# analisis

April 16, 2018

In [68]:
```
# importacion general de librerias y de visualizacion (matplotlib y seaborn)
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline

plt.style.use('default') # haciendo los graficos un poco mas bonitos en matplotlib

sns.set(style="whitegrid") # seteando tipo de grid en seaborn
```

### 0.0.1 Carga de datos corregidos

In [2]:
```
# %timeit sirve para evaluar el tiempo de ejecucion
print("------------------");
df_postulantes_educacion = pd.read_csv('../csv/datos_navent_fiuba/fiuba_1_postulantes_
print("###", end='');
df_postulantes_genero_edad = pd.read_csv('../csv/datos_navent_fiuba/fiuba_2_postulantes
print("###", end='');
df_vistas_general = pd.read_csv('../csv/datos_navent_fiuba/fiuba_3_vistas.csv')
print("###", end='');
datetimes=["fechapostulacion"]
df_postulaciones = pd.read_csv('../csv/datos_navent_fiuba/fiuba_4_postulaciones.csv',
print("###", end='');
df_avisos_online = pd.read_csv('../csv/datos_navent_fiuba/fiuba_5_avisos_online.csv')
print("###", end='');
types= {"nombre_area": "category", "nivel_laboral": "category", "tipo_de_trabajo": "ca
df_avisos_detalle = pd.read_csv('../csv/datos_navent_fiuba/fiuba_6_avisos_detalle.csv'
print("###");
print("------------------");
```

```
------------------
##################
------------------
```

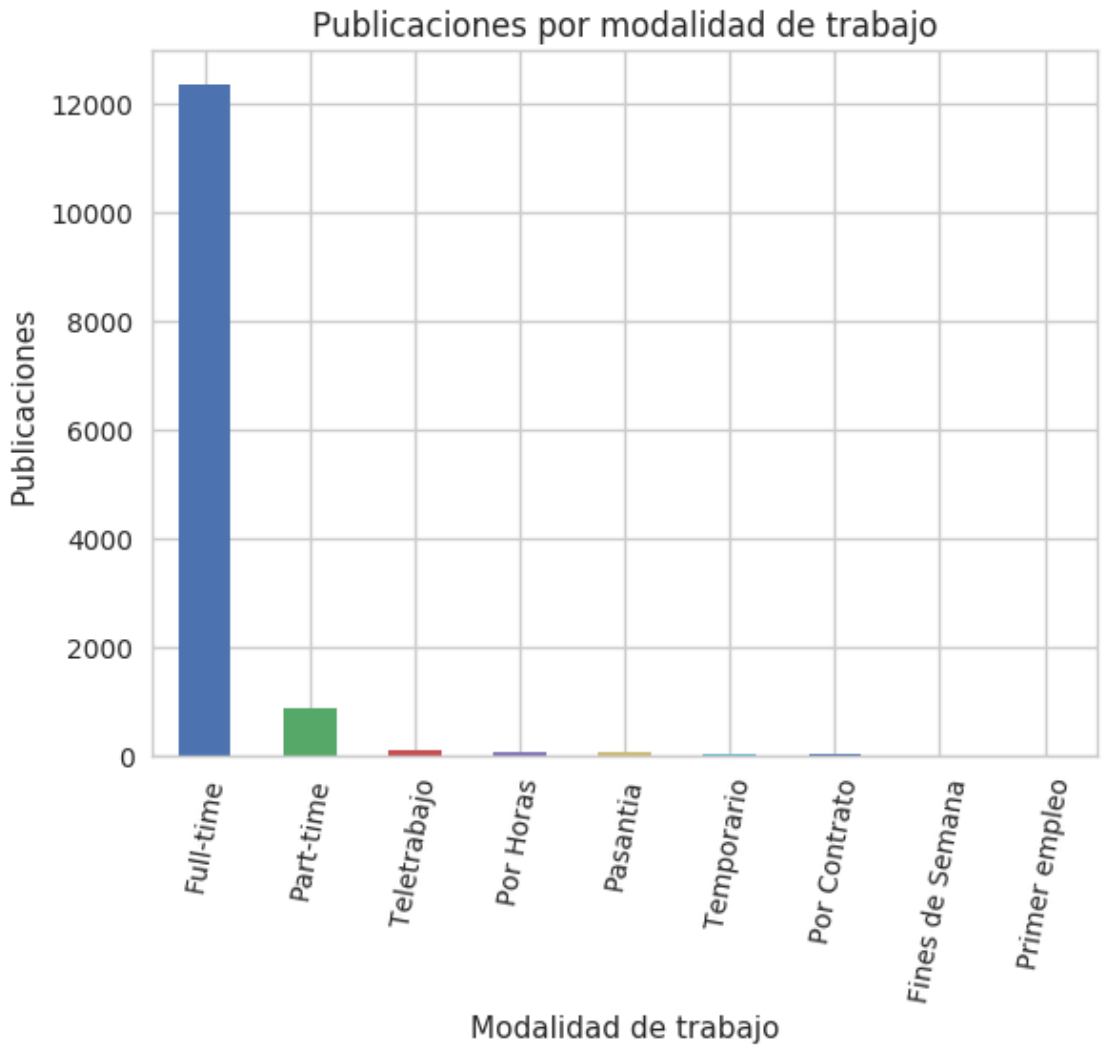
### 0.0.2 Publicaciones por categorias

```
In [3]: print (df_avisos_detalle['tipo_de_trabajo'].value_counts())
        print ("Cantidad de modalidades de trabajo: ",df_avisos_detalle['tipo_de_trabajo'].valu
```

```
Full-time          12339
Part-time            863
Teletrabajo          110
Por Horas             63
Pasantia              63
Temporario            42
Por Contrato          37
Fines de Semana       14
Primer empleo          3
Name: tipo_de_trabajo, dtype: int64
Cantidad de modalidades de trabajo:  9
```

### 0.0.3 Publicaciones por modalidad de trabajo (tiempo)

```
In [77]: df_por_tipo_de_trabajo = df_avisos_detalle['tipo_de_trabajo'].value_counts()
         df_por_tipo_de_trabajo.plot(kind='bar', rot=80, title='Publicaciones por modalidad de
         ax = plt.gca()
         ax.set_ylabel('Publicaciones');
         ax.set_xlabel('Modalidad de trabajo');
```

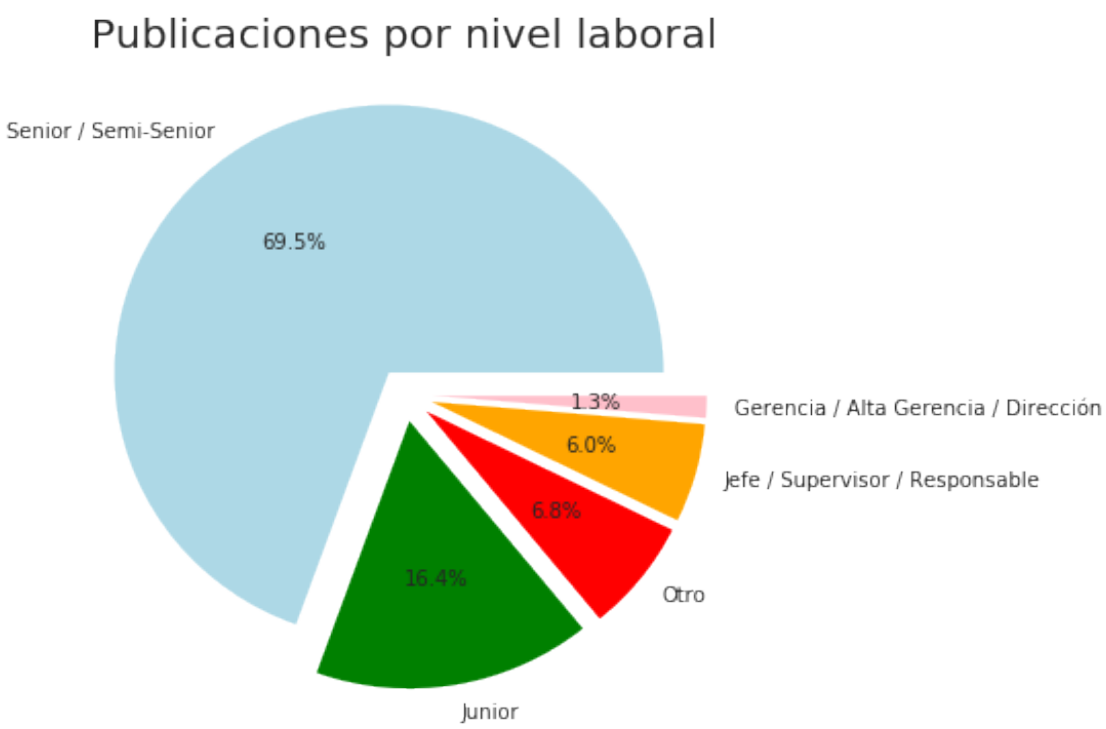



### 0.0.4 Cantidad de publicaciones por nivel laboral

```
In [5]: df_avisos_detalle['nivel_laboral'].value_counts()
        print ("Cantidad de niveles de trabajo: ",df_avisos_detalle['nivel_laboral'].value_cou

Cantidad de niveles de trabajo:  5
```

```
In [6]: labels='Senior / Semi-Senior', 'Junior', 'Otro', 'Jefe / Supervisor / Responsable', 'G
        explodes=0.1, 0.1, 0.1, 0.1, 0.1
        colors='lightblue', 'green', 'red', 'orange', 'pink'

        plt.figure(figsize=(6, 6))
        plt.title('Publicaciones por nivel laboral', fontsize=20)
        plt.pie(df_avisos_detalle['nivel_laboral'].value_counts(), labels=labels, autopct='%1.
```

```
    colors=colors,explode=explodes)
plt.show()
```



