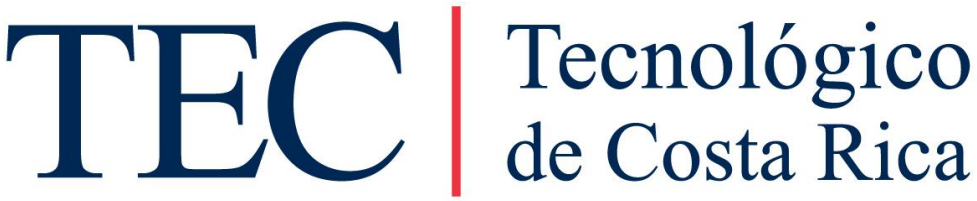
Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación



Proyecto III

Bases de datos II

Grupo 20

Profesor:
Alberto Shum Chan

Alumnos:
Alberto Zumbado Abarca
Jonathan Quesada Salas

2
1
1
0
0
1
1
2
1
1
1
1
1
1
1
1
1
1
1
90.48

Alajuela, noviembre 2021

Descripción general del sistema

Con lo que respecta a la descripción general del sistema, es que se divide en varias secciones, las cuales son las siguientes:

Utilizar únicamente la funcionalidad de Spark en Python: Este punto en cuestión consiste en un conjunto de datos por medio del nombre del distrito. Para que la integración sea exitosa se debe primero procesar los datos de forma que el campo para la reunión o join en ambos conjuntos de datos, en este caso el distrito, coincida (documente muy bien todo el proceso).

Se tienen que guardar los datos limpios del INEC y del OIJ en una base de datos en PostgreSQL

Visualización de datos: Se deben de realizar las siguientes visualizaciones

- Compare la cantidad de delitos y la tasa de ocupación para los 10 distritos con más delitos en el país.
- Grafique la cantidad de delitos por día de la semana para el distrito con más delitos.
- Grafique la cantidad de delitos por tipo y por distrito. Es decir, para el distrito seleccionado se debe graficar la cantidad de delitos por tipo.
- Grafique la cantidad de delitos por sexo para todo el conjunto de datos.
- Proponga una visualización de su interés.

Descripción de las funciones

Parte A:

- **Descripción general:** La función se encargará de eliminar los espacios en blanco de la columna distrito de ambos conjuntos de datos.
- **Descripción de parámetros de entrada:** Los respectivos archivos de extensión .csv
- **Descripción de salida:** Retorna el csv como tal, pero con la diferencia que en la columna “Distrito” se presentan los mismos sin espacios.
- **Descripción de bloques relevantes:** La funcionalidad de “withColumn” se analiza una respectiva columna para luego poder mostrar el resultado con otra funcionalidad llamada .show(). Y adicionalmente la funcionalidad de “delete_spaces” es el que se encarga de borrar los espacios en la columna Distrito

Parte B:

- **Descripción general:** la función se encargará de convertir a minúsculas el contenido de la columna “distrito” de ambos conjuntos de datos.

- **Descripción de parámetros de entrada:** Los respectivos archivos de extensión .csv
- **Descripción de salida:** Retorna el csv como tal, pero con la diferencia que en la columna “Distrito” se presentan sus elementos en letras minúsculas.
- **Descripción de bloques relevantes:** La funcionalidad de “withColumn” se analiza una respectiva columna para luego poder mostrar el resultado con otra funcionalidad llamada .show(). Y adicionalmente la funcionalidad de “lower_case_columnmnes” el que se encarga de convertir las letras a minúscula en la columna Distrito.

Parte E:

- **Descripción general:** En está función se procede a editar los nombres de los cantones del INEC para que puedan coincidir con algunos de los cantones presentados en el OIJ
- **Descripción de parámetros de entrada:** Los respectivos archivos de extensión .csv
- **Descripción de salida:** Retorna los respectivos archivos de extensión .csv y estos mismos se pueden notar coincidencias entre columnas.
- **Descripción de bloques relevantes:** Se realiza la funcionalidad de eliminar las tildes como tal de las palabras que puedan tener las columnas de ambos archivos .csv (normalice), para que posteriormente se puedan parsear los datos de las columnas (string to_float), de esta manera se trabajaría los encabezados de las columnas, pero con lo que respecta a los datos se puedan hacer la conversión de string a flotante.

