In [13]:
```
archivo = 'dataCreditos.xls'
data = pd.read_excel(archivo)
print(data.head())
print(data.info())
print(data.describe())
```

```
  CIUDAD FECHA LIMITE    ZONA  VALOR FACTURA   SALDO FECHA LLAMADA  \
0  COSTA    2016-05-06  18402         162948   29122    5/05/2016
1  COSTA    2016-05-11  17102         215156   85156    5/05/2016
2  COSTA    2016-05-06  18202         113631  113525    30/04/2016
3  COSTA    2016-05-11  17403         187560   87438    3/05/2016
4  COSTA    2016-05-16  46106         177603   32465    3/05/2016

   TIPO DE CONTACTO ESTADO CODIFICACION FECHA DE COMPROMISO   ...  SECCION  \
0  Contacto Directo   pago         pago                NaN  ...        3
1  Contacto Directo   pago         pago                NaN  ...        1
2  Contacto Directo   pago         pago                NaN  ...        3
3  Contacto Directo   pago         pago                NaN  ...        4
4  Contacto Directo   pago         pago                NaN  ...        3

   CANTIDAD_CAMPANAS  ANTIGUEDAD               NOMBRE POBLADO  \
0                 17          29                    EL BANCO
1                  2           5                   SINCELEJO
2                 19          36                  VALLEDUPAR
3                126         144  SAN SEBASTIAN DE BUENAVISTA
4                 14          37                BARRANQUILLA

  BARRIO RESIDENCIA     FECHA NACIMIENTO FECHA FACTURA ULTIMO PAGO  \
0      PUEBLO NUEVO  1988-05-30 00:00:00    2016-01-05  2016-03-16
1            CHOCHO  1989-08-07 00:00:00    2016-01-07  2016-02-20
2       20 DE JULIO  1993-11-08 00:00:00    2016-01-05         NaT
3        SAN MIGUEL  1962-04-06 00:00:00    2016-01-08  2016-03-02
4          ALVORADA  1995-09-20 00:00:00    2016-01-13  2016-03-22

  FECHA PRIM FACT DIAS MORA
0      2014-08-29        85
1      2015-12-16        80
2      2014-04-10        85
3      2008-07-08        77
4      2014-04-01        77

[5 rows x 21 columns]
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 911 entries, 0 to 910
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   CIUDAD                911 non-null    object
 1   FECHA LIMITE          911 non-null    datetime64[ns]
 2   ZONA                  911 non-null    int64
 3   VALOR FACTURA         911 non-null    int64
 4   SALDO                 911 non-null    int64
 5   FECHA LLAMADA         911 non-null    object
 6   TIPO DE CONTACTO      911 non-null    object
 7   ESTADO                459 non-null    object
 8   CODIFICACION          911 non-null    object
 9   FECHA DE COMPROMISO   114 non-null    object
 10  Vr. COMPROMISO        135 non-null    float64
 11  SECCION               911 non-null    int64
 12  CANTIDAD_CAMPANAS     911 non-null    int64
 13  ANTIGUEDAD            911 non-null    int64
 14  NOMBRE POBLADO        911 non-null    object
 15  BARRIO RESIDENCIA     910 non-null    object
 16  FECHA NACIMIENTO      910 non-null    object
```

```
 17  FECHA FACTURA          911 non-null    datetime64[ns]
 18  ULTIMO PAGO            310 non-null    datetime64[ns]
 19  FECHA PRIM FACT        911 non-null    datetime64[ns]
 20  DIAS MORA              911 non-null    int64
dtypes: datetime64[ns](4), float64(1), int64(7), object(9)
memory usage: 149.6+ KB
None
                ZONA  VALOR FACTURA          SALDO  Vr. COMPROMISO     SECCION  \
count    911.000000     911.000000     911.000000      135.000000  911.000000
mean   18109.037322  186377.140505  140256.656422    66137.303704    3.582876
std     7075.327785   92760.053874   95246.787313    76973.131287    2.214709
min    12102.000000   28118.000000   15236.000000        0.000000    0.000000
25%    16114.000000  131893.000000   73200.500000        0.000000    2.000000
50%    17104.000000  163947.000000  128719.000000    50000.000000    3.000000
75%    18101.000000  217165.500000  174032.000000   100000.000000    5.000000
max    48501.000000  978789.000000  673631.000000   594000.000000   11.000000

       CANTIDAD_CAMPANAS  ANTIGUEDAD   DIAS MORA
count         911.000000  911.000000  911.000000
mean           14.459934   31.654226   77.761800
std            23.946283   48.255229    6.194406
min             1.000000    4.000000    0.000000
25%             2.000000    5.000000   76.000000
50%             6.000000   10.000000   78.000000
75%            15.000000   34.000000   80.000000
max           205.000000  287.000000   85.000000
```