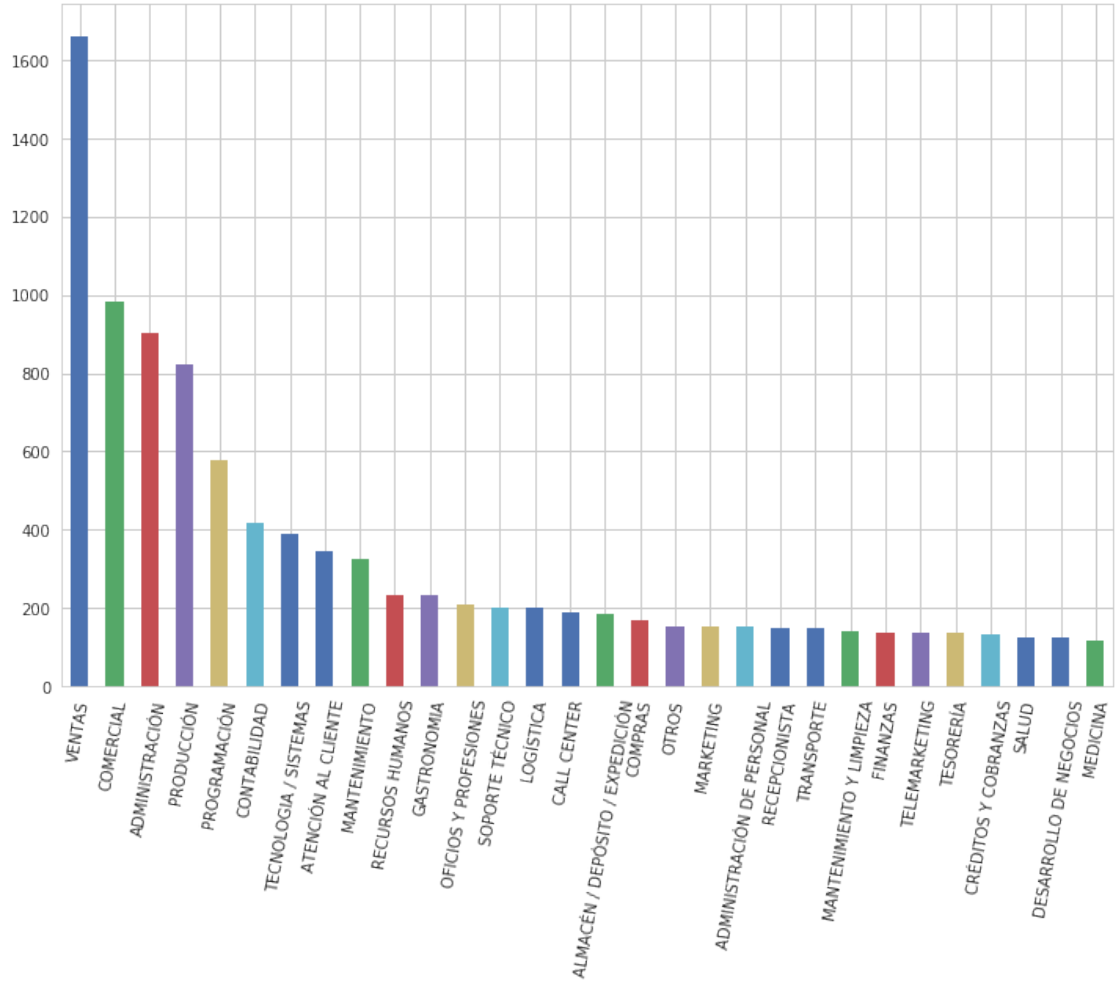

### 0.0.5 TOP 30 Áreas de trabajo con mas publicaciones

```
In [7]: df_avisos_detalle['nombre_area'].str.upper().value_counts().count()
        df_avisos_detalle['nombre_area'].str.upper().value_counts()[:30]
```

```
Out[7]: VENTAS                              1659
        COMERCIAL                            983
        ADMINISTRACIÓN                       901
        PRODUCCIÓN                           821
        PROGRAMACIÓN                         576
        CONTABILIDAD                         416
        TECNOLOGIA / SISTEMAS                388
        ATENCIÓN AL CLIENTE                  347
        MANTENIMIENTO                        324
        RECURSOS HUMANOS                     235
        GASTRONOMIA                          234
        OFICIOS Y PROFESIONES                209
        SOPORTE TÉCNICO                      203
        LOGÍSTICA                            200
        CALL CENTER                          191
        ALMACÉN / DEPÓSITO / EXPEDICIÓN      184
```

```
COMPRAS                          170
OTROS                            153
MARKETING                        153
ADMINISTRACIÓN DE PERSONAL       152
RECEPCIONISTA                    151
TRANSPORTE                       148
MANTENIMIENTO Y LIMPIEZA         141
FINANZAS                         138
TELEMARKETING                    138
TESORERÍA                        137
CRÉDITOS Y COBRANZAS             132
SALUD                            127
DESARROLLO DE NEGOCIOS           126
MEDICINA                         119
Name: nombre_area, dtype: int64
```

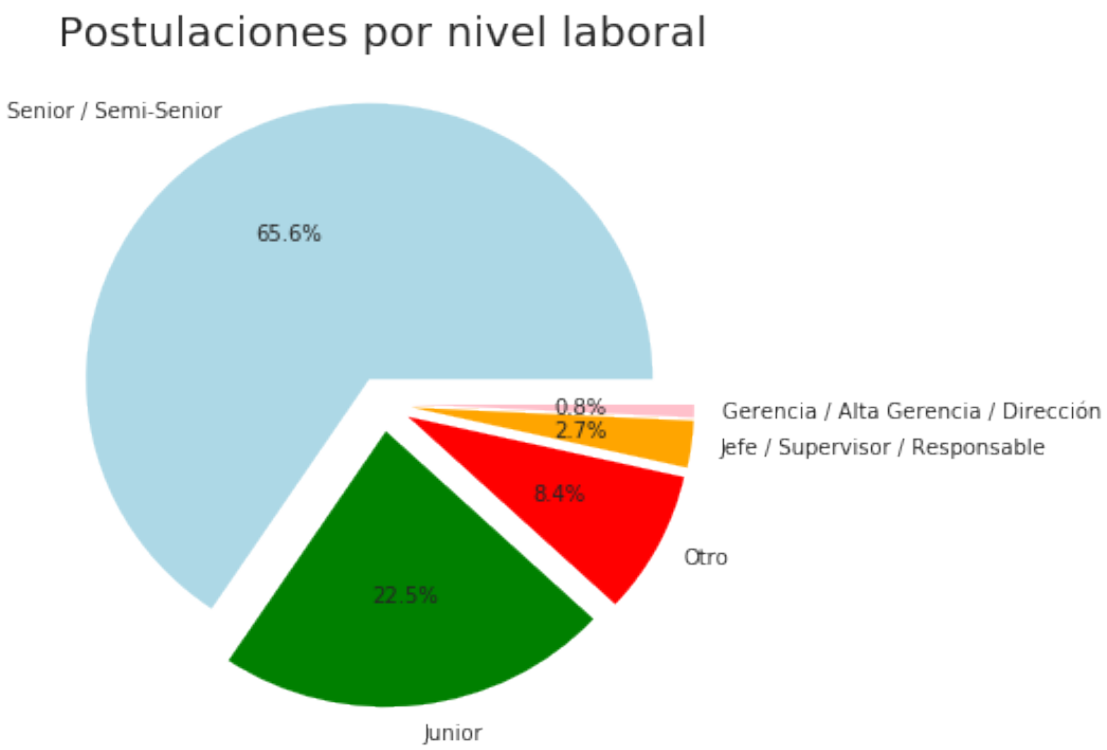
In [8]:
```
df_por_area_de_trabajo = df_avisos_detalle['nombre_area'].str.upper().value_counts()[:
df_por_area_de_trabajo.plot(kind='bar',rot=80,figsize=(12,8));
```



### 0.0.6 Postulaciones por categorias

```
In [9]: df_postulaciones_avisos = pd.merge(df_postulaciones, df_avisos_detalle, on='idaviso', h
```

```
In [10]: labels='Senior / Semi-Senior', 'Junior', 'Otro', 'Jefe / Supervisor / Responsable', 'G
         explodes=0.1, 0.1, 0.1, 0.1, 0.1
         colors='lightblue', 'green', 'red', 'orange', 'pink'

         plt.figure(figsize=(6, 6))
         plt.title('Postulaciones por nivel laboral', fontsize=20)
         plt.pie(df_postulaciones_avisos['nivel_laboral'].value_counts(), labels=labels, autopc
             colors=colors,explode=explodes)
         plt.show()
```



