

# Análisis Geográfico de las sucursales OXXO en la Ciudad de México

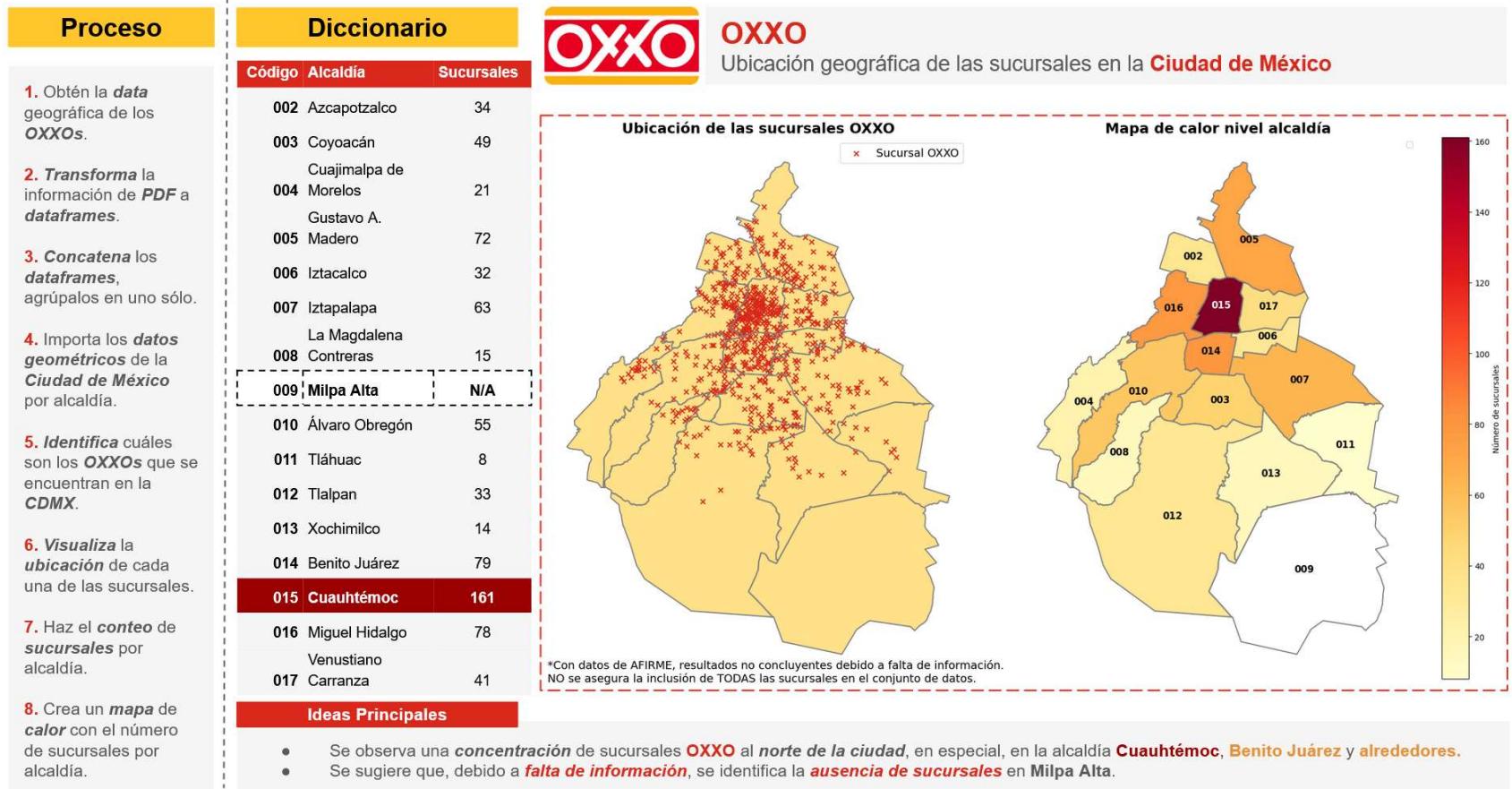
Fuente de información: OXXO Tiendas

```
In [ ]: import pandas as pd          # Data manipulation
import numpy as np           # Numerical python
import matplotlib.pyplot as plt # Data visualization
import geopandas as gpd        # Geospatial data manipulation
import tabula                 # Library for PDF data extraction
from PIL import Image          # Image manipulation
import warnings               # Warnings util

warnings.filterwarnings('ignore') # Ignore filter warnings
```

```
In [ ]: # Abre el onepager FINAL
Image.open("data/onepager_OXXO.png")
```