In [22]:

```python
import pandas as pd
from datetime import datetime

# modulo para actualizar la edad del cliente
ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)
data['FECHA NACIMIENTO'] = pd.to_datetime(data['FECHA NACIMIENTO'])

def calcular_edad(fecha_nacimiento):
    fecha_actual = datetime.now()
    edad = fecha_actual.year - fecha_nacimiento.year - ((fecha_actual.month, fecha_actua
    return edad

data['EDAD ACTUAL'] = data['FECHA NACIMIENTO'].apply(calcular_edad)

print(data[['FECHA NACIMIENTO', 'EDAD ACTUAL']])
```

```
    FECHA NACIMIENTO  EDAD ACTUAL
0         1988-05-30         35.0
1         1989-08-07         34.0
2         1993-11-08         29.0
3         1962-04-06         61.0
4         1995-09-20         27.0
..               ...          ...
906       1984-11-21         38.0
907       1978-11-18         44.0
908       1962-06-22         61.0
909       1947-04-07         76.0
910       1973-12-18         49.0

[911 rows x 2 columns]
```

In [21]:

```python
import pandas as pd
from datetime import datetime

# MODULO PARA ACTUALIZAR LA ANTIGUEDADA DE LA FACTURA
ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)
```

```
data['FECHA FACTURA'] = pd.to_datetime(data['FECHA FACTURA'])

def calcular_antiguedad(fecha_factura):
    fecha_actual = datetime.now().date()
    antiguedad = (fecha_actual - fecha_factura.date()).days
    return antiguedad

data['ANTIGUEDAD'] = data['FECHA FACTURA'].apply(calcular_antiguedad)

print(data[['FECHA FACTURA', 'ANTIGUEDAD']])
```

```
    FECHA FACTURA  ANTIGUEDAD
0      2016-01-05        2784
1      2016-01-07        2782
2      2016-01-05        2784
3      2016-01-08        2781
4      2016-01-13        2776
..            ...         ...
906    2016-01-07        2782
907    2016-01-13        2776
908    2016-01-15        2774
909    2016-01-08        2781
910    2016-01-18        2771

[911 rows x 2 columns]
```

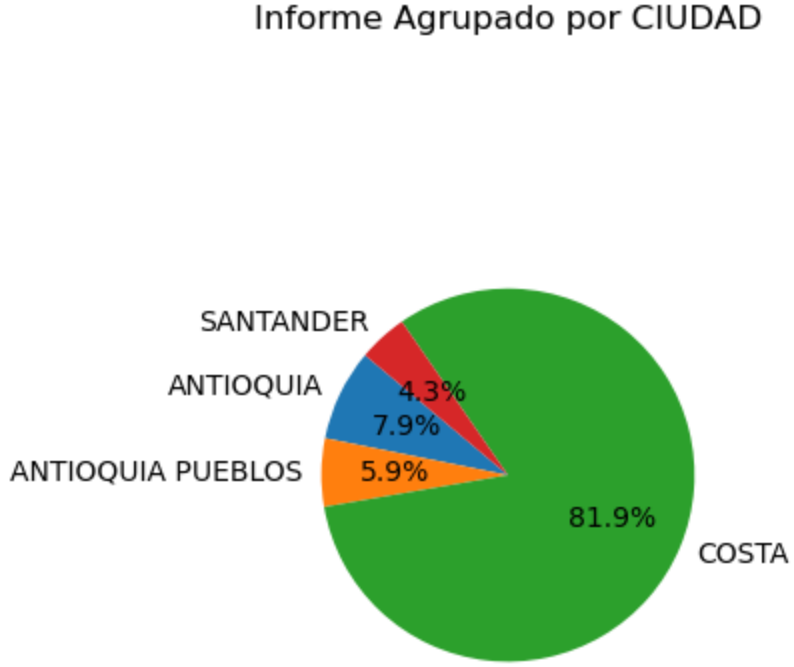
In [9]:

```
import pandas as pd

ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

informe = data.groupby('CIUDAD')['VALOR FACTURA'].sum()

print("Informe Agrupado por Nombre Poblado:")
print(informe)
```

```
Informe Agrupado por Nombre Poblado:
CIUDAD
ANTIOQUIA             13473861
ANTIOQUIA PUEBLOS     10026537
COSTA                139054822
SANTANDER              7234355
Name: VALOR FACTURA, dtype: int64
```

