

Thumbprint

No certificates have been added for this application.

Client secrets

A secret string that the application uses to prove its identity

[+ New client secret](#)

Description

- En fileName y storageAccountName asigno los nombres correspondientes

storagechallenge | Containers

Storage account

Search (Ctrl+/)

Overview

Activity log

Tags

Diagnose and solve problems

Container

Change access level

Search containers by prefix

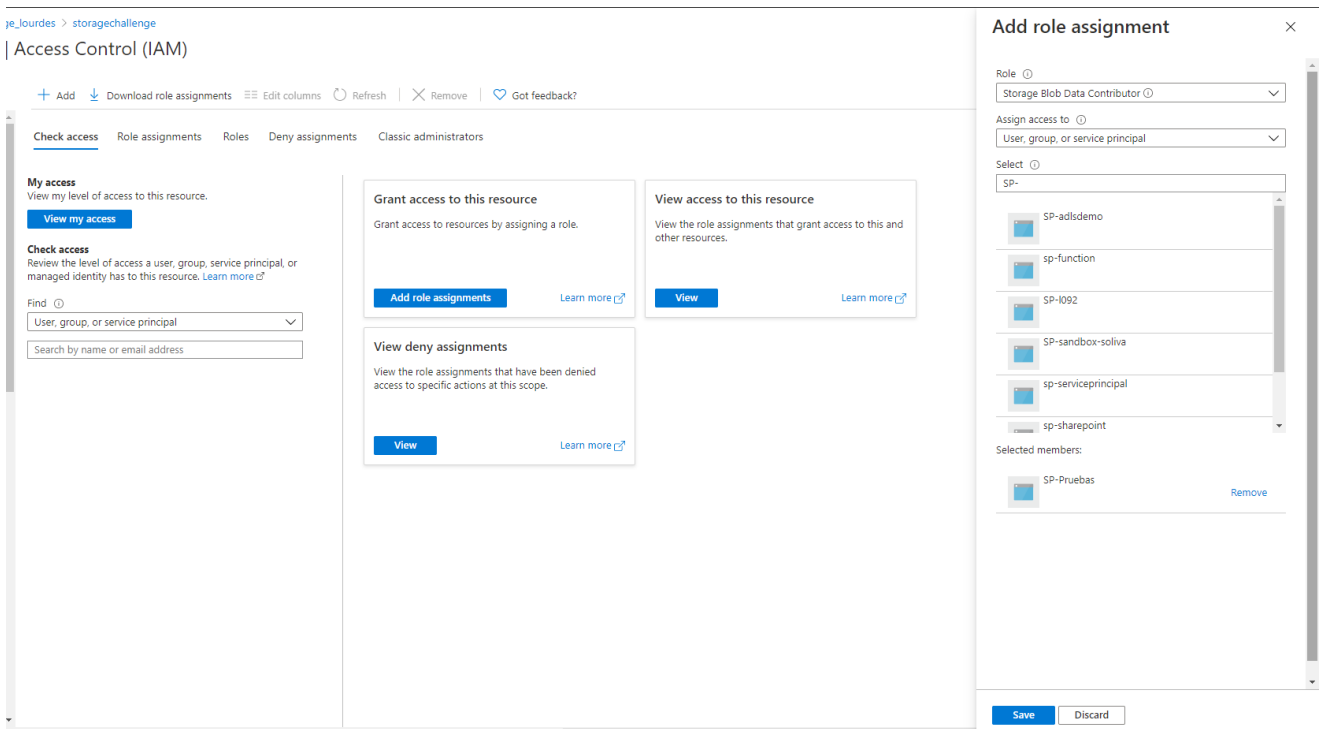
Name

☐ challenge

Ahora, si ejecuto este código me va a dar error, esto se debe a que nunca le dimos a la app permiso de acceso al data lake:

- Dentro del storage account, ir a Access Control
- Add
- Add Role Assignment
- Elegir Storage Blob data contributor
- Elegir mi aplicación (esto es para darle a databricks los permisos de acceso al storage)
- Save

Debo esperar de 1 a 5 minutos antes de volver a correr mi código para que la configuración se propague.