Out[ ]:



```
In [ ]: # Lee las tablas dentro del pdf
tables = tabula.read_pdf("data/OXXO_TIENDAS.pdf", pages = "all", pandas_options = {"header": None})

# Muestra el objeto
tables
```

```
In [ ]: # Concatena toda la info de los datos en un mismo dataframe
oxxos = pd.concat(tables[1:])

# Columns de interés (sin NAN)
oxxos_cols = list(set(oxxos.columns).difference(set([5, 13, 15, 16, 17])))

# Selecciona solo las columnas de interés
oxxos = oxxos[oxxos_cols]

# Genera una lista con las columnas de las tablas
oxxo_cols = ["NOMBRE_PLAZA", "NOMBRE TIENDA", "EMPRESA_PERTENECE", "CALLE",
             "NUMERO & ENTRE_CALLES", "CODIGO", "FRANJA FRONTERIZA",
             "COLONIA", "MUNICIPIO_DELEGACION", "CIUDAD", "ESTADO", "LATITUD",
             "LONGITUD"]

# Cambia el nombre de las columnas
oxxos.columns = oxxo_cols

# Imprime la estructura
print(oxxos.info(), "\n")

# Imprime el dataframe
oxxos.head()
```

```

<class 'pandas.core.frame.DataFrame'>
Int64Index: 15934 entries, 0 to 72
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   NOMBRE_PLAZA      15934 non-null   object  
 1   NOMBRE TIENDA     15934 non-null   object  
 2   EMPRESA_PERTENECE 15934 non-null   object  
 3   CALLE             15933 non-null   object  
 4   NUMERO & ENTRE_CALLES 15934 non-null   object  
 5   CODIGO            15932 non-null   object  
 6   FRANJA FRONTERIZA 15934 non-null   object  
 7   COLONIA           15932 non-null   object  
 8   MUNICIPIO_DELEGACION 15676 non-null   object  
 9   CIUDAD            15932 non-null   object  
 10  ESTADO             15676 non-null   object  
 11  LATITUD            15701 non-null   object  
 12  LONGITUD           8181 non-null    object  
dtypes: object(13)
memory usage: 1.7+ MB
None

```

	NOMBRE_PLAZA	NOMBRE_TIENDA	EMPRESA_PERTENECE	CALLE	NUMERO & ENTRE_CALLES	CODIGO	FRANJA FRONTERIZA	COLONIA	MUNICIPIO_DELI
0	Mexico	Valenciana MEX	CADENA COMERCIAL OXXO SA. DE CV.	José María Parras	300 Gral. Fco. Leyva y Gral. F.Arce.	9100	N	JUAN ESCUTIA	IZT
1	Laguna	SAN IGNACIO TRC	CADENA COMERCIAL OXXO SA. DE CV.	San Ignacio	704 Calzada la calle y san Ignacio	35113	N	JERUSALEN	GOMEZ
2	Laguna	Valle de Chapala TRC	CADENA COMERCIAL OXXO SA. DE CV.	AV. CENTRAL	317 CALLE DURANGO Y CALLE GUANAJUATO	35025	N	CHAPALA ORIENTE	GOMEZ
3	Chihuahua	Lealtad CUF	CADENA COMERCIAL OXXO SA. DE CV.	Av. Solidaridad y Canal	1000 Av. Solidaridad y Canal	33029	N	OBRERA	
4	Ciudad Juarez	La Cuesta CJS	CADENA COMERCIAL OXXO SA. DE CV.	CARR. JUAREZ CASAS GRANDES	6120 CORDILLERA DE LOS ANDES Y CARR. JUAREZ CA...	32650	S	LA CUESTA 1	

```

In [ ]: # Selecciona solo los datos sin NAN
oxxos = oxxos.dropna().astype({"LONGITUD":"float64", "LATITUD":"float64"})

# Genera un geodataframe
oxxos = gpd.GeoDataFrame(oxxos)

# Imprime la estructura del dataframe
print(oxxos.info(), "\n")

# Genera Los points
oxxos["ubicacion"] = gpd.points_from_xy(oxxos["LONGITUD"], oxxos["LATITUD"])

# Imprime el head
oxxos.head()

