**Traits Auxiliares  
se creó Traits de Session para la creación de la SparkSession y Trait Time para la manipulación de fechas  
estas se encuentran en   
 ./DesafioNubimetrics/src/Session**

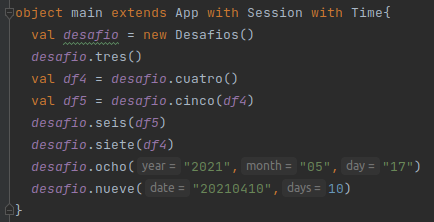
**./DesafioNubimetrics/src/Time**

**Todos los desafíos fueron colocados en una clase Desafios que posee métodos por cada ejercicio desarrollado.**

**esta se encuentra en**

**./DesafioNubimetrics/src/Desafios**

**La ejecuccion de todos los ejercicios se puede realizar desde el objeto main.**

****

**Desafío 1**

**No se envio el mail**

**Desafío 2**

**Se realizo un request a la API de ML desde el postman y se coloco el archivo en ./Desafio2/MLA1000json202202.json**

**Desafío 3**

1. **cargamos el archivo JSON Sellers.json en un DataFrame**
2. **seleccionamos solo las columnas id,nickname,points**
3. **establecemos el path donde se crearan las archivos**
4. **renombramos las columnas**
5. **escribimos 3 select distintos para cada uno de los casos y se escriben estos en archivos csv**

****

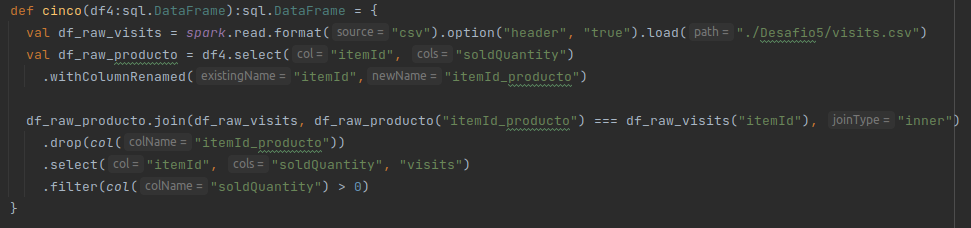
**Desafío 4**

1. **cargamos el archivo JSON MPE1004.json en un DataFrame**
2. **seleccionamos las 3 columnas que necesitamos y creamos una nueva “allinone” con la estructura de un array que incluye los valores de esas 3**
3. **tiramos las 3 columnas originales porque ahora tenemos todo en la columna “allinone”**
4. **explotamos el array**
5. **agregamos la columna rowId**
6. **renombramos las columnas**
7. **el método devuelve un DataFrame**

****

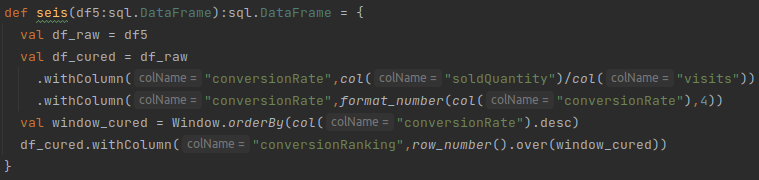
**Desafío 5**

1. **ingresamos como parámetro el DataFrame del desafío 4**
2. **cargamos el csv visits.csv**
3. **realizamos un INNER JOIN de ambos archivos y solo nos quedamos con los que cumplan la condición que tenga ventas mayores a 0**

****

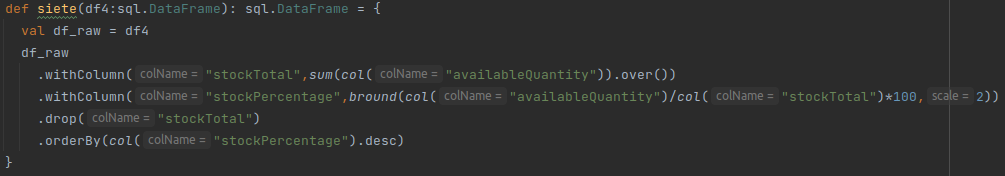
**Desafío 6**

1. **ingresamos como parámetro el DataFrame del ejercicio 5**
2. **agregamos la columna conversionRate construida del cálculo de ventas/visitas**
3. **para el ranking usamos una función de agregación row\_number() con una ventana de la ordenación descendente de la conversionRate creando numeros del 1 al X en el orden marcado.**

****

**Desafío 7**

1. **ingresamos como parámetro el DataFrame del ejercicio 4**
2. **agregamos una columna con el stockTotal sumando todas las cantidades disponibles**
3. **dividimos las cantidades disponibles de cada file por el stockTotal y lo multiplicamos por 100 para obtener el stockPercentage**
4. **ordenamos de manera descendente el stockPercentage**

