Actividad modulo 48 - Big Data II - PyArrow

○ Exportación de datos desde Spark

In [ ]:
```
from pyarrow import csv
import pyarrow as pa
```

○ Conexión desde Python al servicio Spark donde se instaló la información

In [ ]:
```
archivo = "D:/WORK IN PROGRESS/Data Analytics course/Archivos ejercicios/Housing.csv"
tab_housing = csv.read_csv(archivo)
```

In [ ]:
```
# Estructura de Housing visto desde PyArrow
tab_housing
```

Out[ ]:
```
pyarrow.Table
price: int64
area: int64
bedrooms: int64
bathrooms: int64
stories: int64
mainroad: string
guestroom: string
basement: string
hotwaterheating: string
airconditioning: string
parking: int64
prefarea: string
furnishingstatus: string
----
price: [[13300000,12250000,12250000,12215000,11410000,...,1820000,1767150,1750000,1750000,1750000]]
area: [[7420,8960,9960,7500,7420,...,3000,2400,3620,2910,3850]]
bedrooms: [[4,4,3,4,4,...,2,3,2,3,3]]
bathrooms: [[2,4,2,2,1,...,1,1,1,1,1]]
stories: [[3,4,2,2,2,...,1,1,1,1,2]]
mainroad: [["yes","yes","yes","yes","yes",...,"yes","no","yes","no","yes"]]
guestroom: [["no","no","no","no","yes",...,"no","no","no","no","no"]]
basement: [["no","no","yes","yes","yes",...,"yes","no","no","no","no"]]
hotwaterheating: [["no","no","no","no","no",...,"no","no","no","no","no"]]
airconditioning: [["yes","yes","no","yes","yes",...,"no","no","no","no","no"]]
...
```

○ Selección de datos de housing con filtros simples:

1) listado completo de columnas ordenado por price

In [ ]:
```
sorted_table = tab_housing.sort_by([('price','descending')])
```

In [ ]:
```
sorted_table
```

Out[ ]:
```
pyarrow.Table
price: int64
area: int64
bedrooms: int64
bathrooms: int64
stories: int64
mainroad: string
guestroom: string
basement: string
hotwaterheating: string
airconditioning: string
parking: int64
prefarea: string
furnishingstatus: string
----
price: [[13300000,12250000,12250000,12215000,11410000,...,1820000,1767150,1750000,1750000,1750000]]
area: [[7420,8960,9960,7500,7420,...,3000,2400,3620,2910,3850]]
bedrooms: [[4,4,3,4,4,...,2,3,2,3,3]]
bathrooms: [[2,4,2,2,1,...,1,1,1,1,1]]
stories: [[3,4,2,2,2,...,1,1,1,1,2]]
mainroad: [["yes","yes","yes","yes","yes",...,"yes","no","yes","no","yes"]]
guestroom: [["no","no","no","no","yes",...,"no","no","no","no","no"]]
basement: [["no","no","yes","yes","yes",...,"yes","no","no","no","no"]]
hotwaterheating: [["no","no","no","no","no",...,"no","no","no","no","no"]]
airconditioning: [["yes","yes","no","yes","yes",...,"no","no","no","no","no"]]
...
```

In [ ]:
```
df1= sorted_table.to_pandas()
df1.head()
```

Out[ ]:

	price	area	bedrooms	bathrooms	stories	mainroad	guestroom	basement	hotwaterheating	airconditioning	parking	prefarea	furnishingstat
0	13300000	7420	4	2	3	yes	no	no	no	yes	2	yes	furnish
1	12250000	8960	4	4	4	yes	no	no	no	yes	3	no	furnish
2	12250000	9960	3	2	2	yes	no	yes	no	no	2	yes	semi-furnish
3	12215000	7500	4	2	2	yes	no	yes	no	yes	3	yes	furnish
4	11410000	7420	4	1	2	yes	yes	yes	no	yes	2	no	furnish

2) para las casas con mayor numero de habitaciones, calcular el promedio de precio, y tamaño en m2

In [ ]:
```
# Agrupa la tabla por numero de habitaciones, para odentificar que valor tiene el maor numero de inmuebles
grouped_table = tab_housing.group_by("bedrooms").aggregate([("bedrooms","count")]).sort_by([('bedrooms_count','descending')])
```

In [ ]:
```
grouped_table
```

Out[ ]:
```
pyarrow.Table
bedrooms_count: int64
bedrooms: int64
----
bedrooms_count: [[300,136,95,10,2,2]]
bedrooms: [[3,2,4,5,6,1]]
```

In [ ]:
```
df2= grouped_table.to_pandas()
df2.head()
```

Out[ ]:

	bedrooms_count	bedrooms
0	300	3
1	136	2
2	95	4
3	10	5
4	2	6

In [ ]:
```
# Se evidencia que la opcion con mayor numero de ocurrencias es 3 camas.
```

In [ ]:
```
import pyarrow.compute as pc

# Assuming you have a table named 'apartment_table' with columns 'price', 'area', and 'bedrooms'
# Filter the table to select only apartments with 3 bedrooms
filtered_table = tab_housing.filter(pc.field('bedrooms')== 3)

grouped_table_2= filtered_table.group_by("bedrooms").aggregate([
   ("area", "mean"),
   ("price", "mean")
])
```

```
grouped_table_2
```

Out[ ]:
```
pyarrow.Table
area_mean: double
price_mean: double
bedrooms: int64
----
area_mean: [[5226.62]]
price_mean: [[4954598.133333334]]
bedrooms: [[3]]
```

In [ ]:
```
df3= grouped_table_2.to_pandas()
df3.head()
```

Out[ ]:

	area_mean	price_mean	bedrooms
0	5226.62	4.954598e+06	3

○ Agrupamiento en PyArrow, por número de habitaciones y baños, del precio. Ej: # habitaciones | # baños | precio promedio, esto por furnishingstatus

In [ ]:
```
grouped_table_3= tab_housing.group_by("furnishingstatus").aggregate([
   ("bedrooms","count"),
   ("bathrooms","count"),
   ("price", "mean")
])
grouped_table_3
```

Out[ ]:
```
pyarrow.Table
bedrooms_count: int64
bathrooms_count: int64
price_mean: double
furnishingstatus: string
----
bedrooms_count: [[140,227,178]]
bathrooms_count: [[140,227,178]]
price_mean: [[5495696,4907524.22907489,4013831.4606741574]]
furnishingstatus: [["furnished","semi-furnished","unfurnished"]]
```

In [ ]:
```
df4= grouped_table_3.to_pandas()
df4.head()
```

Out[ ]:

	bedrooms_count	bathrooms_count	price_mean	furnishingstatus
0	140	140	5.495696e+06	furnished
1	227	227	4.907524e+06	semi-furnished
2	178	178	4.013831e+06	unfurnished