```

```

<class 'geopandas.geodataframe.GeoDataFrame'>
Int64Index: 7915 entries, 0 to 72
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   NOMBRE_PLAZA      7915 non-null   object  
 1   NOMBRE TIENDA     7915 non-null   object  
 2   EMPRESA_PERTENECE 7915 non-null   object  
 3   CALLE             7915 non-null   object  
 4   NUMERO & ENTRE_CALLES 7915 non-null   object  
 5   CODIGO            7915 non-null   object  
 6   FRANJA FRONTERIZA 7915 non-null   object  
 7   COLONIA           7915 non-null   object  
 8   MUNICIPIO_DELEGACION 7915 non-null   object  
 9   CIUDAD            7915 non-null   object  
 10  ESTADO             7915 non-null   object  
 11  LATITUD            7915 non-null   float64 
 12  LONGITUD           7915 non-null   float64 
dtypes: float64(2), object(11)
memory usage: 865.7+ KB
None

```

	NOMBRE_PLAZA	NOMBRE TIENDA	EMPRESA_PERTENECE	CALLE	NUMERO & ENTRE_CALLES	CODIGO	FRANJA FRONTERIZA	COLONIA	MUNICIPIO_DELI
0	Mexico	Valenciana MEX	CADENA COMERCIAL OXXO SA. DE CV.	José María Parras	300 Gral. Fco. Leyva y Gral. F.Arce.	9100	N	JUAN ESCUTIA	IZT
1	Laguna	SAN IGNACIO TRC	CADENA COMERCIAL OXXO SA. DE CV.	San Ignacio	704 Calzada la calle y san Ignacio	35113	N	JERUSALEN	GOMEZ
2	Laguna	Valle de Chapala TRC	CADENA COMERCIAL OXXO SA. DE CV.	AV. CENTRAL	317 CALLE DURANGO Y CALLE GUANAJUATO	35025	N	CHAPALA ORIENTE	GOMEZ
3	Chihuahua	Lealtad CUF	CADENA COMERCIAL OXXO SA. DE CV.	Av. Solidaridad y Canal	1000 Av. Solidaridad y Canal	33029	N	OBRERA	
4	Ciudad Juarez	La Cuesta CJS	CADENA COMERCIAL OXXO SA. DE CV.	CARR. JUAREZ CASAS GRANDES	6120 CORDILLERA DE LOS ANDES Y CARR. JUAREZ CA...	32650	S	LA CUESTA 1	

```
In [ ]: # Importa los datos con Los Limites de Las alcaldias
cdmx = gpd.read_file("data/CDMX_mpal.geojson") # Debido al tamaño del archivo, este no se encuentra en el repositorio

# Pon el geometry
cdmx.set_geometry("geometry", inplace = True)

# Calcula el centroid
cdmx["centroid"] = cdmx.centroid

# Imprime los datos
cdmx
```

<ipython-input-7-668fe109bb7e>:8: UserWarning: Geometry is in a geographic CRS. Results from 'centroid' are likely incorrect. Use 'GeoSeries.to\_crs()' to re-project geometries to a projected CRS before this operation.

```
cdmx["centroid"] = cdmx.centroid
```