In [14]:

```
import pandas as pd
import matplotlib.pyplot as plt

ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

# Generar informe agrupado por ciudad
informe = data.groupby('CIUDAD')['VALOR FACTURA'].sum()

plt.subplot(1, 2, 2)
plt.pie(informe, labels=informe.index, autopct='%1.1f%%', startangle=140)
plt.title('Informe Agrupado por CIUDAD')
plt.axis('equal')
plt.tight_layout()

plt.show()
```

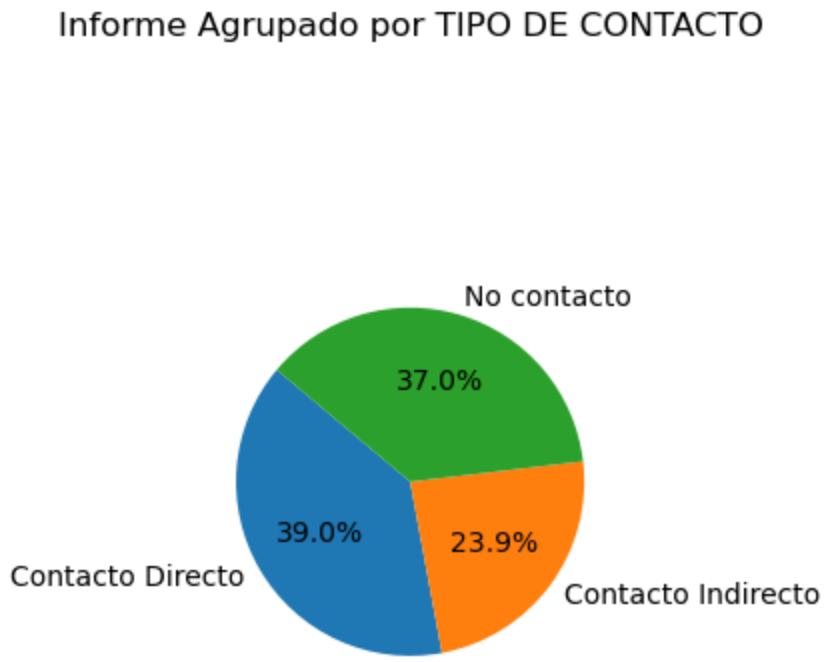


In [29]:
```python
import pandas as pd

# Cargar el archivo Excel
ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

# Clasificación de los rangos de mora y creación de la columna 'RANGO MORA'
data['RANGO MORA'] = pd.cut(data['DIAS MORA'], bins=[-1, 90, 240, float('inf')],
                            labels=['RECUPERABLE', 'RIESGO', 'CASTIGO'])


print(data[['DIAS MORA', 'RANGO MORA']])
```

```
     DIAS MORA   RANGO MORA
0           85  RECUPERABLE
1           80  RECUPERABLE
2           85  RECUPERABLE
3           77  RECUPERABLE
4           77  RECUPERABLE
..         ...          ...
906         79  RECUPERABLE
907         83  RECUPERABLE
908         72  RECUPERABLE
909         83  RECUPERABLE
910         63  RECUPERABLE

[911 rows x 2 columns]
```

In [24]:
```python
import pandas as pd
import matplotlib.pyplot as plt

ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

# Generar informe agrupado por TIPO DE CONTACTO
```