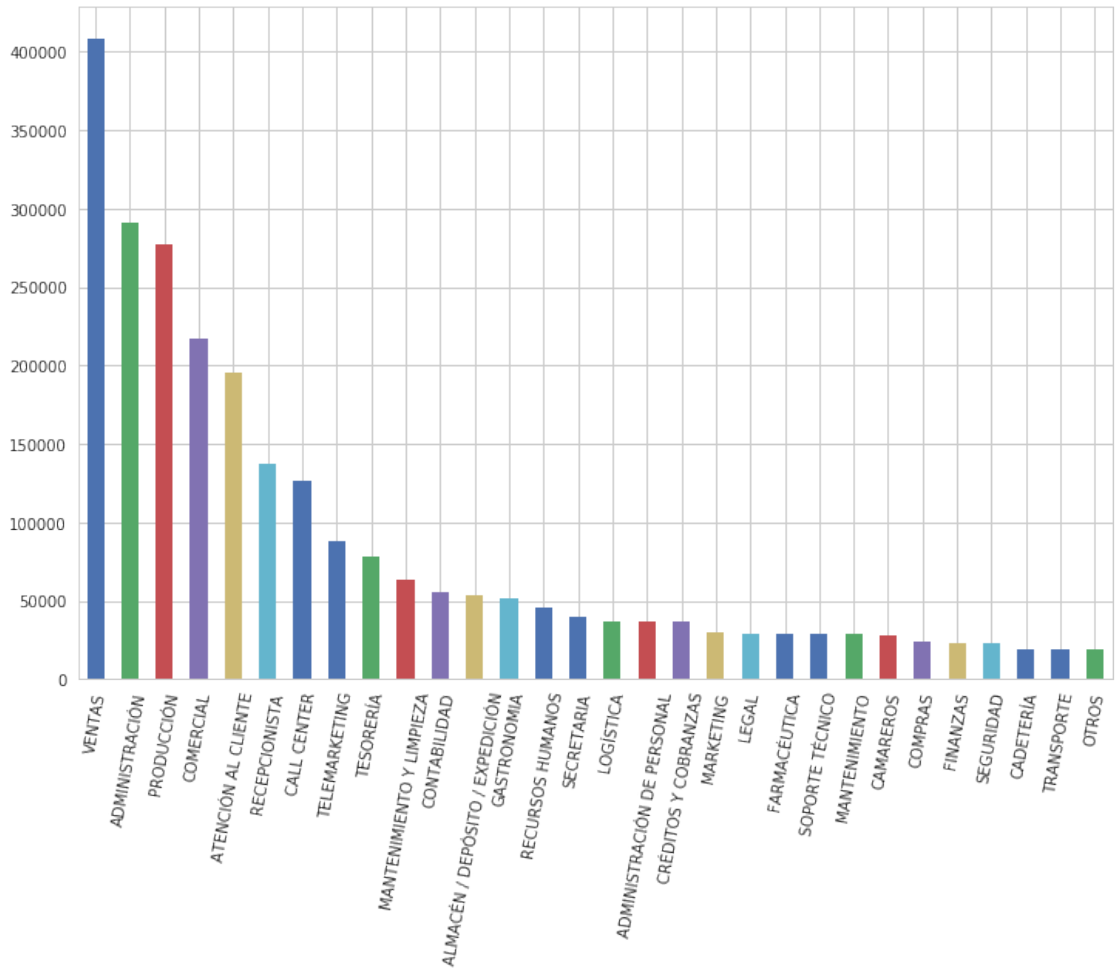

### 0.0.7 TOP 30 Áreas con mas postulaciones

```
In [11]: df_postulaciones_avisos['nombre_area'].str.upper().value_counts()[:30]
```

```
Out[11]: VENTAS                          408148
         ADMINISTRACIÓN                  291135
         PRODUCCIÓN                      277089
```

```
COMERCIAL                          216677
ATENCIÓN AL CLIENTE                195636
RECEPCIONISTA                      137485
CALL CENTER                        126430
TELEMARKETING                       87506
TESORERÍA                           78450
MANTENIMIENTO Y LIMPIEZA            63308
CONTABILIDAD                        55563
ALMACÉN / DEPÓSITO / EXPEDICIÓN     53261
GASTRONOMIA                         51213
RECURSOS HUMANOS                    45668
SECRETARIA                          39188
LOGÍSTICA                           37139
ADMINISTRACIÓN DE PERSONAL          37011
CRÉDITOS Y COBRANZAS                36690
MARKETING                           29861
LEGAL                               29295
FARMACÉUTICA                        29241
SOPORTE TÉCNICO                     29048
MANTENIMIENTO                       28462
CAMARAROS                           27482
COMPRAS                             23921
FINANZAS                            23206
SEGURIDAD                           22429
CADETERÍA                           19290
TRANSPORTE                          18908
OTROS                               18669
Name: nombre_area, dtype: int64
```

In [12]:
```
df_postulaciones_avisos['nombre_area'].str.upper().value_counts()[:30] \
    .plot(kind='bar',rot=80,figsize=(12,8));
```




```
In [13]: from wordcloud import WordCloud, STOPWORDS
         stopwords = set(STOPWORDS)
         stopwords = stopwords | {'a', 'ante', 'bajo', 'con', 'de', 'desde', 'durante',
                                  'en', 'entre', 'excepto', 'hacia', 'hasta', 'mediante',
                                  'para', 'por', 'salvo', 'según', 'sin', 'sobre', 'y', 'tras'
                                 'el', 'la', 'lo', 'su', 'un', 'una', 'que', 'al'}
```

#### 0.0.8 PreProcesamiento

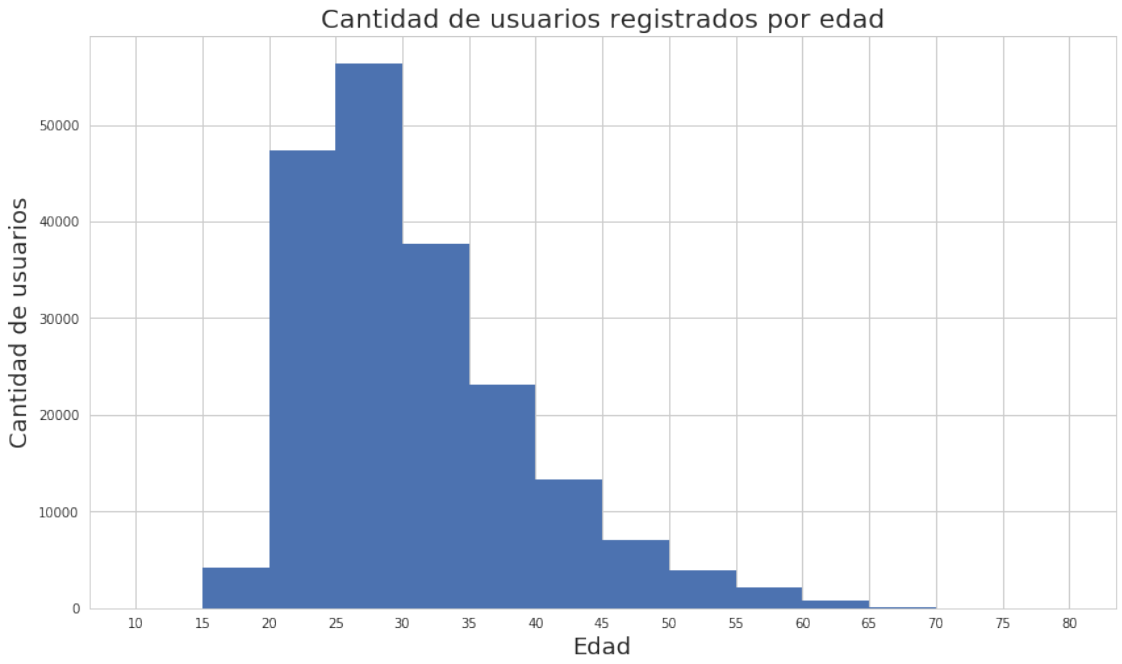
```
In [14]: df_postulantes_genero_edad['edad'] = (pd.to_datetime('today').year - pd.to_datetime(d
```

---

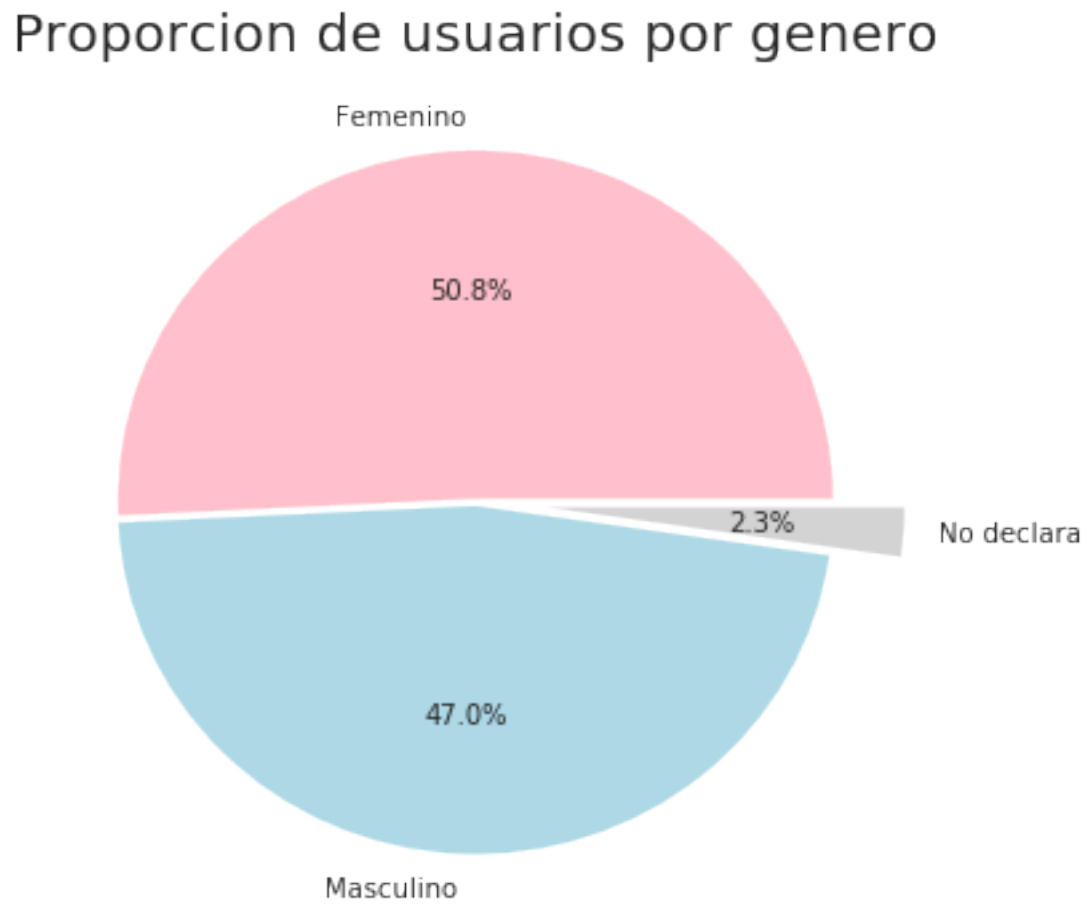
## 0.1 Cantidad de usuarios registrados por edad y genero

```
In [15]: df_postulantes_genero_edad['edad'].dropna().hist(figsize=(14,8), bins=14, range=[10, 8
         plt.xlabel('Edad', fontsize=18);
```

```
plt.ylabel('Cantidad de usuarios', fontsize=18)
plt.title('Cantidad de usuarios registrados por edad', fontsize=20);
plt.xticks(range(10, 81, 5))
plt.show();
```




In [16]:
```
plt.figure(figsize=(6, 6))
plt.title('Proporcion de usuarios por genero', fontsize=20)
plt.pie(df_postulantes_genero_edad['sexo'].value_counts(),
        labels=['Femenino', 'Masculino', 'No declara'], autopct='%1.1f%%',
        startangle=0,
        colors=['pink', 'lightblue', 'lightgray'],
        explode=(0.01, 0.01, 0.2))
plt.show()
```




```
In [17]: # TO DO: Stacked hist con usuarios por genero y edad
```