	CVE_ENT	CVE_MUN	NOM_MUN	OID_1	cov_	cov_id	geometry	centroid
0	09	002	Azcapotzalco	337	337	338	POLYGON ((-99.18245 19.50756, -99.18231 19.507...	POINT (-99.18211 19.48533)
1	09	005	Gustavo A. Madero	338	338	339	POLYGON ((-99.11789 19.59059, -99.11860 19.584...	POINT (-99.11586 19.50407)
2	09	016	Miguel Hidalgo	339	339	340	POLYGON ((-99.19044 19.47046, -99.19058 19.467...	POINT (-99.20457 19.42806)
3	09	006	Iztacalco	340	340	341	POLYGON ((-99.05579 19.42214, -99.05584 19.421...	POINT (-99.09433 19.39691)
4	09	017	Venustiano Carranza	341	341	342	POLYGON ((-99.10946 19.45292, -99.10895 19.452...	POINT (-99.09311 19.43050)
5	09	007	Iztapalapa	342	342	343	POLYGON ((-99.05813 19.40072, -99.05814 19.400...	POINT (-99.05680 19.34917)
6	09	012	Tlalpan	2448	2448	2449	POLYGON ((-99.19671 19.30240, -99.19629 19.302...	POINT (-99.20622 19.19834)
7	09	013	Xochimilco	2449	2449	2450	POLYGON ((-99.09880 19.32045, -99.09870 19.319...	POINT (-99.09036 19.24515)
8	09	008	La Magdalena Contreras	2450	2450	2451	POLYGON ((-99.20819 19.33674, -99.20859 19.336...	POINT (-99.26841 19.26898)
9	09	014	Benito Juárez	2451	2451	2452	POLYGON ((-99.13659 19.40281, -99.13679 19.401...	POINT (-99.16113 19.38064)
10	09	015	Cuauhtémoc	2452	2452	2453	POLYGON ((-99.12951 19.46265, -99.12919 19.462...	POINT (-99.14906 19.43137)
11	09	010	Álvaro Obregón	2453	2453	2454	POLYGON ((-99.18906 19.39559, -99.18871 19.394...	POINT (-99.24682 19.33618)
12	09	004	Cuajimalpa de Morelos	2454	2454	2455	POLYGON ((-99.25738 19.40112, -99.25698 19.400...	POINT (-99.31073 19.32463)
13	09	003	Coyoacán	2455	2455	2456	POLYGON ((-99.13984 19.35692, -99.13923 19.356...	POINT (-99.15038 19.32667)
14	09	011	Tláhuac	2456	2456	2457	POLYGON ((-98.97893 19.32393, -98.97856 19.323...	POINT (-99.00282 19.27700)
15	09	009	Milpa Alta	2457	2457	2458	POLYGON ((-98.99718 19.22747, -98.99723 19.227...	POINT (-99.05110 19.13946)

```
In [ ]: # La union de La geometria de todas Las alcaldias
cdmx_boundary = cdmx["geometry"].unary_union
```

```
In [ ]: # Crea un nuevo dataframe SOLO con los oxxos dentro de la ciudad
oxxos_cdmx = oxxos[oxxos["ubicacion"].within(cdmx_boundary) == True].set_geometry("ubicacion")

# Imprime el dataframe
oxxos_cdmx
```

Out[ ]:

	NOMBRE_PLAZA	NOMBRE TIENDA	EMPRESA_PERTENECE	CALLE	NUMERO & ENTRE_CALLES	CODIGO	FRANJA FRONTERIZA	COLONIA	MUNICIPI
0	Mexico	Valenciana MEX	CADENA COMERCIAL OXXO SA. DE CV.	José María Parras	300 Gral. Fco. Leyva y Gral. F.Arce.	9100	N	JUAN ESCUTIA	
27	Mexico	De Dios MEX	CADENA COMERCIAL OXXO SA. DE CV.	VIADUCTO TLALPAN	104 ESQ SAN JUAN DE DIOS	14370	N	SAN LORENZO HUIPULCO	
28	Mexico	Triangulo MEX	CADENA COMERCIAL OXXO SA. DE CV.	Av. Tlahuac	7260 Mar de la Tranquilidad y Monte de las Cor...	13460	N	EL TRIANGULO	
29	Mexico	Acahualtepec MEX	CADENA COMERCIAL OXXO SA. DE CV.	Av. Octavio Senties	39 Bugambilia y Cda, Magnolia	7183	N	6 DE JUNIO	
33	Mexico	Felicidad MEX	CADENA COMERCIAL OXXO SA. DE CV.	AV. MIGUEL HIDALGO	LOTE 7 MARIANO ABASOLO Y DILIGENCIAS	14700	N	SAN MIGUEL AJUSCO	
...	...	...	...	...	...	...	...	...	...
34	Mexico	La Cebada MEX	CADENA COMERCIAL OXXO SA. DE CV.	Av. Plan de Muyuguarda	192 10 de Abril de 1919 y Reforma laboral	16035	N	SAN LORENZO LA CEBADA	
35	Mexico	YMCA Mayorca MEX	CADENA COMERCIAL OXXO SA. DE CV.	Laboristas	49 Playa Pie de la Cuesta y Normandía	9440	N	*ZACAHUITZCO	
36	Mexico	Tulipan MEX	CADENA COMERCIAL OXXO SA. DE CV.	Av. Santiago	48 Esquina Abasolo	8800	N	SANTIAGO SUR	
37	Mexico	Nueva York MEX	CADENA COMERCIAL OXXO SA. DE CV.	NUEVA YORK	115 A NUEVA YORK Y ARIZONA	3810	N	NAPOLES	
69	Mexico	Bolivar MEX	CADENA COMERCIAL OXXO SA. DE CV.	BOLIVAR	608 GALICIA Y CASTILLA	3400	N	ALAMOS	

755 rows × 14 columns

```
In [ ]: def ubica_alcaldia(ubicacion):
```

```
# Define alcaldia con un valor None al inicio
alcaldia = None

# Para cada uno de los registros de las alcaldias identifica si el punto esta dentro
# del poligono
for i in range(cdmx.shape[0]):

    # Identifica el i record dentro de cdmx
    cdmx_record = cdmx.iloc[i]

    # Identifica si el punto esta dentro del poligono de la alcaldia
    if ubicacion.within(cdmx_record["geometry"]):
        alcaldia = cdmx_record["NOM_MUN"]
        break

return alcaldia
```

```
In [ ]: # Ubica la alcaldia con los mismos valores que el dataframe de cdmx
oxxos_cdmx["alcaldia_cdmx"] = oxxos_cdmx["ubicacion"].apply(ubica_alcaldia)

# Imprime el head
oxxos_cdmx.head()
```

	NOMBRE_PLAZA	NOMBRE TIENDA	EMPRESA_PERTENECE	CALLE	NUMERO & ENTRE_CALLES	CODIGO	FRANJA FRONTERIZA	COLONIA	MUNICIPIO_DE
0	Mexico	Valenciana MEX	CADENA COMERCIAL OXXO SA. DE CV.	José María Parras	300 Gral. Fco. Leyva y Gral. F.Arce.	9100	N	JUAN ESCUTIA	I
27	Mexico	De Dios MEX	CADENA COMERCIAL OXXO SA. DE CV.	VIADUCTO TLALPAN	104 ESQ SAN JUAN DE DIOS	14370	N	SAN LORENZO HUIPULCO	I
28	Mexico	Triangulo MEX	CADENA COMERCIAL OXXO SA. DE CV.	Av. Tlahuac	7260 Mar de la Tranquilidad y Monte de las Cor...	13460	N	EL TRIANGULO	I
29	Mexico	Acahualtepec MEX	CADENA COMERCIAL OXXO SA. DE CV.	Av. Octavio Senties	39 Bugambilias y Cda, Magnolia	7183	N	6 DE JUNIO	I
33	Mexico	Felicidad MEX	CADENA COMERCIAL OXXO SA. DE CV.	AV. MIGUEL HIDALGO	LOTE 7 MARIANO ABASOLO Y DILIGENCIAS	14700	N	SAN MIGUEL AJUSCO	I

```
In [ ]: # Crea un nuevo dataframe con Los value_counts de oxxos
cdmx_agg = pd.merge(cdmx, oxxos_cdmx["alcaldia_cdmx"].value_counts(), left_on="NOM_MUN", right_index = True, how = "outer")

# Computa el centroide de la alcaldia
cdmx_agg["centroid"] = cdmx_agg["geometry"].centroid

# Muestra el dtaframe
cdmx_agg
```