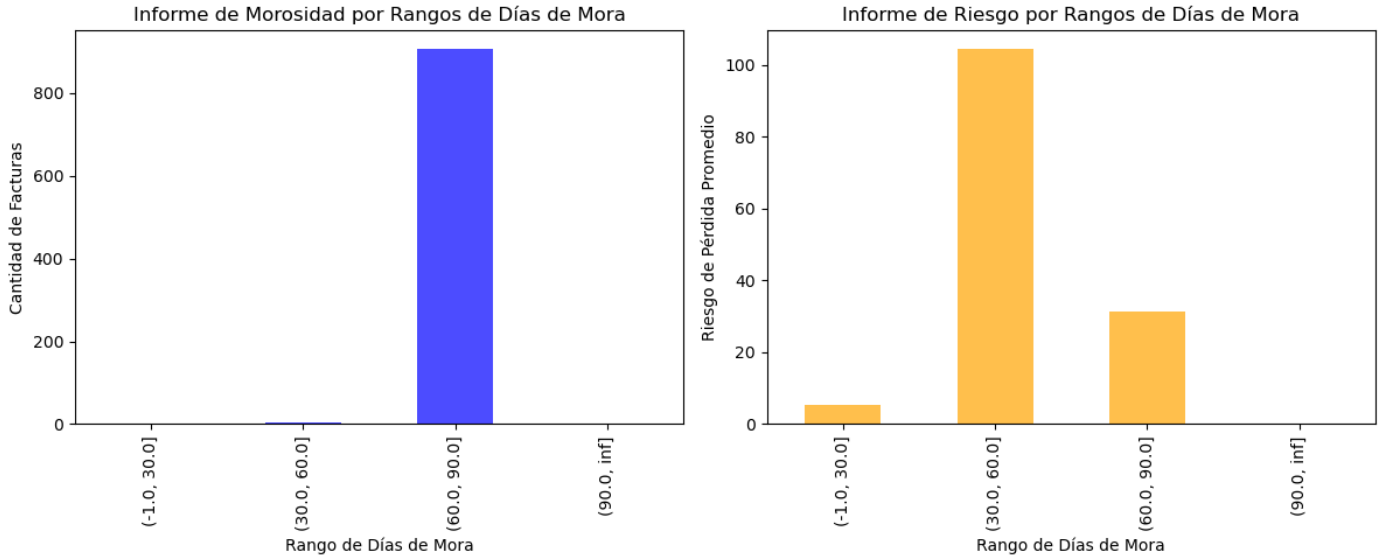
```
informe = data.groupby('TIPO DE CONTACTO')['VALOR FACTURA'].sum()

plt.subplot(1, 2, 2)  # Subplot para el gráfico de torta
plt.pie(informe, labels=informe.index, autopct='%1.1f%%', startangle=140)
plt.title('Informe Agrupado por TIPO DE CONTACTO')
plt.axis('equal')
plt.tight_layout()

plt.show()
```



In [26]:

```
import pandas as pd
import matplotlib.pyplot as plt

ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

# Informe de morosidad por rangos de días de mora
mora_por_dias = data.groupby(pd.cut(data['DIAS MORA'], bins=[-1, 30, 60, 90, float('inf'

# Informe de riesgo por rangos de días de mora
riesgo_por_dias = data.groupby(pd.cut(data['DIAS MORA'], bins=[-1, 30, 60, 90, float('in


plt.figure(figsize=(12, 5))
plt.subplot(1, 2, 1)
mora_por_dias.plot(kind='bar', color='blue', alpha=0.7)
plt.title('Informe de Morosidad por Rangos de Días de Mora')
plt.xlabel('Rango de Días de Mora')
plt.ylabel('Cantidad de Facturas')

plt.subplot(1, 2, 2)
riesgo_por_dias.plot(kind='bar', color='orange', alpha=0.7)
plt.title('Informe de Riesgo por Rangos de Días de Mora')
plt.xlabel('Rango de Días de Mora')
plt.ylabel('Riesgo de Pérdida Promedio')
```

```
plt.tight_layout()

plt.show()
```