---

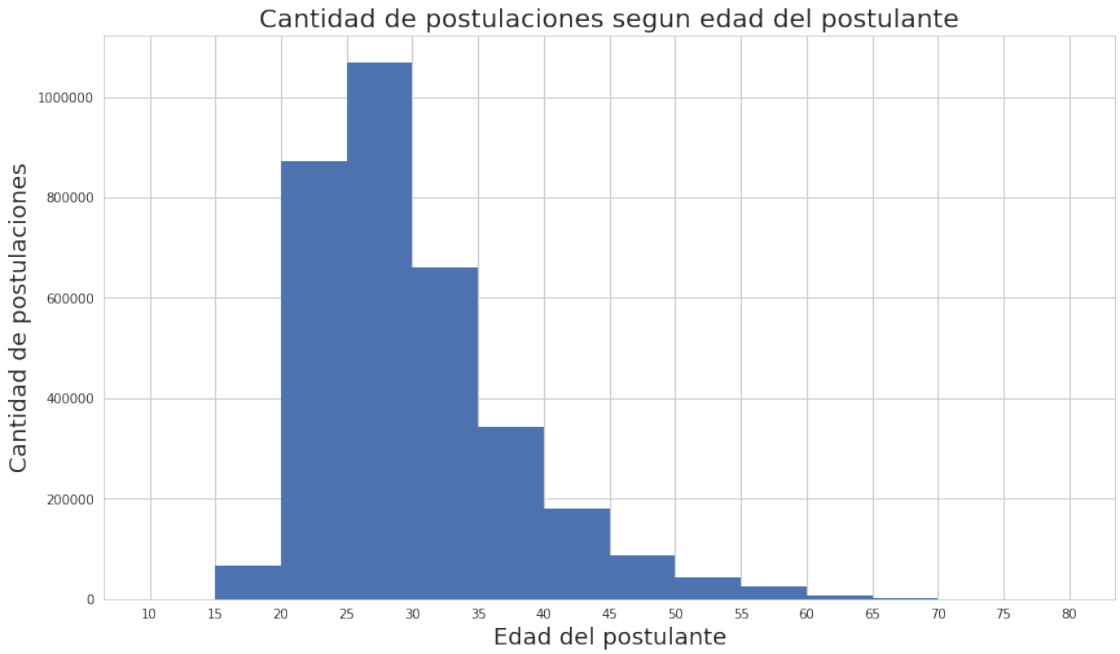
## 0.2 Postulaciones de trabajo según rango de edad y genero

```
In [18]: df_temp = pd.merge(df_postulantes_genero_edad, df_postulaciones,
                           on='idpostulante', how='inner')
         df_temp.head()
```

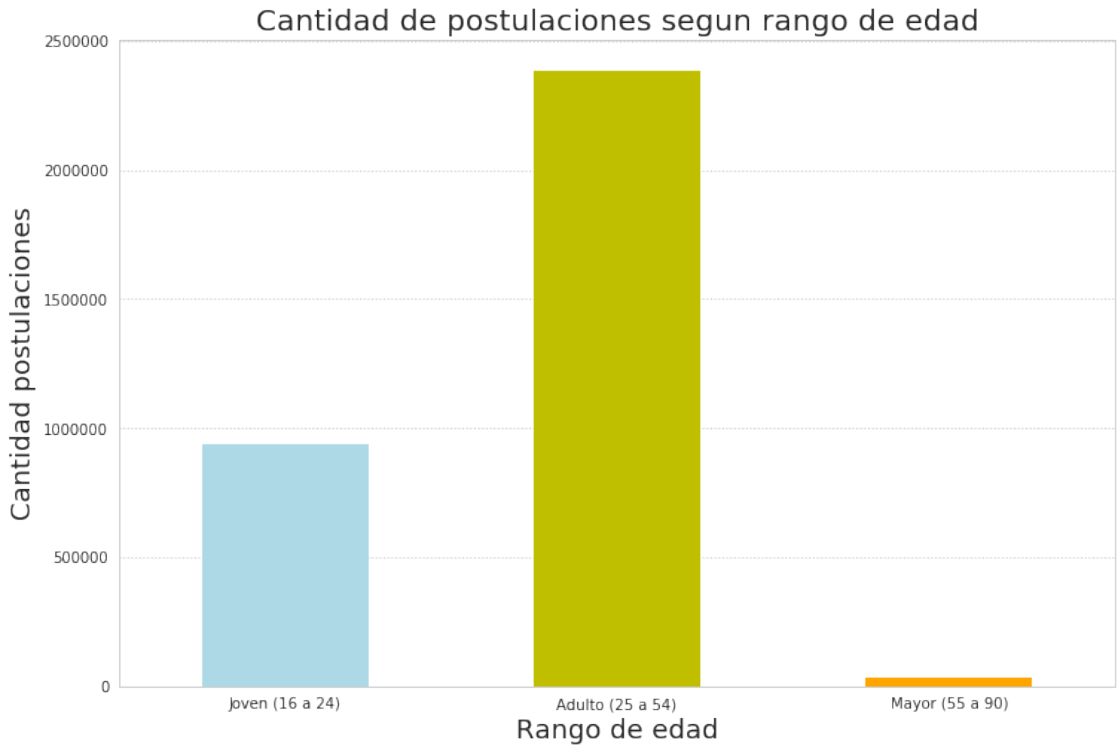
```
Out[18]:    Unnamed: 0 idpostulante fechanacimiento sexo  edad     idaviso  \
         0           0         NM5M      1970-12-03  FEM  48.0  1112257047
         1           0         NM5M      1970-12-03  FEM  48.0  1111920714
         2           0         NM5M      1970-12-03  FEM  48.0  1112346945
         3           0         NM5M      1970-12-03  FEM  48.0  1112345547
         4           1         5awk      1962-12-04  FEM  56.0  1112237522

              fechapostulacion
```

```
0 2018-01-15 16:22:34
1 2018-02-06 09:04:50
2 2018-02-22 09:04:47
3 2018-02-22 09:04:59
4 2018-01-25 18:55:03
```

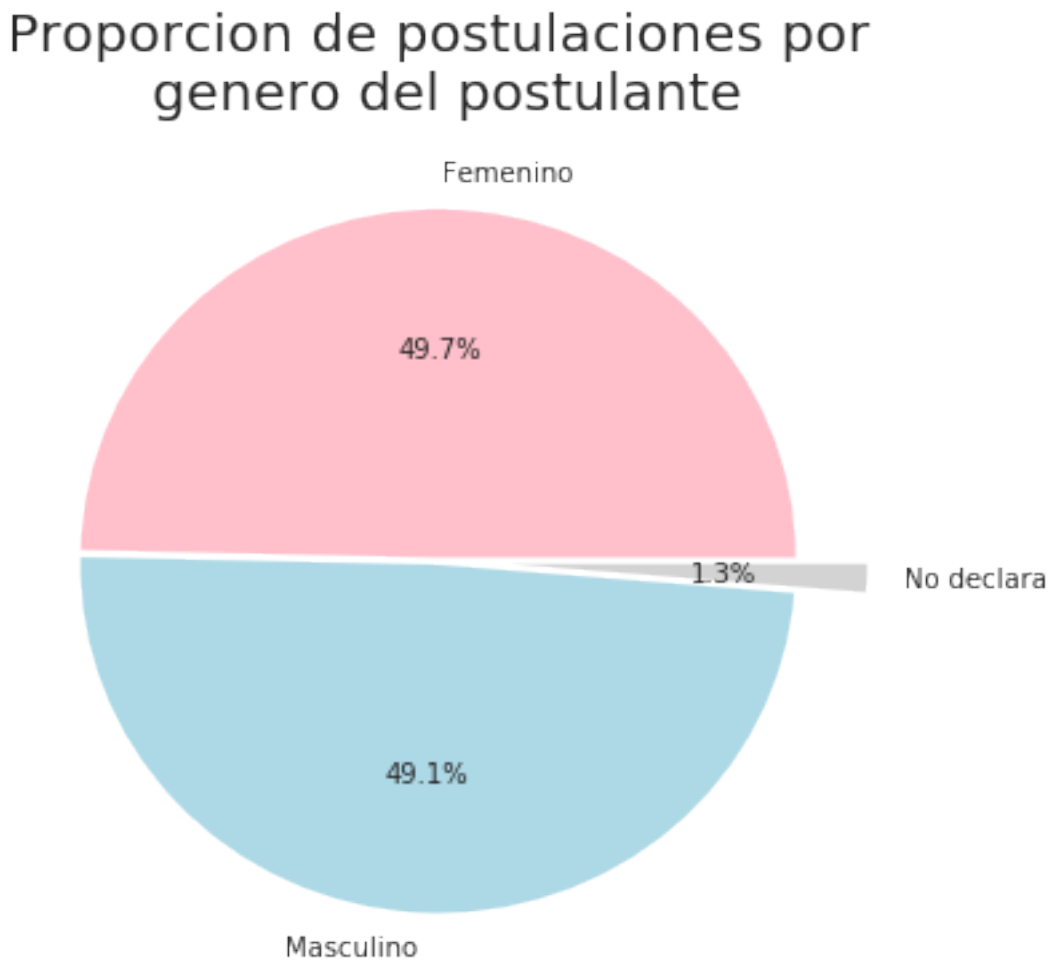
In [19]:
```
df_temp['edad'].dropna().hist(figsize=(14,8), bins=14, range=[10, 80]);
plt.xlabel('Edad del postulante', fontsize=18);
plt.ylabel('Cantidad de postulaciones', fontsize=18)
plt.title('Cantidad de postulaciones segun edad del postulante', fontsize=20);
plt.xticks(range(10, 81, 5))
plt.show();
```




In [20]:
```
res_temp = pd.Series([(df_temp['edad'].dropna().between(16, 24, inclusive=True)).sum()
                      (df_temp['edad'].dropna().between(25, 54, inclusive=True)).sum(),
                      (df_temp['edad'].dropna().between(55, 90, inclusive=True)).sum()])
col = ['lightblue','y','Orange']
plot = res_temp.plot(kind='bar', figsize=(12,8), rot=0, color=col)
plot.set_xlabel('Rango de edad', fontsize=18)
plot.set_ylabel('Cantidad postulaciones', fontsize=18)
plot.set_title('Cantidad de postulaciones segun rango de edad', fontsize=20)
plot.set_xticklabels( ('Joven (16 a 24)', 'Adulto (25 a 54)','Mayor (55 a 90)'))
plot.grid(linestyle='dotted')
plot.xaxis.grid(False);
```