<ipython-input-12-d9ca192c996c>:5: UserWarning: Geometry is in a geographic CRS. Results from 'centroid' are likely incorrect. Use 'GeoSeries.to\_crs()' to re-project geometries to a projected CRS before this operation.

```
cdmx_agg["centroid"] = cdmx_agg["geometry"].centroid
```

	CVE_ENT	CVE_MUN	NOM_MUN	OID_1	cov_	cov_id	geometry	centroid	alcaldia_cdmx
0	09	002	Azcapotzalco	337	337	338	POLYGON ((-99.18245 19.50756, -99.18231 19.507...)	POINT (-99.18211 19.48533)	34.0
1	09	005	Gustavo A. Madero	338	338	339	POLYGON ((-99.11789 19.59059, -99.11860 19.584...)	POINT (-99.11586 19.50407)	72.0
2	09	016	Miguel Hidalgo	339	339	340	POLYGON ((-99.19044 19.47046, -99.19058 19.467...)	POINT (-99.20457 19.42806)	78.0
3	09	006	Iztacalco	340	340	341	POLYGON ((-99.05579 19.42214, -99.05584 19.421...)	POINT (-99.09433 19.39691)	32.0
4	09	017	Venustiano Carranza	341	341	342	POLYGON ((-99.10946 19.45292, -99.10895 19.452...)	POINT (-99.09311 19.43050)	41.0
5	09	007	Iztapalapa	342	342	343	POLYGON ((-99.05813 19.40072, -99.05814 19.400...)	POINT (-99.05680 19.34917)	63.0
6	09	012	Tlalpan	2448	2448	2449	POLYGON ((-99.19671 19.30240, -99.19629 19.302...)	POINT (-99.20622 19.19834)	33.0
7	09	013	Xochimilco	2449	2449	2450	POLYGON ((-99.09880 19.32045, -99.09870 19.319...)	POINT (-99.09036 19.24515)	14.0
8	09	008	La Magdalena Contreras	2450	2450	2451	POLYGON ((-99.20819 19.33674, -99.20859 19.336...)	POINT (-99.26841 19.26898)	15.0
9	09	014	Benito Juárez	2451	2451	2452	POLYGON ((-99.13659 19.40281, -99.13679 19.401...)	POINT (-99.16113 19.38064)	79.0
10	09	015	Cuauhtémoc	2452	2452	2453	POLYGON ((-99.12951 19.46265, -99.12919 19.462...)	POINT (-99.14906 19.43137)	161.0