Documentación de visualizaciones

- **Descripción de la visualización 1:** Primeramente para la realización de está visualización se procede a establecer la sentencia de SQL como entrada respectiva para extraer la comparación de delitos y tasas de ocupación por los 10 distritos con más delitos, para que luego se proceda a juntar ambos data frames con ayuda de un join, para que de está manera se pueda pasar el dataframe de Spark a Panda, ya que es necesario para poder llegar a graficar dicha funcionalidad, para luego con un for se procede a recorrer todos los distritos para añadir continuamente la cantidad de delitos y la tasa de ocupación respectiva para luego añadir el mismo. Adicionalmente ya teniendo el recorrido respectivo se establece el siguiente código fuente para poder crear el gráfico de barras:

```
plt.figure(figsize=(12,6), tight_layout=True)
grafico = sns.barplot(x="Distrito", y="Cantidad", hue="Tipo cantidad",␣
 ↪palette="hls", data=avance)
grafico.set(title=titulo, xlabel="Distrito", ylabel="Cantidad")
plt.show()
```

- **Descripción de la visualización 2:** Primeramente para la resolución de esta visualización es que establecer la sentencia respectiva de SQL como entrada para la cantidad de delitos por día para el distrito con más delitos, para que de esta manera ya teniendo dicha sentencia se pueda obtener el distrito con más delitos, para poder llegar a graficarlos con ayuda de data frames (Pandas). Para poder establecer el gráfico de barras se realizó lo siguiente para poder graficar lo solicitado:

```
plt.figure(figsize=(12,20), tight_layout=True)
grafico = sns.barplot(x="count", y="Fecha", hue="Distrito", palette="hls",␣
 ↪data=grafico1)
grafico.set(title=titulo, xlabel="Cantidad De DElitos", ylabel="Fecha")
plt.show()
```

- **Descripción de la visualización 3:** Primeramente para la resolución de esta visualización es establecer la sentencia SQL como entrada a usar para poder establecer la cantidad de delitos por tipo y distrito escogido, después se trata el data frame para poder desarrollarlo con la libreria de Pandas, con ayuda de la sentencia de “toPandas()” y lo que faltaría sería establecer el gráfico de barras el cual se hace de la siguiente manera:

```
plt.figure(figsize=(20,6), tight_layout=True)
grafico = sns.barplot(x="Delito", y="cantidadDelitos", palette="hls",␣
 ↪data=distrio_tipo)
grafico.set(title=titulo, xlabel="Tipo de delito", ylabel="Cantidad de delitos")
plt.show()
```

- **Descripción de la visualización 4:** Primeramente para la resolución de esta visualización se debe de establecer la sentencia de SQL como entrada a usar para poder llegar a establecer la cantidad de delitos por género, se transforma el data frame para poder con ayuda de “toPandas()”, para que de está manera se pueda usar respectivamente la configuración necesaria para poder usar el gráfico de barras, establecido de la siguiente manera:

```
plt.figure(figsize=(20,6), tight_layout=True)
grafico = sns.barplot(x="cantidadDeDelitos", y="Genero", palette="hls",␣
 ↪data=delitos_por_sexo)
grafico.set(title=titulo, xlabel="Cantidad de delitos", ylabel="Genero")
plt.show()
```

