Importamos librerias necesarias

In [44]:
```
import pyspark
from pyspark.sql import SparkSession
from pyspark.sql.types import *
sc = pyspark.SparkContext('local[*]')
```

Creamos nuestra sesión de spark

In [45]:
```
spark = SparkSession.builder.appName('test').getOrCreate()
```

In [46]:
```
spark
```

Out[46]:

SparkSession - in-memory

SparkContext

Spark UI

Version	`v3.3.0`
Master	`local[*]`
AppName	`test`

Ejercicio 7

Origen: Cargar información de productos al metastore: product: /public/retail_db/products/part-00000

Resultado: Almacenar en Metastore table products

Definimos el esquema de la tabla productos

In [ ]:
```
ProductSchema = StructType([
   StructField("product_id", IntegerType(), True),
   StructField("product_category_id", IntegerType(), True),
   StructField("product_name", StringType(), True),
   StructField("product_description", StringType(), True),
   StructField("product_price", FloatType(), True),
   StructField("product_image", StringType(), True)
   ]
)
```

Leemos el archivo con el esquema anterior

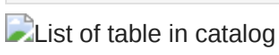
In [ ]:
```
products_df = (
    spark
    .read
    .format('csv')
    .options(inferSchema=True)
    .schema(ProductSchema)
    .load('/public/retail_db/products/part-00000')
)
```

escribimos en el Metastore

In [ ]:
```
products_df.write.format('hive').saveAsTable('products')
```

Listamos las tablas para validar

In [ ]:
```
spark.catalog.listTables()
```


List of table in catalog

Ejercicio 7

Cargar información de productos al metastore: product: /public/retail_db/products/part-00000

Resultado: Almacenar en Metastore table products

Ejercicio 8

Agrupar cantidad de transacciones por mes utilizando el metastore: orders: /public/retail_db/orders/part-00000

Resultado: Almacenar el resultado, en formato parquet sin comprimir en la ruta: /user/vagrant/lab1/pregunta8/resultado

El archivo deberá ser almacenado con el esquema Count, month (format YYYYMM)

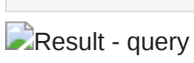
In [ ]:
```
orders_schema = StructType([
   StructField("order_id", IntegerType(), True),
   StructField("order_date", DateType(), True),
   StructField("order_customer_id", IntegerType(), True),
   StructField("order_status", StringType(), True)
])
```

In [48]:
```
orders_df = (
    spark.read
    .format('csv')
    .options(inferSchema=True)
    .schema(orders_schema)
    .load('public/retail_db/orders/part-00000')
)
```

In [ ]:
```
orders_df.write.format('hive').saveAsTable('orders')
```

In [50]:
```
query = """ SELECT COUNT(*) AS count, DATE_FORMAT(order_date, 'YYYYMM') AS month FROM orders GROUP BY DATE_FORMAT(order_date, 'YYYYMM')"""
```

In [51]:
```
result = spark.sql(query)
```


Result - query

In [ ]:
```
result.write.option('compression', 'uncompressed').format('parquet').save('/user/vagrant/lab1/pregunta8/resultado')
```