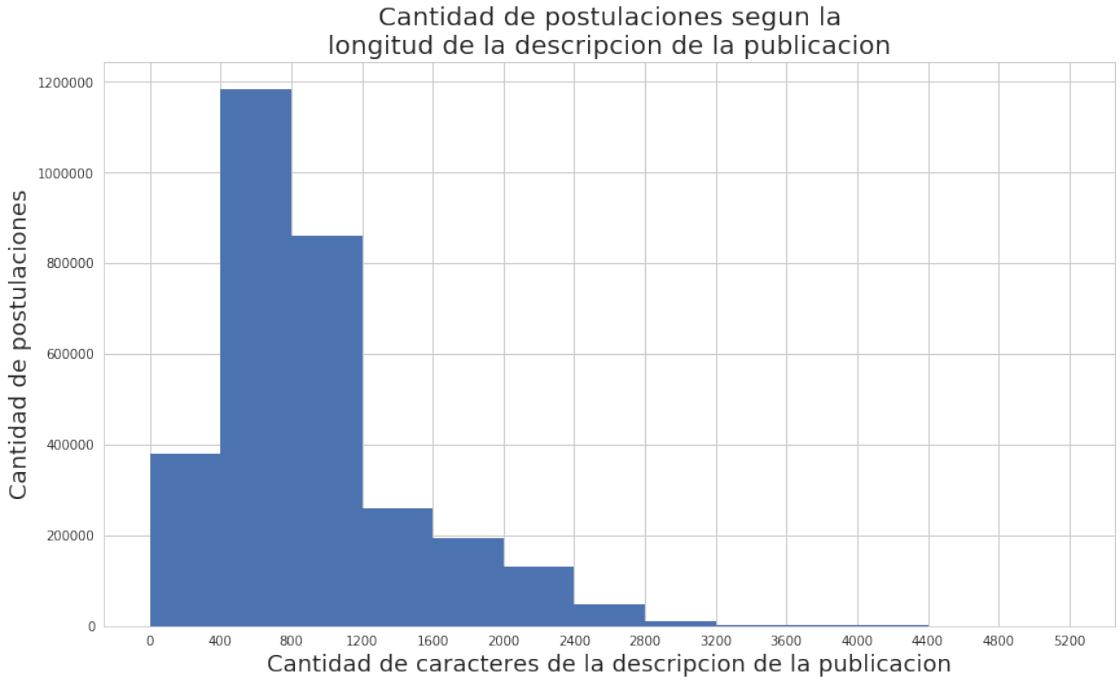

```
In [21]: plt.figure(figsize=(6, 6))
         plt.title('Proporcion de postulaciones por\n genero del postulante', fontsize=20)
         plt.pie(df_temp['sexo'].value_counts(),
                 labels=['Femenino', 'Masculino', 'No declara'], autopct='%1.1f%%',
                 startangle=0,
                 colors=['pink', 'lightblue', 'lightgray'],
                 explode=(0.01, 0.01, 0.2))
         plt.show()
```



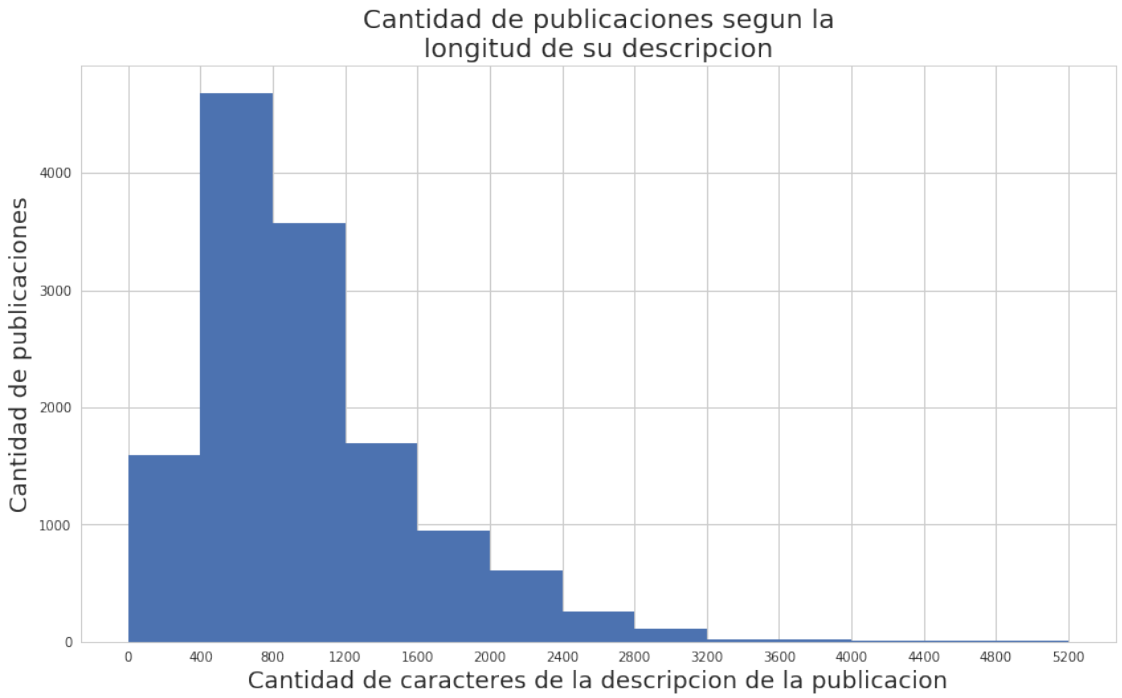

### 0.3 Publicaciones y Postulaciones según longitud de la descripción

A la descripcion se le quitan las etiquetas HTML porque no determinan la longitud del **contenido** de la descripcion

```
In [22]: df_temp = df_avisos_detalle.copy()
         df_temp['descripcion'] = df_temp['descripcion'].replace('<[^>]*>', '', regex=True)
         df_temp = pd.merge(df_temp, df_postulaciones,
                  on='idaviso', how='inner')
         df_temp['descripcion'].dropna().apply(len).hist(
             figsize=(14,8), bins=13, range=[0, 5201]);
         plt.xlabel('Cantidad de caracteres de la descripcion de la publicacion', fontsize=18)
         plt.ylabel('Cantidad de postulaciones', fontsize=18)
         plt.title('Cantidad de postulaciones segun la\nlongitud de la descripcion de la publi
         plt.xticks(range(0, 5201, 400))
         plt.show();
```

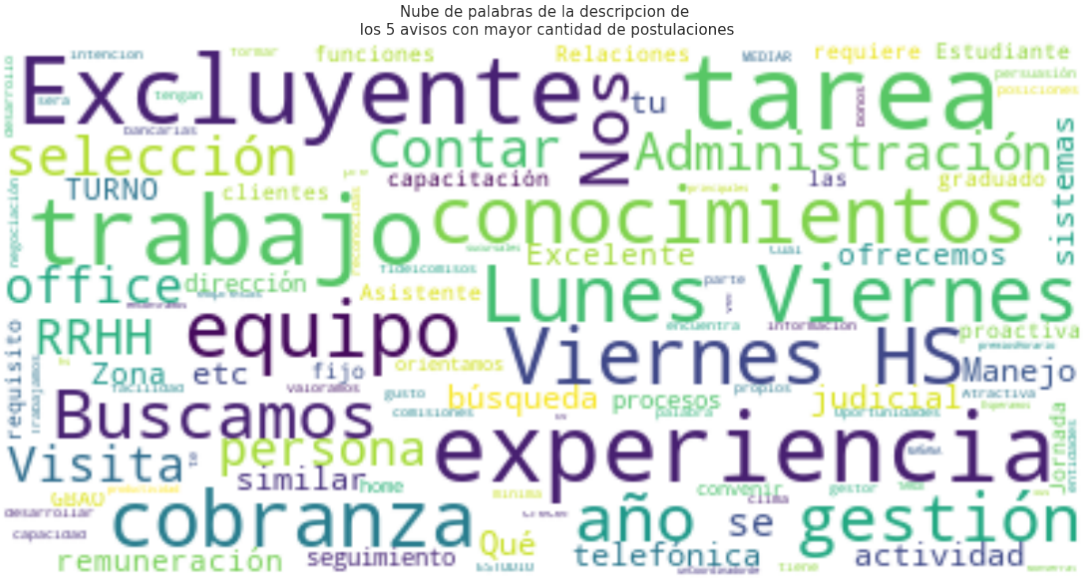



```
In [23]: df_avisos_detalle['descripcion'].dropna().replace('<[^>]*>', '', regex=True).apply(le
             figsize=(14,8), bins=13, range=[0, 5201]);
         plt.xlabel('Cantidad de caracteres de la descripcion de la publicacion', fontsize=18)
         plt.ylabel('Cantidad de publicaciones', fontsize=18)
         plt.title('Cantidad de publicaciones segun la\nlongitud de su descripcion', fontsize=
         plt.xticks(range(0, 5201, 400))
         plt.show();
```