Ahora puedo empezar a desarrollar mi código para realizar las consignas del challenge. Para estoy voy a proponer 3 formas posibles:

- Solución en Spark (DataBricks)
- Solución en Pandas (DataBricks)
- Solución en SQL (SQL Server)

Para cada una de mis soluciones en DataBricks voy a crear un nuevo notebook.

PANDAS:

Cmd 1

```
1 import pandas as pd
2 import datetime as dt
```

Command took 0.01 seconds -- by ldelaorden@piconsulting.com.ar at 3/11/2020 12:23:12 on challenge

Cmd 2

Traigo mi archivo CSV

Cmd 3

```
1 pd_df = pd.read_csv('/dbfs/mnt/data/RAW/nuevas_filas.csv')
2 pd_df.head()
```

Cmd 4

Elimino las filas duplicadas

Cmd 5

```
1 noduplicates = pd_df.drop_duplicates(subset=['ID', 'MUESTRA', 'RESULTADO'], keep="first")
2 # noduplicates.head()
```

Command took 0.01 seconds -- by ldelaorden@piconsulting.com.ar at 3/11/2020 12:31:35 on challenge

Cmd 6

Agrego la fecha y hora actual a la columna FECHA_COPIA

Cmd 7

```
1 datetime = dt.datetime.today().strftime("%d/%m/%Y %I:%M:%S")
2
3 pd_noduplicates = noduplicates.assign(FECHA_COPIA=datetime)
4 pd_noduplicates.head()
```

Cmd 8

Traigo mi CSV unificado

Cmd 9

```
1 pd_dfU = pd.read_csv('/dbfs/mnt/data/TRANSIENT/dbo.Unificado.csv')
2 pd_dfU.head()
```

Cmd 10

Uno mis dos archivos CSV

Cmd 11

```
1 pd_unificado = pd.concat([pd_dfU, pd_noduplicates])
2 pd_unificado.head()
```

Cmd 12

Ordeno mi DataFrame

Cmd 13

```
1 pd_unificado0r = pd_unificado.sort_values(by='FECHA_COPIA')
2 pd_unificado0r.head()
```

Cmd 14

Elimino las filas duplicadas

Cmd 15

```
1 clean_unificado = pd_unificado0r.drop_duplicates(subset=['ID', 'MUESTRA', 'RESULTADO'], keep="first")
2 clean_unificado.head()
```

Cmd 18

Guardo mi dataframe en un archivo CSV en el blobstorage

Cmd 19

```
1 date = dt.datetime.today().strftime("%d-%m-%Y")
2
3 outname = 'clean_Unificado_'+str(date)+'.csv'
4 outdir = '/dbfs/mnt/data/TRANSIENT/'
5
6 clean_unificado.to_csv(outdir+outname, index=False, header=True, encoding="utf-8")
```

Command took 0.29 seconds -- by ldelaorden@piconsulting.com.ar at 3/11/2020 12:52:03 on challenge

SPARK:

Cmd 1

Traigo mi archivo CSV

Cmd 2

```
1 df = spark.read.option("header", "true").csv("/mnt/data/RAW/nuevas_filas.csv")
2
3 display(df)
```

Cmd 3

Elimino las filas duplicadas

Cmd 4

```
1 df_noduplicates = df.drop_duplicates(subset=['ID', 'MUESTRA', 'RESULTADO'])
2 display(df_noduplicates)
```

Cmd 5

Agrego la fecha y hora actual a la columna FECHA_COPIA

Cmd 6

```
1 from pyspark.sql.functions import *
2 from pyspark.sql.functions import current_timestamp
3
4 df = df.withColumn('FECHA_COPIA', when(df.FECHA_COPIA.isNull(), lit(current_timestamp())).otherwise(current_timestamp()))
5 display(df)
```