11	09	010	Álvaro Obregón	2453	2453	2454	POLYGON ((-99.18906 19.39559, -99.18871 19.394...)	POINT (-99.24682 19.33618)	55.0
12	09	004	Cuajimalpa de Morelos	2454	2454	2455	POLYGON ((-99.25738 19.40112, -99.25698 19.400...)	POINT (-99.31073 19.32463)	21.0
13	09	003	Coyoacán	2455	2455	2456	POLYGON ((-99.13984 19.35692, -99.13923 19.356...)	POINT (-99.15038 19.32667)	49.0
14	09	011	Tláhuac	2456	2456	2457	POLYGON ((-98.97893 19.32393, -98.97856 19.323...)	POINT (-99.00282 19.27700)	8.0
15	09	009	Milpa Alta	2457	2457	2458	POLYGON ((-98.99718 19.22747, -98.99723 19.227...)	POINT (-99.05110 19.13946)	NaN

```
In [ ]: plt.figure(figsize = (18, 10))

# Colores
rojo_oxxo = "#da291c"
amarillo_oxxo = "#ffc72c"

# Eje izquierdo
ax_1 = plt.subplot(1, 2, 1)
```

```

# Grafica los limites de la alcaldia
cdmx_agg[ "geometry" ].boundary.plot( color = "gray", ax = ax_l, linewidth = 1.5 )
# Grafica el relleno de las alcaldias
cdmx_agg[ "geometry" ].plot( color = "#ffe189", ax = ax_l )
# Grafica cada uno de los oxxos
oxxos_cdmx.plot( color = rojo_oxxo, ax = ax_l, marker = "x", label = "Sucursal OXXO" )
# Agrega un legend para conocer el significado de "x"
ax_l.legend( fontsize = 14 )

# Agrega un titulo para la imagen de la izquierda
ax_l.set_title("Ubicación de las sucursales OXXO", weight = "bold", size = 18)

# Eje derecho
ax_r = plt.subplot(1, 2, 2)

# Grafica los limites de la alcaldia
cdmx_agg[ "geometry" ].boundary.plot( color = "gray", linewidth = 1.5, ax = ax_r )
# Grafica el numero de oxxos por alcaldia
cdmx_agg.plot("alcaldia_cdmx", cmap = "YlOrRd", ax = ax_r, legend = True,