****

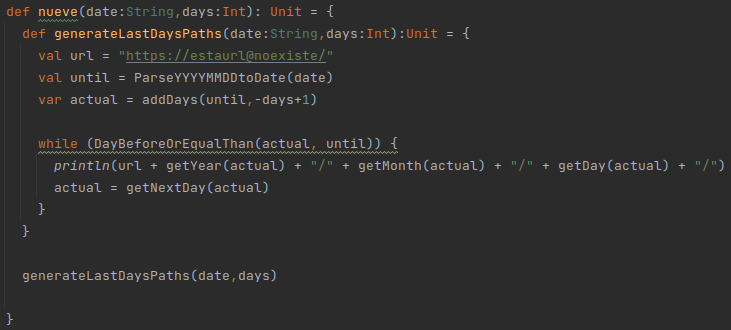
**Desafío 8**

1. **el método establece una url**
2. **obtenemos la fecha actual**
3. **obtenemos la fecha hasta (until)**
4. **mientras que la fecha “actual” no sea igual o menor a la “until” se imprime la url con las fechas, en este bucle le vamos sumando un dia a la fecha actual.**

****

**Desafío 9**

1. **el método establece una url**
2. **obtenemos la fecha hasta como parámetro (until) y la convertimos del formato YYYYMMDD al YYYY-MM-DD para que coincida con el tipo date de LocalDate**
3. **la fecha actual (de inicio) será la fecha until menos la cantidad de días pasados por parametros+1**
4. **mientras que la fecha “actual” no sea igual o menor a la “until” se imprime la url con las fechas, en este bucle le vamos sumando un día a la fecha actual.**

****

**Desafío 10**

**Se instaló el servicio de mssql en ubuntu, se restauró la base datos con el archivo AdventureWorks2014.bak luego nos conectamos a la base con la herramienta Dbeaver.**

**Desafío 11**

**SELECT**

**sp.CountryRegionCode country\_region\_code,**

**cast(AVG(str.TaxRate) as float) average\_taxRate**

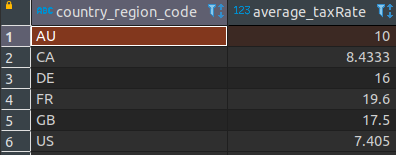
**FROM [Sales].[SalesTaxRate] str**

**INNER JOIN [Person].[StateProvince] sp**

**ON str.StateProvinceID = sp.StateProvinceID**

**GROUP BY sp.CountryRegionCode**

**;**

****

**Desafío 12**

**with currencies as (**

**SELECT**

**cor.Name country\_name**

**,cor.CountryRegionCode**

**,c.Name currency\_name**

**,c.CurrencyCode currency\_code**

**FROM [Person].[CountryRegion] cor**

**INNER JOIN [Sales].[CountryRegionCurrency] crc**

**ON cor.CountryRegionCode = crc.CountryRegionCode**

**INNER JOIN [Sales].[Currency] c**

**ON crc.CurrencyCode = c.CurrencyCode**

**INNER JOIN [Sales].[CurrencyRate] cr**

**ON crc.CurrencyCode = cr.ToCurrencyCode**

**)**

**,taxes as (**

**SELECT**

**cor.Name country\_name,**

**cor.CountryRegionCode,**

**str.TaxRate**

**FROM [Person].[CountryRegion] cor**

**INNER JOIN [Person].[StateProvince] sp**

**ON cor.CountryRegionCode = sp.CountryRegionCode**

**INNER JOIN [Sales].[SalesTaxRate] str**

**ON sp.StateProvinceID = str.StateProvinceID**

**)**

**SELECT**

**c.country\_name,**

**c.currency\_name,**

**cast(AVG(c.AverageRate) as numeric(10,2)) currency\_rate,**

**cast(AVG(t.TaxRate) as numeric(10,2)) average\_tax\_rate**

**FROM currencies c**

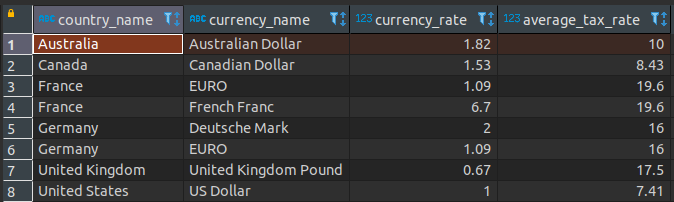
**INNER JOIN taxes t**

**ON c.CountryRegionCode = t.CountryRegionCode**

**GROUP BY c.country\_name,c.currency\_name**

**ORDER BY country\_name asc**

**;**

****