## 0.4 Frecuencia de palabras en titulos y descripciones

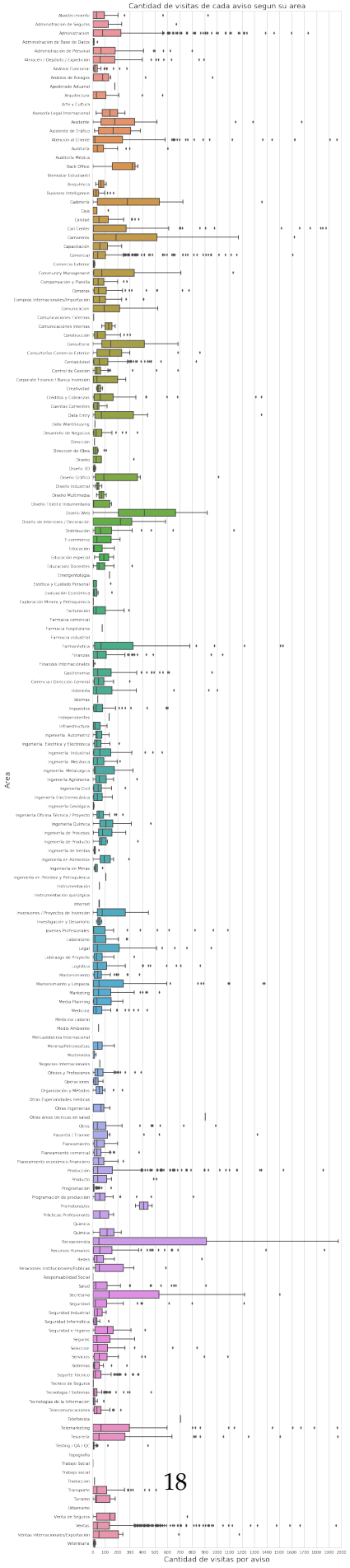
```
In [24]: arr_temp = pd.merge(df_avisos_detalle, df_postulaciones.loc[df_postulaciones['idaviso
                    on='idaviso', how='inner')['descripcion']
         arr_temp = arr_temp.replace('<[^>]*>', '', regex=True).replace({'\xa0': ''}, regex=Tru

         wordcloud = WordCloud(
                                    background_color='white',
                                    stopwords=stopwords,
                                    max_words=150,
                                    max_font_size=40,
                                    random_state=42
                                   ).generate(' '.join(arr_temp.values))

         plt.figure(figsize = (30,10))
         plt.imshow(wordcloud, interpolation="bilinear")
         plt.axis('off')
         plt.title('Nube de palabras de la descripcion de\n los 5 avisos con mayor cantidad de
         plt.show()
```



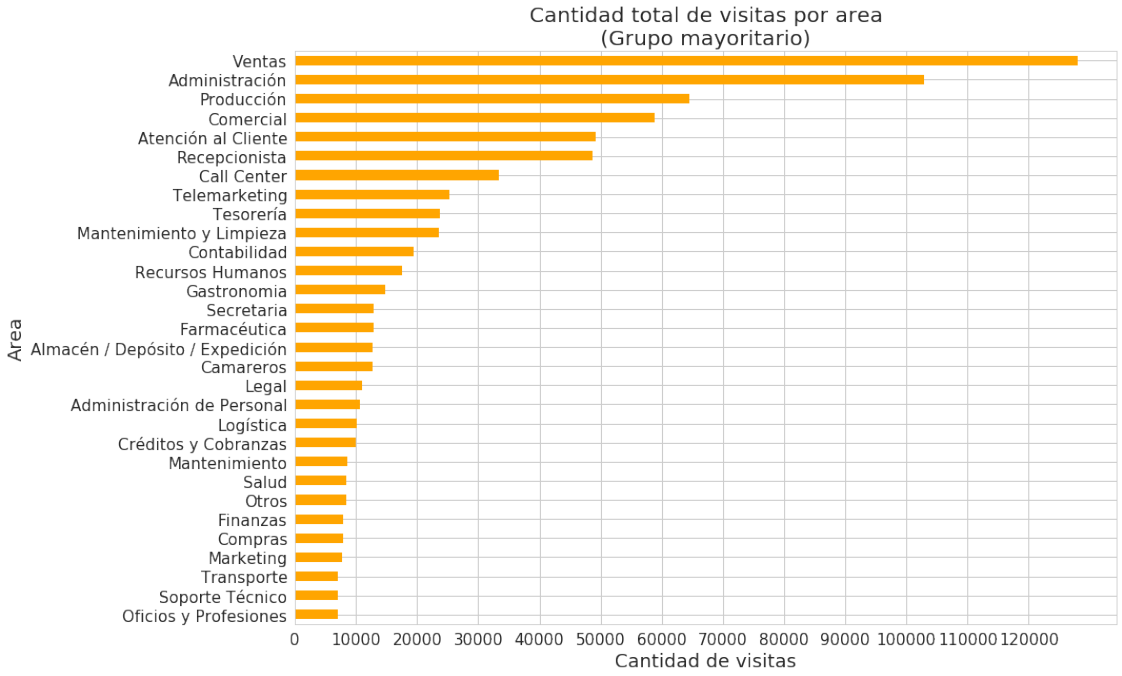

```
In [25]: arr_temp = pd.merge(df_avisos_detalle, df_vistas_general.loc[df_vistas_general['idAvis
                    left_on='idaviso', right_on='idAviso', how='inner')['titulo']

         wordcloud = WordCloud(
                                    background_color='white',
                                    stopwords=stopwords,
                                    max_words=150,
                                    max_font_size=40,
                                    random_state=42
                                   ).generate(' '.join(arr_temp.values))

         plt.figure(figsize = (30,10))
         plt.imshow(wordcloud, interpolation="bilinear")
         plt.axis('off')
         plt.title('Nube de palabras del titulo de\n los 20 avisos con mayor cantidad de visita
         plt.show()
```

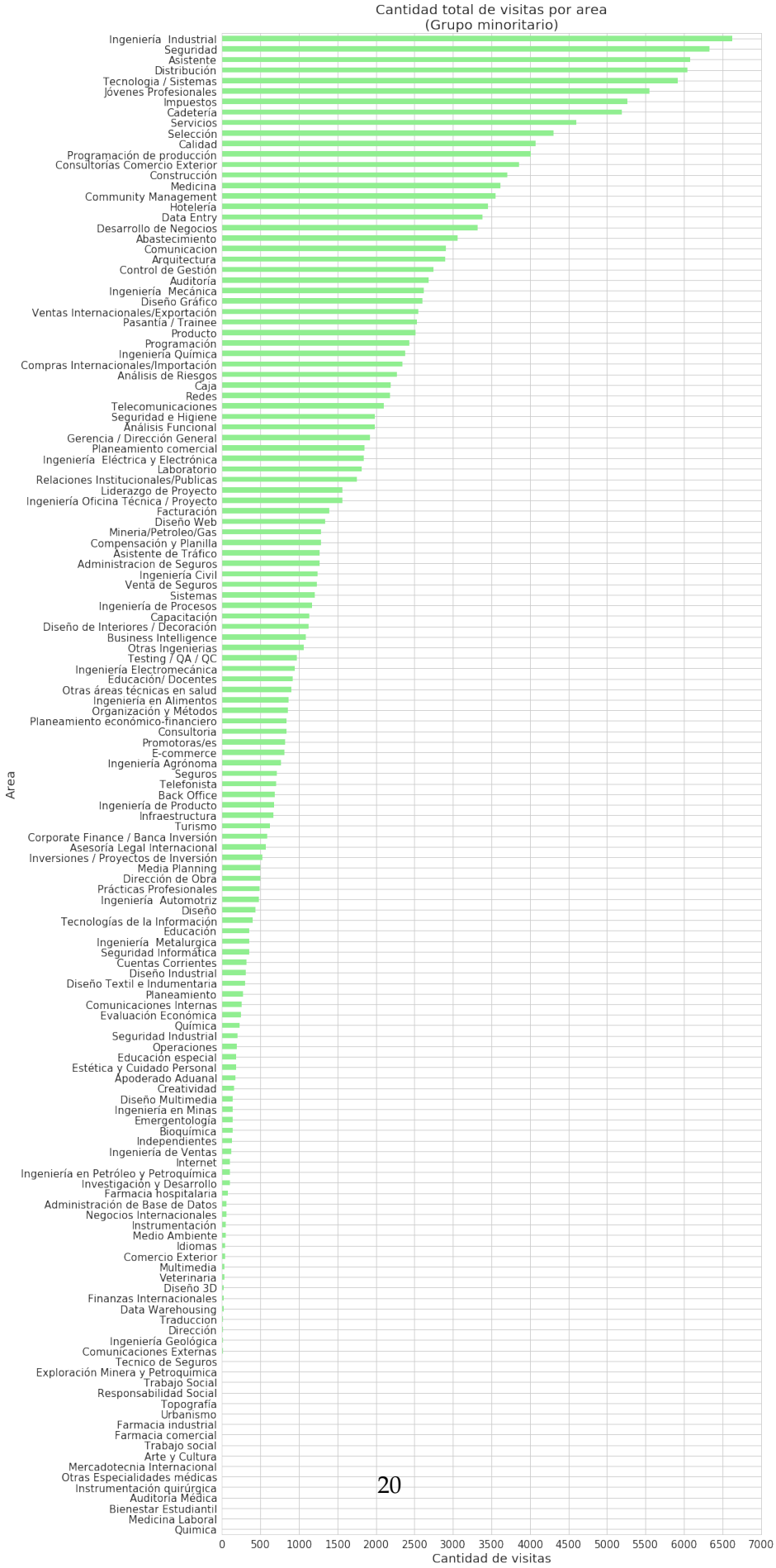

## 0.5 Visitas de avisos por area

```
In [26]: df_temp = pd.merge(df_avisos_detalle, df_vistas_general,
                            left_on='idaviso', right_on='idAviso', how='inner')[['idaviso', 'nombre_area

         df_temp['cant'] = df_temp.groupby('idaviso')['idaviso'].transform('count')
         df_temp.drop_duplicates(inplace=True)
         df_temp.sort_values(by='nombre_area', inplace=True)

         plt.figure(figsize = (11,70))
         plot = sns.boxplot(y='nombre_area',x='cant',data=df_temp, )
         plot.set_xlim([0, 1000])
         plot.set_ylabel('Area',size=18)
         plot.set_xlabel('Cantidad de visitas por aviso',size=18)
         plot.set_title('Cantidad de visitas de cada aviso segun su area',fontsize=18)
         plot.set_xticks(range(0, 2001, 100))
         plt.show();
```


Cantidad de visitas de cada aviso segun su area
Area
Cantidad de visitas por aviso
0
100
200
300
400
500
600
700
800
900
1000
1100
1200
1300
1400
1500
1600
1700
1800
1900
2000

```
In [27]: plot = df_temp.groupby('nombre_area').sum()['cant'].sort_values().tail(30).plot.barh(
         plot.set_title("Cantidad total de visitas por area\n (Grupo mayoritario) ", fontsize=
         plot.set_ylabel("Area", fontsize=18)
         plot.set_xlabel("Cantidad de visitas", fontsize=18)
         plot.set_xticks(range(0, 120001, 10000))
         plt.show()

         plot = df_temp.groupby('nombre_area').sum()['cant'].sort_values()[:-30].plot.barh(figs
         plot.set_title("Cantidad total de visitas por area\n (Grupo minoritario) ", fontsize=
         plot.set_ylabel("Area", fontsize=18)
         plot.set_xlabel("Cantidad de visitas", fontsize=18)
         plot.set_xticks(range(0, 7001, 500))
         plt.show()
```