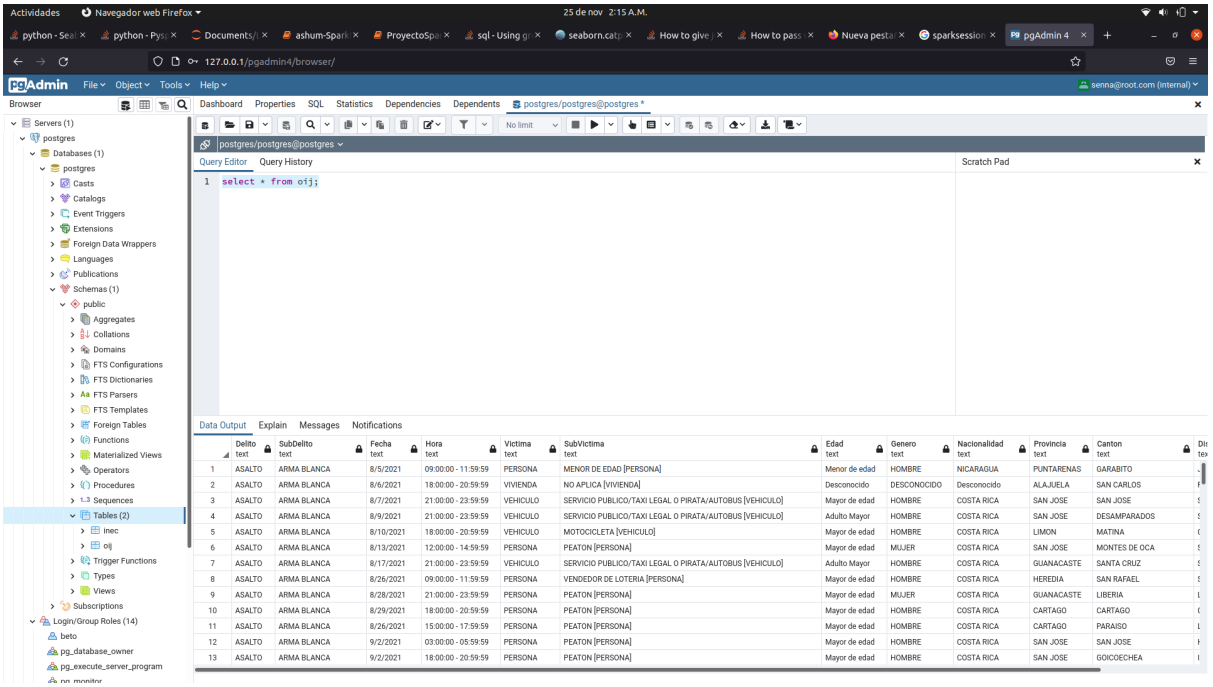
- **Descripción de la visualización 5:** Para esta respectiva visualización se estableció graficar los delitos por edad y el procedimiento como tal es el siguiente, se establece como entrada la sentencia SQL a usar para que de

esta manera se pueda transforma el data frame respectivo con ayuda de la funcionalidad “toPandas()” para que luego de una manera similar se pueda graficar dichos datos en un gráfico de barras con una configuración similar a las ya antes vistas en visualizaciones anteriores, establecida de la siguiente manera:

```
plt.figure(figsize=(20,6), tight_layout=True)
grafico = sns.barplot(x="Edad", y="cantidadDeDelitos", palette="hls",␣
 ↪data=delitos_por_edad)
grafico.set(title=titulo, xlabel="Edad", ylabel="Cantidad de delitos")
plt.show()
```

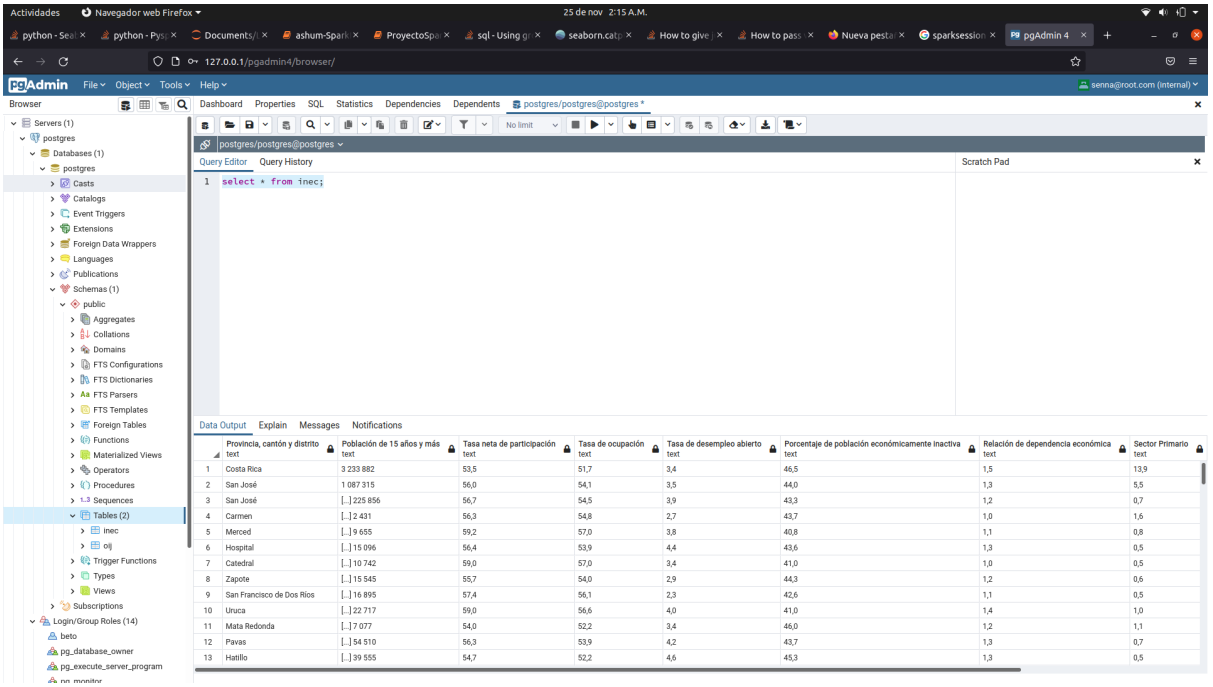
Documentación de guardar los datos limpios del INEC y OIJ en PostgreSQL