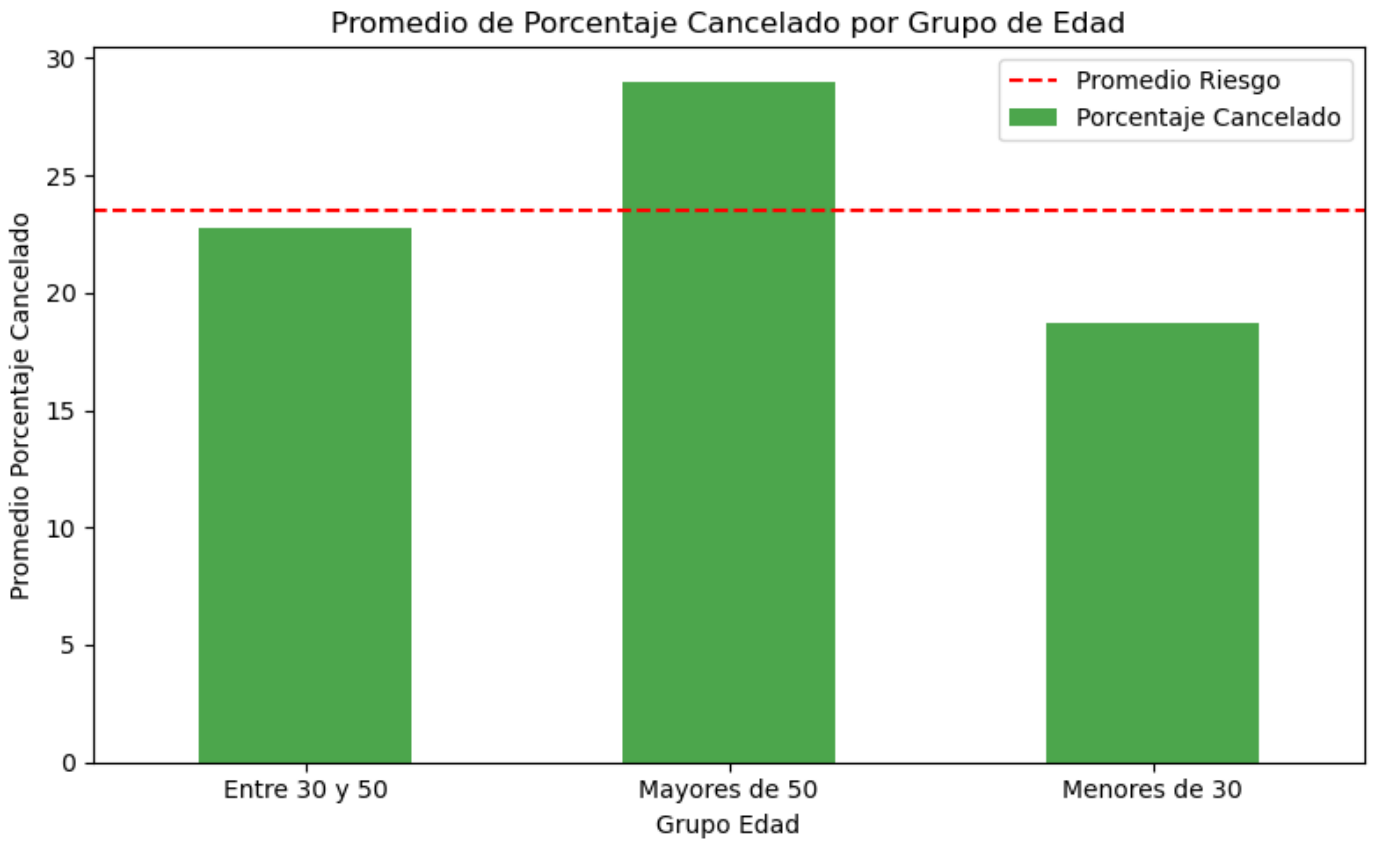


In [35]:
```
import pandas as pd

# Cargar el archivo Excel
ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

# Realizar la resta entre las columnas 'Valor Factura' y 'Saldo'
data['Valor Cancelado'] = data['VALOR FACTURA'] - data['SALDO']

# Calcular el porcentaje del valor cancelado con respecto al valor factura
data['Porcentaje Cancelado'] = ((data['Valor Cancelado'] / data['VALOR FACTURA']) * 100)

# Agregar el símbolo '%' al porcentaje
data['Porcentaje Cancelado'] = data['Porcentaje Cancelado'].astype(str) + '%'

# Mostrar solo las columnas 'Valor Factura', 'Saldo' y 'Porcentaje Cancelado'
print(data[['VALOR FACTURA', 'SALDO', 'Porcentaje Cancelado']])
```

```
     VALOR FACTURA   SALDO Porcentaje Cancelado
0           162948   29122                 82%
1           215156   85156                 60%
2           113631  113525                  0%
3           187560   87438                 53%
4           177603   32465                 81%
..             ...     ...                 ...
906         225425  100386                 55%
907         128619   30820                 76%
908         334580   54542                 83%
909         208044   22913                 88%
910         160606   37412                 76%

[911 rows x 3 columns]
```

In [43]:
```
import pandas as pd
import matplotlib.pyplot as plt

ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

data['FECHA NACIMIENTO'] = pd.to_datetime(data['FECHA NACIMIENTO'])

hoy = pd.to_datetime('today')
data['EDAD ACTUAL'] = (hoy - data['FECHA NACIMIENTO']).astype('<m8[Y]')
```

```
# Definir los grupos de edad
def asignar_grupo_edad(edad):
    if edad < 30:
        return 'Menores de 30'
    elif 30 <= edad < 50:
        return 'Entre 30 y 50'
    else:
        return 'Mayores de 50'

data['Grupo Edad'] = data['EDAD ACTUAL'].apply(asignar_grupo_edad)

data['Valor Cancelado'] = data['VALOR FACTURA'] - data['SALDO']