Cmd 7

Voy a traer mi CSV unificado

Cmd 8

```
1 dfU = spark.read.option("header", "true").csv("/mnt/data/TRANSIENT/dbo.Unificado.csv")
2
3 display(dfU)
```

Cmd 9

Uno mis dos archivos CSV

Cmd 10

```
1 union_dfs = df.union(dfU)
2 # display(union_dfs)
```

Cmd 11

Ordeno mi DataFrame

Cmd 12

```
1 union_dfs.orderBy("FECHA_COPIA", ascending=False)
2 display(union_dfs)
```

Cmd 13

Elimino las filas duplicadas

Cmd 14

```
1 clean_df = union_dfs.drop_duplicates(subset=['ID', 'MUESTRA', 'RESULTADO'])
2 # display(clean_df)
```

Cmd 17



Guardo mi dataframe en un archivo CSV en el blobstorage

Cmd 18

```
1 from datetime import datetime
2 from pyspark.sql.functions import current_date
3 from datetime import date
4
5 outname = 'clean_Unificado_'+str(datetime.now())+'.csv'
6 outdir = '/mnt/data/TRANSIENT/'
7
8 clean_df.coalesce(1).write.csv(outdir+outname, header="true")
```

SQL:

- Para ejecutar mi código SQL en Data Factory voy a crear un Stored Procedure

```
CREATE PROCEDURE procedure_name
AS

UPDATE dbo.newData
SET FECHA_COPIA = GETDATE()

INSERT INTO dbo.Unificado
SELECT * FROM dbo.newData

;WITH masDeUna AS (
SELECT ID, MUESTRA, RESULTADO, MAX(FECHA_COPIA) ultima_fecha
FROM dbo.Unificado
GROUP BY ID, MUESTRA, RESULTADO
HAVING COUNT(*) > 1),

masReciente AS (
SELECT uni.*
FROM dbo.Unificado uni
JOIN masDeUna
ON masDeUna.ID = uni.ID AND masDeUna.MUESTRA = uni.MUESTRA AND masDeUna.RESULTADO = uni.RESULTADO AND masDeUna.ultima_fecha = uni.FECHA_COPIA)

DELETE uni
FROM dbo.Unificado uni
JOIN masReciente
ON masReciente.ID = uni.ID AND masReciente.MUESTRA = uni.MUESTRA AND masReciente.RESULTADO = uni.RESULTADO
WHERE uni.FECHA_COPIA != masReciente.FECHA_COPIA

GO;
```

```
CREATE PROCEDURE procedure_name
AS

UPDATE dbo.newData
SET FECHA_COPIA = GETDATE()

INSERT INTO dbo.Unificado
SELECT * FROM dbo.newData

;WITH masDeUna AS (
SELECT ID, MUESTRA, RESULTADO, MAX(FECHA_COPIA) ultima_fecha
FROM dbo.Unificado
GROUP BY ID, MUESTRA, RESULTADO
HAVING COUNT(*) > 1),

masReciente AS (
SELECT uni.*
FROM dbo.Unificado uni
JOIN masDeUna
ON masDeUna.ID = uni.ID AND masDeUna.MUESTRA = uni.MUESTRA AND masDeUna.RESULTADO =
uni.RESULTADO AND masDeUna.ultima_fecha = uni.FECHA_COPIA)

DELETE uni
FROM dbo.Unificado uni
JOIN masReciente
ON masReciente.ID = uni.ID AND masReciente.MUESTRA = uni.MUESTRA AND masReciente.RESULTADO =
uni.RESULTADO
WHERE uni.FECHA_COPIA != masReciente.FECHA_COPIA

GO;
```

- Por último, antes de empezar a trabajar con DF, voy a crear un notebook en Databricks para descargar mi archivo CSV de la url dada

DESTINATION_PATH : FILE_NAME : SOURCE_URL :