               legend_kwds = { "label": "Número de sucursales"}, label = "alcaldia_cdmx")

# Ponle un titulo a la grafica
ax_r.set_title("Mapa de calor nivel alcaldía", weight = "bold", size = 18)

for i in range(cdmx.shape[0]):
    # Identifica el registro actual
    cur_record = cdmx_agg.iloc[i]
    cur_centroid = cur_record[ "centroid" ]
    cur_alcaldia = cur_record[ "NOM_MUN" ]
    cur_cve = cur_record[ "CVE_MUN" ]

    # Identifica si se habla de Cuahmetoc (el color mas oscuro)
    if cur_cve == "015":
        color = "white"
    else:
        color = "black"

    # Anotalo en la grafica
    ax_r.annotate(
        cur_cve,
        (cur_centroid.x, cur_centroid.y),
        ha = "center",
        color = color,
        weight = "bold",
        size = 12
    )

ax_r.legend()

# Apaga los axis de las graficas
ax_r.axis("off")
ax_l.axis("off")

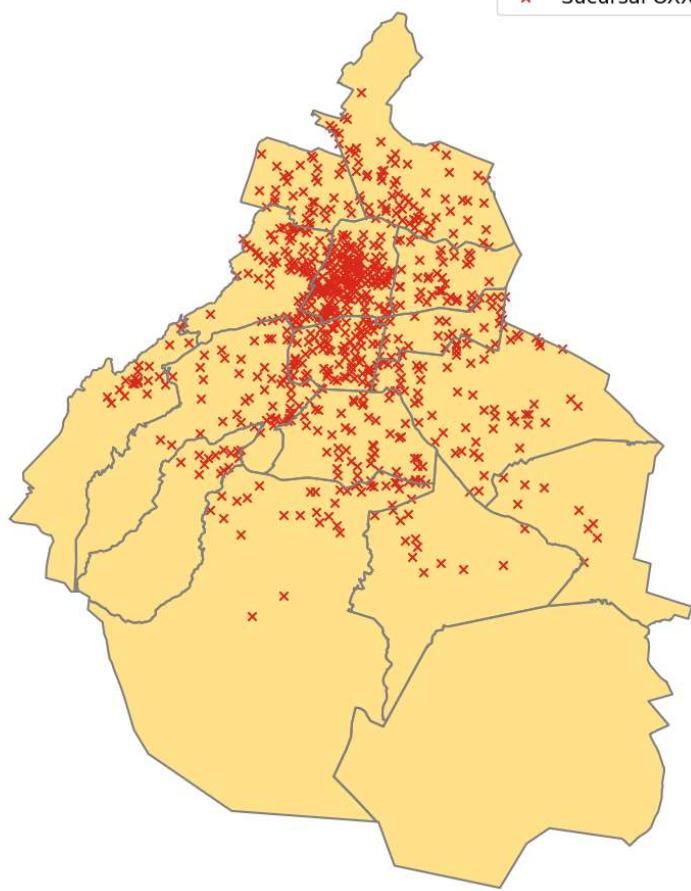
# Agrega un texto abajo
plt.gcf().text(0.05, 0.01, "*Con datos de AFIRME, resultados no concluyentes debido a falta de información.\nNO se aseg",
               size = 14)

# Comprime los datos
plt.tight_layout()

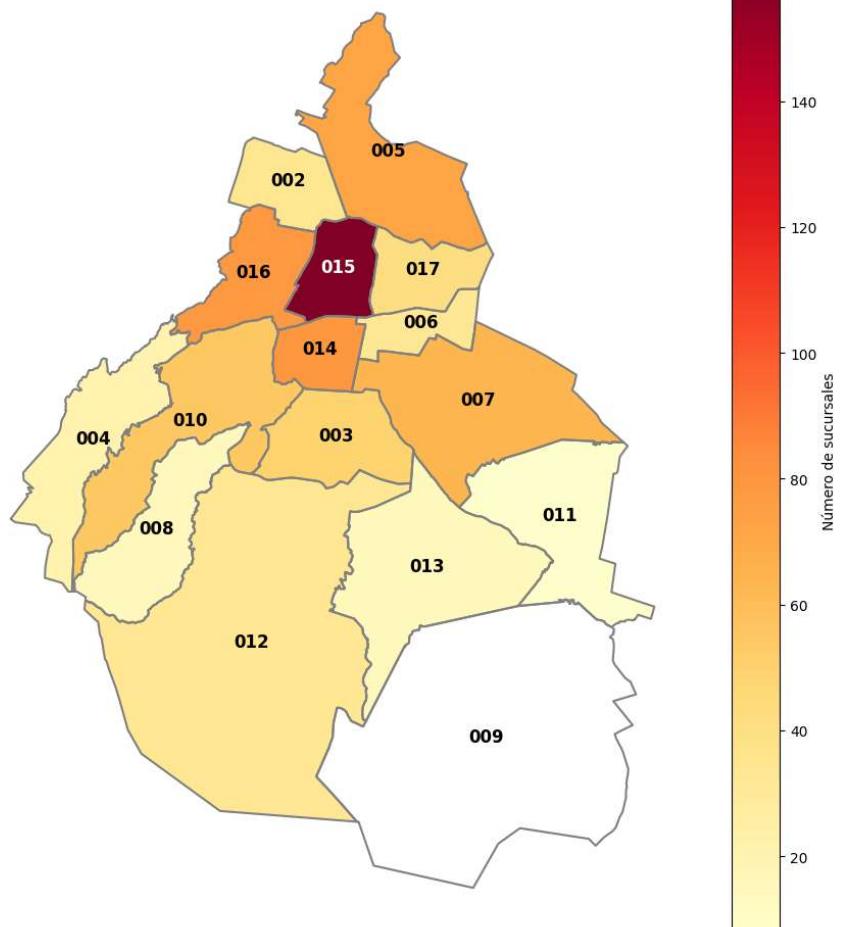
```

WARNING:matplotlib.legend:No artists with labels found to put in legend. Note that artists whose label start with an underscore are ignored when legend() is called with no argument.

**Ubicación de las sucursales OXXO**



**Mapa de calor nivel alcaldía**



\*Con datos de AFIRME, resultados no concluyentes debido a falta de información.  
NO se asegura la inclusión de TODAS las sucursales en el conjunto de datos.