data['Porcentaje Cancelado'] = ((data['Valor Cancelado'] / data['VALOR FACTURA']) * 100)

promedio_por_grupo = data.groupby('Grupo Edad')['Porcentaje Cancelado'].mean()

plt.figure(figsize=(8, 5))
promedio_por_grupo.plot(kind='bar', color='green', alpha=0.7)
plt.axhline(y=promedio_por_grupo.mean(), color='red', linestyle='--', label='Promedio Ri
plt.title('Promedio de Porcentaje Cancelado por Grupo de Edad')
plt.xlabel('Grupo Edad')
plt.ylabel('Promedio Porcentaje Cancelado')
plt.xticks(rotation=0)
plt.legend()
plt.tight_layout()

plt.show()
```



In [45]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Cargar el archivo Excel
ruta_archivo_excel = 'dataCreditos.xls'
data = pd.read_excel(ruta_archivo_excel)

# Convertir la columna 'Fecha de Nacimiento' a datetime
```

```
data['Fecha de Nacimiento'] = pd.to_datetime(data['FECHA NACIMIENTO'])

# Calcular la edad actual basada en la fecha de nacimiento
hoy = pd.to_datetime('today')
data['EDAD ACTUAL'] = (hoy - data['FECHA NACIMIENTO']).astype('<m8[Y]')

# Definir los grupos de edad
def asignar_grupo_edad(edad):
    if edad < 30:
        return 'Menores de 30'
    elif 30 <= edad < 50:
        return 'Entre 30 y 50'
    else:
        return 'Mayores de 50'

# Agregar la columna 'Grupo Edad'
data['Grupo Edad'] = data['EDAD ACTUAL'].apply(asignar_grupo_edad)

# Crear un gráfico de barras agrupando por tipo de contacto y grupo de edad
grupo_edad_tipo_contacto = data.groupby(['TIPO DE CONTACTO', 'Grupo Edad'])['TIPO DE CON

# Crear el gráfico de barras agrupado por tipo de contacto
plt.figure(figsize=(10, 6))
grupo_edad_tipo_contacto.plot(kind='bar', stacked=True)
plt.title('Distribución de Grupos de Edad por Tipo de Contacto')
plt.xlabel('Tipo de Contacto')
plt.ylabel('Cantidad de Clientes')
plt.xticks(rotation=45)
plt.legend(title='Grupo Edad')
plt.tight_layout()

plt.show()
```

```
<Figure size 1000x600 with 0 Axes>
```