Cantidad total de visitas por area
(Grupo minoritario)
Area
Ingeniería Industrial
Seguridad
Asistente
Distribución
Tecnologia / Sistemas
Jóvenes Profesionales
Impuestos
Cadetería
Servicios
Selección
Calidad
Programación de producción
Consultorías Comercio Exterior
Construcción
Medicina
Community Management
Hotelería
Data Entry
Desarrollo de Negocios
Abastecimiento
Comunicacion
Arquitectura
Control de Gestión
Auditoría
Ingeniería Mecánica
Diseño Gráfico
Ventas Internacionales/Exportación
Pasantía / Trainee
Producto
Programación
Ingeniería Química
Compras Internacionales/Importación
Análisis de Riesgos
Caja
Redes
Telecomunicaciones
Seguridad e Higiene
Análisis Funcional
Gerencia / Dirección General
Planeamiento comercial
Ingeniería Eléctrica y Electrónica
Laboratorio
Relaciones Institucionales/Publicas
Liderazgo de Proyecto
Ingeniería Oficina Técnica / Proyecto
Facturación
Diseño Web
Mineria/Petroleo/Gas
Compensación y Planilla
Asistente de Tráfico
Administracion de Seguros
Ingeniería Civil
Venta de Seguros
Sistemas
Ingeniería de Procesos
Capacitación
Diseño de Interiores / Decoración
Business Intelligence
Otras Ingenierías
Testing / QA / QC
Ingeniería Electromecánica
Educación/ Docentes
Otras áreas técnicas en salud
Ingeniería en Alimentos
Organización y Métodos
Planeamiento económico-financiero
Consultoria
Promotoras/es
E-commerce
Ingeniería Agrónoma
Seguros
Telefonista
Back Office
Ingeniería de Producto
Infraestructura
Turismo
Corporate Finance / Banca Inversión
Asesoría Legal Internacional
Inversiones / Proyectos de Inversión
Media Planning
Dirección de Obra
Prácticas Profesionales
Ingeniería Automotriz
Diseño
Tecnologías de la Información
Educación
Ingeniería Metalurgica
Seguridad Informática
Cuentas Corrientes
Diseño Industrial
Diseño Textil e Indumentaria
Planeamiento
Comunicaciones Internas
Evaluación Económica
Química
Seguridad Industrial
Operaciones
Educación especial
Estética y Cuidado Personal
Apoderado Aduanal
Creatividad
Diseño Multimedia
Ingeniería en Minas
Emergentología
Bioquímica
Independientes
Ingeniería de Ventas
Internet
Ingeniería en Petróleo y Petroquímica
Investigación y Desarrollo
Farmacia hospitalaria
Administración de Base de Datos
Negocios Internacionales
Instrumentación
Medio Ambiente
Idiomas
Comercio Exterior
Multimedia
Veterinaria
Diseño 3D
Finanzas Internacionales
Data Warehousing
Traduccion
Dirección
Ingeniería Geológica
Comunicaciones Externas
Tecnico de Seguros
Exploración Minera y Petroquímica
Trabajo Social
Responsabilidad Social
Topografía
Urbanismo
Farmacia industrial
Farmacia comercial
Trabajo social
Arte y Cultura
Mercadotecnia Internacional
Otras Especialidades médicas
Instrumentación quirúrgica
Auditoría Médica
Bienestar Estudiantil
Medicina Laboral
Quimica
0
500
1000
1500
2000
2500
3000
3500
4000
4500
5000
5500
6000
6500
7000
Cantidad de visitas

# 1 Porcentajes de Avisos por nivel agrupados por Carga Horaria

A continuación se realizara un análisis de los avisos agrupandolos por Nivel requerido y la carga horaria solicitada para cada uno.

```
In [28]: avisos_x_nivel_y_carga = df_avisos_detalle.groupby(['tipo_de_trabajo', 'nivel_laboral
         avisos_x_nivel_y_carga
```

Out[28]:

| nivel_laboral | Gerencia / Alta Gerencia / Dirección | Jefe / Supervisor / Responsable | Junior | Otro | Senior / Semi-Senior |
|---|---|---|---|---|---|
| tipo_de_trabajo | | | | | |
| Fines de Semana | 0 | 0 | 1 | 3 | 10 |
| Full-time | 179 | 745 | 1777 | 751 | 8887 |
| Part-time | 1 | 7 | 350 | 132 | 373 |
| Pasantia | 0 | 0 | 49 | 10 | 4 |
| Por Contrato | 0 | 5 | 10 | 5 | 17 |
| Por Horas | 0 | 1 | 8 | 15 | 39 |
| Primer empleo | 0 | 0 | 2 | 1 | 0 |
| Teletrabajo | 1 | 50 | 1 | 1 | 57 |
| Temporario | 0 | 1 | 18 | 3 | 20 |

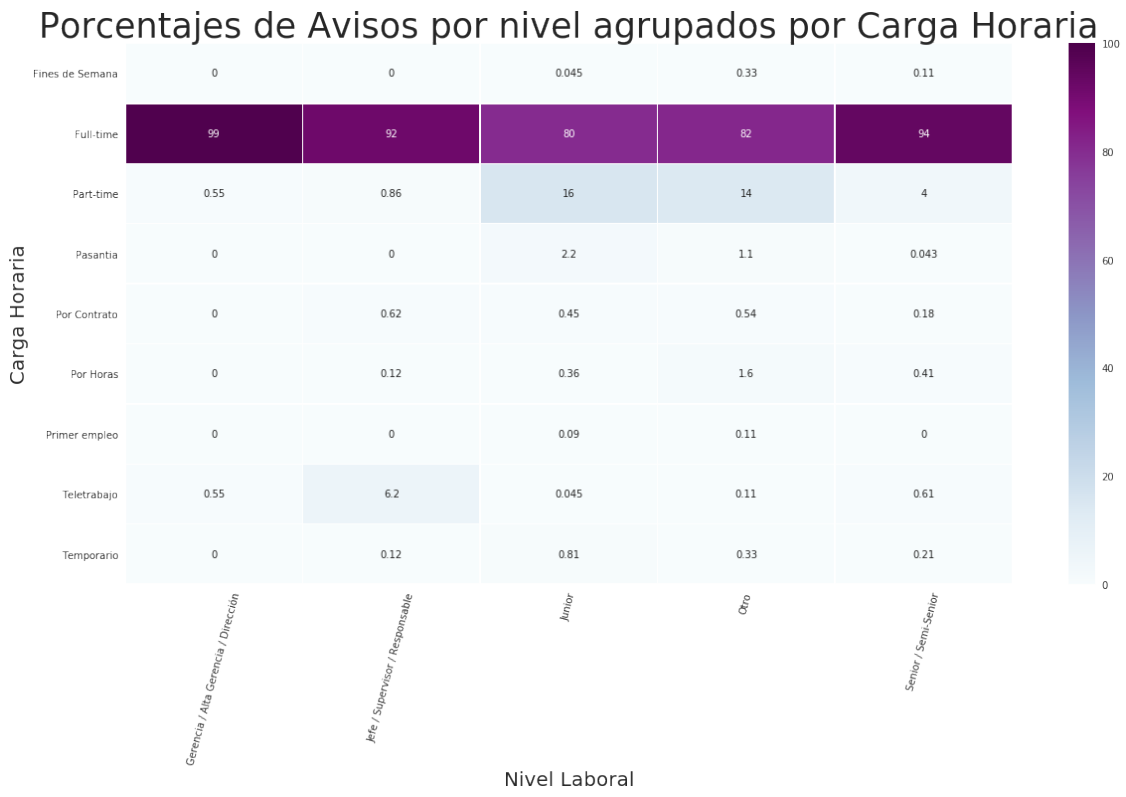
Como se puede observar hay una diferencia importante entre las solicitudes para trabajo Full-time con respecto al resto. A fin de poder evaluar como es la distribución de Cargas solicitadas

con respecto a cada nivel solicitado, lo que se hará es calcular el porcentaje para cada

```
In [29]: avisos_nivel_tipo_carga = avisos_x_nivel_y_carga.apply(lambda x: 100 * x / float(x.sum
         avisos_nivel_tipo_carga
```

```
Out[29]: nivel_laboral    Gerencia / Alta Gerencia / Dirección  \
         tipo_de_trabajo
         Fines de Semana                                 0.000
         Full-time                                      98.895
         Part-time                                       0.552
         Pasantia                                        0.000
         Por Contrato                                    0.000
         Por Horas                                       0.000
         Primer empleo                                   0.000
         Teletrabajo                                     0.552
         Temporario                                      0.000

         nivel_laboral    Jefe / Supervisor / Responsable  Junior    Otro  \
         tipo_de_trabajo
         Fines de Semana                            0.000   0.045   0.326
         Full-time                                 92.089  80.190  81.542
         Part-time                                  0.865  15.794  14.332
         Pasantia                                   0.000   2.211   1.086
         Por Contrato                               0.618   0.451   0.543
         Por Horas                                  0.124   0.361   1.629
         Primer empleo                              0.000   0.090   0.109
         Teletrabajo                                6.180   0.045   0.109
         Temporario                                 0.124   0.812   0.326

         nivel_laboral    Senior / Semi-Senior
         tipo_de_trabajo
         Fines de Semana                 0.106
         Full-time                      94.472
         Part-time                       3.965
         Pasantia                        0.043
         Por Contrato                    0.181
         Por Horas                       0.415
         Primer empleo                   0.000
         Teletrabajo                     0.606
         Temporario                      0.213
```

```
In [30]: plt.figure(figsize = (20,10))
         avisos_heatMap = sns.heatmap(avisos_nivel_tipo_carga,vmin=0,vmax=100, cmap='BuPu', lin
         avisos_heatMap.set_ylabel("Carga Horaria", fontsize = 20)
         avisos_heatMap.set_xlabel("Nivel Laboral", fontsize = 20)
         avisos_heatMap.set_title("Porcentajes de Avisos por nivel agrupados por Carga Horaria"
         avisos_heatMap.set_xticklabels(labels = avisos_nivel_tipo_carga.columns.values, rotati
```




### 1.1 Conclusión

Se puede observar que predomina los trabajos con dedicación Full-time, aunque en el caso de los cargos de nivel menor, existe una mayor variación.