Cmd 1

```
1 dbutils.widgets.removeAll()
2
3
4 dbutils.widgets.text("SOURCE_URL","")
5 csv_url = dbutils.widgets.get("SOURCE_URL")
6
7 dbutils.widgets.text("DESTINATION_PATH","")
8 outdir = dbutils.widgets.get("DESTINATION_PATH")
9
10 dbutils.widgets.text("FILE_NAME","")
11 outname = dbutils.widgets.get("FILE_NAME")
12
13 # outname = newData.csv
14 # outdir = /dbfs/mnt/data/RAW/newData.csv
15 # csv_url = https://gen2cluster.blob.core.windows.net/challenge/csv/nuevas_filas.csv?sv=2019-12-12&ss=bfqt&srt=sco&sp=rwdlacupx&se=2020-11-20T22:16:16Z&st=2020-11-10T14:16:16Z&spr=https&sig=QaPsF78hVQu0WYR82nZeYkEI2teJDcQ2vewXy7RiLV0%3D
```

Command took 0.02 seconds -- by ldeLaorden@piconsulting.com.ar at 14/11/2020 19:09:39 on challenge

Cmd 2

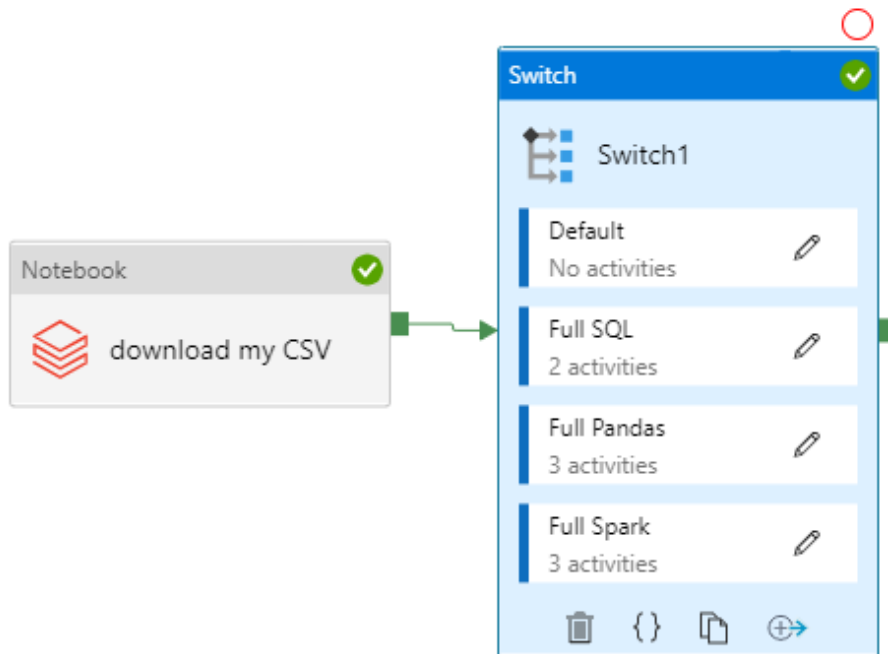
Descargo mi CSV y lo guardo

Cmd 3

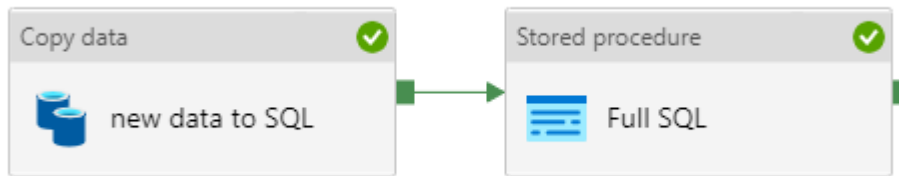
```
1 import pandas as pd
2
3 data = pd.read_csv(csv_url)
4
5 data.to_csv(outdir+outname, index=False, header=True, encoding="utf-8")
```

DATA FACTORY:

- Para la estructura principal del pipeline voy a tener dos activities, la primera va a ser de Databricks para descargar mi archivo CSV, y la segunda una condición switch donde cada uno de los case será una forma diferente de resolver el problema (con Pandas, Spark o SQL).



- Dentro de Full SQL



- Dentro de Full Pandas



- Dentro de Full Spark