A continuación se muestra los datos en limpio del OIJ en PostgreSQL previamente trabajados:



	Delito	SubDelito	Fecha	Hora	Victima	SubVictima	Edad	Genero	Nacionalidad	Provincia	Canton
1	ASALTO	ARMA BLANCA	8/5/2021	09:00:00 - 11:59:59	PERSONA	MENOR DE EDAD [PERSONA]	Menor de edad	HOMBRE	NICARAGUA	PUNTARENAS	GARABITO
2	ASALTO	ARMA BLANCA	8/6/2021	18:00:00 - 20:59:59	VIVIENDA	NO APLICA [VIVIENDA]	Desconocido	DESCONOCIDO	Desconocido	ALAJUELA	SAN CARLOS
3	ASALTO	ARMA BLANCA	8/7/2021	21:00:00 - 23:59:59	VEHICULO	SERVICIO PUBLICO/TAXI LEGAL O PIRATA/AUTOBUS [VEHICULO]	Mayor de edad	HOMBRE	COSTA RICA	SAN JOSE	SAN JOSE
4	ASALTO	ARMA BLANCA	8/9/2021	21:00:00 - 23:59:59	VEHICULO	SERVICIO PUBLICO/TAXI LEGAL O PIRATA/AUTOBUS [VEHICULO]	Adulto Mayor	HOMBRE	COSTA RICA	SAN JOSE	DESAMPARADOS
5	ASALTO	ARMA BLANCA	8/10/2021	18:00:00 - 20:59:59	VEHICULO	MOTOCICLETA [VEHICULO]	Mayor de edad	HOMBRE	COSTA RICA	LIMON	MATINA
6	ASALTO	ARMA BLANCA	8/13/2021	12:00:00 - 14:59:59	PERSONA	PEATON [PERSONA]	Mayor de edad	MUJER	COSTA RICA	SAN JOSE	MONTES DE OCA
7	ASALTO	ARMA BLANCA	8/17/2021	21:00:00 - 23:59:59	VEHICULO	SERVICIO PUBLICO/TAXI LEGAL O PIRATA/AUTOBUS [VEHICULO]	Adulto Mayor	HOMBRE	COSTA RICA	GUANACASTE	SANTA CRUZ
8	ASALTO	ARMA BLANCA	8/26/2021	09:00:00 - 11:59:59	PERSONA	VENDEDOR DE LOTERIA [PERSONA]	Mayor de edad	HOMBRE	COSTA RICA	HEREDIA	SAN RAFAEL
9	ASALTO	ARMA BLANCA	8/28/2021	21:00:00 - 23:59:59	PERSONA	PEATON [PERSONA]	Mayor de edad	MUJER	COSTA RICA	GUANACASTE	LIBERIA
10	ASALTO	ARMA BLANCA	8/29/2021	18:00:00 - 20:59:59	PERSONA	PEATON [PERSONA]	Mayor de edad	HOMBRE	COSTA RICA	CARTAGO	CARTAGO
11	ASALTO	ARMA BLANCA	8/26/2021	15:00:00 - 17:59:59	PERSONA	PEATON [PERSONA]	Mayor de edad	HOMBRE	COSTA RICA	CARTAGO	PARAISO
12	ASALTO	ARMA BLANCA	9/2/2021	03:00:00 - 05:59:59	PERSONA	PEATON [PERSONA]	Mayor de edad	HOMBRE	COSTA RICA	SAN JOSE	SAN JOSE
13	ASALTO	ARMA BLANCA	9/2/2021	18:00:00 - 20:59:59	PERSONA	PEATON [PERSONA]	Mayor de edad	HOMBRE	COSTA RICA	SAN JOSE	GOICOECHEA

A continuación se muestra los datos en limpio del INEC en PostgreSQL previamente trabajados:



	Provincia, cantón y distrito text	Población de 15 años y más text	Tasa neta de participación text	Tasa de ocupación text	Tasa de desempleo abierto text	Porcentaje de población económicamente inactiva text	Relación de dependencia económica text	Sector Primario text
1	Costa Rica	3 233 882	53,5	51,7	3,4	46,5	1,5	13,9
2	San José	1 087 315	56,0	54,1	3,5	44,0	1,3	5,5
3	San José	[...] 225 856	56,7	54,5	3,9	43,3	1,2	0,7
4	Carmen	[...] 2 431	56,3	54,8	2,7	43,7	1,0	1,6
5	Merced	[...] 9 655	59,2	57,0	3,8	40,8	1,1	0,8
6	Hospital	[...] 15 096	56,4	53,9	4,4	43,6	1,3	0,5
7	Catedral	[...] 10 742	59,0	57,0	3,4	41,0	1,0	0,5
8	Zapote	[...] 15 545	55,7	54,0	2,9	44,3	1,2	0,6
9	San Francisco de Dos Ríos	[...] 16 895	57,4	56,1	2,3	42,6	1,1	0,5
10	Uruca	[...] 22 717	59,0	56,6	4,0	41,0	1,4	1,0
11	Mata Redonda	[...] 7 077	54,0	52,2	3,4	46,0	1,2	1,1
12	Pavas	[...] 54 510	56,3	53,9	4,2	43,7	1,3	0,7
13	Hatillo	[...] 39 555	54,7	52,2	4,6	45,3	1,3	0,5

Conclusiones

- Con lo que respecta a la plataforma de Spark a primera vista se puede apreciar una gran ventaja del uso del mismo, es que una de las propiedades más interesantes de una solución de código abierto. Algo a tomar en cuenta es que la comunidad no deja de crecer, por lo que la plataforma como tal, puede tener una continua mejora con lo que respecta a actualizaciones que le puedan dar los desarrolladores como tal.
- Una de las principales características que se dió cuenta a la hora de investigar sobre Spark y el uso de la misma que es que las referencias como tal comentan que su velocidad es significante y esta misma está por encima de algunas soluciones , ya que Spark permite a los desarrolladores realizar operaciones sobre un gran volumen de datos en clústeres de forma eficiente y con tolerancia a fallos, adicionalmente se nota que por lo que como desarrollador puede tener controlada la parte de manejar en memoria y no en disco para poder mejorar el rendimiento de un programa como tal.
- Algo que se notó en el desarrollo del mismo proyecto es que Spark es una plataforma unificada para gestionar datos, por lo que agiliza mucho el funcionamiento y el mantenimiento de soluciones el cual permite la consulta de datos estructurados utilizado con variante de usos de lenguajes de programación, pero en el contexto de este proyecto es Python, para poder trabajar de una mejor manera con Spark con consultas interactivas.

Referencias

- Esta referencia ayudó a instalar Spark en el sistema operativo de Ubuntu, ya que la opción de Windows se notó más compleja de tratar de lo que se esperaba: https://www.youtube.com/watch?v=DZ-ciI5_CQw
- Esta referencia aportó en la instalación de apache Spark en Ubuntu: https://netmiko.com/tutorials/data-science/install-apache-spark-on-ubuntu/
- Tutorial para poder entender de una mejor manera la relación entre Spark y Python: https://www.youtube.com/watch?v=gq4defdZ800
- Tutorial para poder entender de una mejor manera la relación entre Spark y Python: https://spark.apache.org/docs/latest/sql-data-sources-jdbc.html
- Esta referencia aportó para poder trasladar las variables de Spark SQL a Python: https://stackoverflow.com/questions/44582450/how-to-pass-variables-in-spark-sql-using-python
- Vídeo para ayudar a integrar Spark a un cuaderno de jupyter: https://www.youtube.com/watch?v=8NWYCbQGX0Y
- Vídeo de apoyo para poder instalar Spark en Ubuntu: https://www.youtube.com/watch?v=HQGJekWGUMI
- El vídeo a continuación, fue de apoyo para poder tener en cuenta un mejor funcionamiento con lo que respecta al uso de Spark junto a un cuaderno de jupyter: https://www.youtube.com/watch?v=XvbEADU0IPU
- Con lo que respecta al vídeo que se va a presentar a continuación, aportó una mejor claridad con lo que respecta al uso de Spark en un cuaderno de jupyter, viéndolo de una forma mas visual que ayuda bastante: https://www.youtube.com/watch?v=_C8kWso4ne4