## 2 Porcentajes de Avisos por carga horaria, agrupados por nivel

En este caso, evaluaremos a partir de los mismos datos, pero revisando como se distribuyen dentro de una misma carga horaria los niveles solicitados

```
In [31]: avisos_x_carga_y_nivel = df_avisos_detalle.groupby(['nivel_laboral','tipo_de_trabajo']
         avisos_x_carga_y_nivel
```

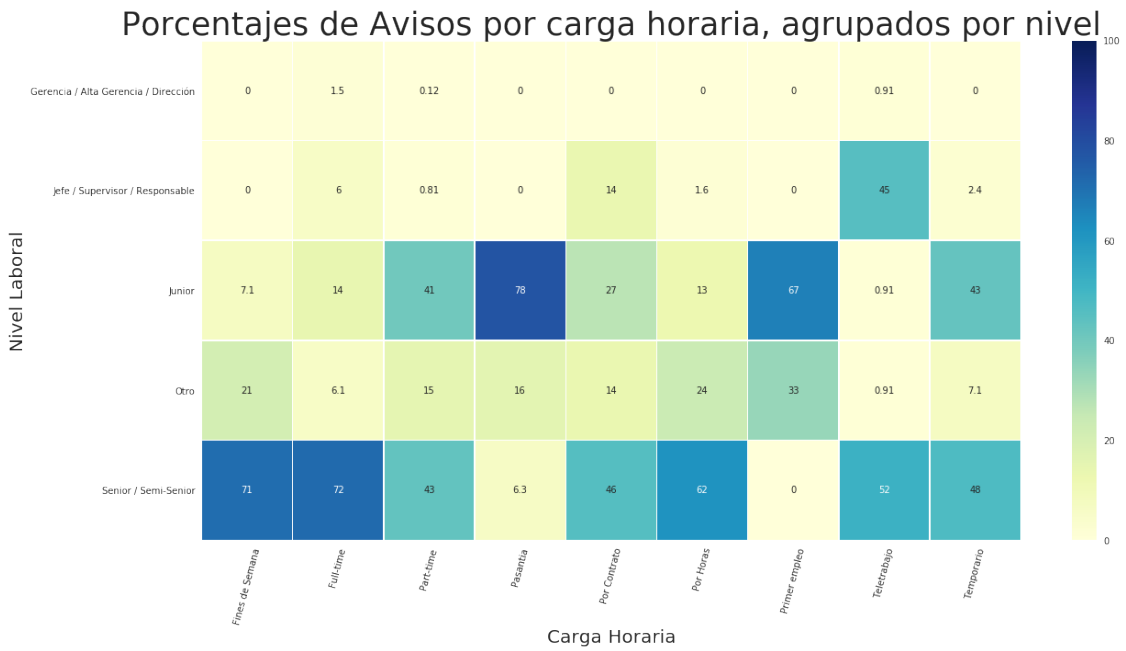
```
Out[31]: tipo_de_trabajo                       Fines de Semana  Full-time  Part-time  \
         nivel_laboral
         Gerencia / Alta Gerencia / Dirección                0        179          1
         Jefe / Supervisor / Responsable                     0        745          7
         Junior                                              1       1777        350
         Otro                                                3        751        132
         Senior / Semi-Senior                               10       8887        373

         tipo_de_trabajo                       Pasantia  Por Contrato  Por Horas  \
```

```
nivel_laboral
Gerencia / Alta Gerencia / Dirección         0             0          0
Jefe / Supervisor / Responsable              0             5          1
Junior                                      49            10          8
Otro                                        10             5         15
Senior / Semi-Senior                         4            17         39

tipo_de_trabajo                      Primer empleo  Teletrabajo  Temporario
nivel_laboral
Gerencia / Alta Gerencia / Dirección              0            1           0
Jefe / Supervisor / Responsable                   0           50           1
Junior                                            2            1          18
Otro                                              1            1           3
Senior / Semi-Senior                              0           57          20
```

In [32]:
```
avisos_nivel_carga_tipo = avisos_x_carga_y_nivel.apply(lambda x: 100 * x / float(x.sum
avisos_nivel_carga_tipo
```

Out[32]:
```
tipo_de_trabajo                      Fines de Semana  Full-time  Part-time  \
nivel_laboral
Gerencia / Alta Gerencia / Dirección            0.000      1.451      0.116
Jefe / Supervisor / Responsable                 0.000      6.038      0.811
Junior                                          7.143     14.401     40.556
Otro                                           21.429      6.086     15.295
Senior / Semi-Senior                           71.429     72.024     43.221

tipo_de_trabajo                      Pasantia  Por Contrato  Por Horas  \
nivel_laboral
Gerencia / Alta Gerencia / Dirección     0.000         0.000      0.000
Jefe / Supervisor / Responsable          0.000        13.514      1.587
Junior                                  77.778        27.027     12.698
Otro                                    15.873        13.514     23.810
Senior / Semi-Senior                     6.349        45.946     61.905

tipo_de_trabajo                      Primer empleo  Teletrabajo  Temporario
nivel_laboral
Gerencia / Alta Gerencia / Dirección          0.000        0.909       0.000
Jefe / Supervisor / Responsable               0.000       45.455       2.381
Junior                                       66.667        0.909      42.857
Otro                                         33.333        0.909       7.143
Senior / Semi-Senior                          0.000       51.818      47.619
```

In [33]:
```
plt.figure(figsize = (20,10))
avisos_heatMap = sns.heatmap(avisos_nivel_carga_tipo,vmin=0,vmax=100, cmap='YlGnBu', l
avisos_heatMap.set_xlabel("Carga Horaria", fontsize = 20)
avisos_heatMap.set_ylabel("Nivel Laboral", fontsize = 20)
avisos_heatMap.set_title("Porcentajes de Avisos por carga horaria, agrupados por nivel
avisos_heatMap.set_xticklabels(avisos_nivel_carga_tipo.columns.values, rotation=75);
```




### 2.1 Conclusion

Aquí se observa como para todos los tipos de contratación, como de acuerdo al mayor la exigencia de la carga, aumenta el nivel laboral solicitado.

## 3 Avisos publicados por Areas y Nivel

Para analizar este punto, se observan muchas areas que cuentan con pocos pedidos. Se filtraran primero para aquellas Areas, donde la cantidad de publicaciones supere a la media, obteniendo asi una muestra que nos permite observar la distribución por Area y el nivel que se solicita para cada una

```
In [34]: cant_x_area = df_avisos_detalle.groupby('nombre_area').size().reset_index(name='counts
         filtro_area_mayor_mean = cant_x_area[cant_x_area["counts"] > cant_x_area["counts"].mea
         aux = pd.merge(df_avisos_detalle,filtro_area_mayor_mean,on="nombre_area",how="inner")
         cant_x_nivel = aux.groupby(['nombre_area','nivel_laboral']).size().unstack(fill_value=
         cant_x_nivel
```

```
Out[34]: nombre_area                          Administración  \
         nivel_laboral
         Gerencia / Alta Gerencia / Dirección              13
         Jefe / Supervisor / Responsable                   33
         Junior                                           236
         Otro                                              49
         Senior / Semi-Senior                             570
```

| nombre_area | Administración de Personal | \ |
|---|---|---|
| nivel_laboral | | |
| Gerencia / Alta Gerencia / Dirección | 0 | |
| Jefe / Supervisor / Responsable | 8 | |
| Junior | 32 | |
| Otro | 3 | |
| Senior / Semi-Senior | 109 | |

| nombre_area | Almacén / Depósito / Expedición | \ |
|---|---|---|
| nivel_laboral | | |
| Gerencia / Alta Gerencia / Dirección | 1 | |
| Jefe / Supervisor / Responsable | 9 | |
| Junior | 34 | |
| Otro | 18 | |
| Senior / Semi-Senior | 122 | |

| nombre_area | Análisis Funcional | Atención al Cliente | \ |
|---|---|---|---|
| nivel_laboral | | | |
| Gerencia / Alta Gerencia / Dirección | 0 | 1 | |
| Jefe / Supervisor / Responsable | 0 | 14 | |
| Junior | 12 | 121 | |
| Otro | 2 | 35 | |
| Senior / Semi-Senior | 86 | 176 | |

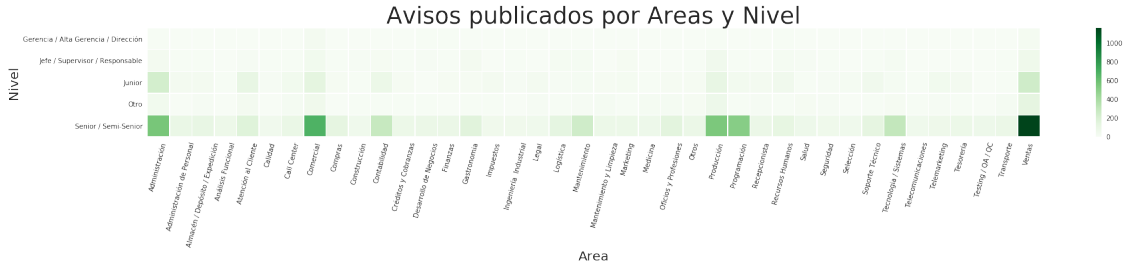
| nombre_area | Calidad | Call Center | Comercial | \ |
|---|---|---|---|---|
| nivel_laboral | | | | |
| Gerencia / Alta Gerencia / Dirección | 2 | 0 | 32 | |
| Jefe / Supervisor / Responsable | 12 | 7 | 49 | |
| Junior | 17 | 51 | 150 | |
| Otro | 4 | 27 | 63 | |
| Senior / Semi-Senior | 59 | 106 | 689 | |

| nombre_area | Compras | Construcción | ... | \ |
|---|---|---|---|---|
| nivel_laboral | | | ... | |
| Gerencia / Alta Gerencia / Dirección | 0 | 0 | ... | |
| Jefe / Supervisor / Responsable | 12 | 18 | ... | |
| Junior | 20 | 5 | ... | |
| Otro | 1 | 10 | ... | |
| Senior / Semi-Senior | 137 | 64 | ... | |

| nombre_area | Seguridad | Selección | Soporte Técnico | \ |
|---|---|---|---|---|
| nivel_laboral | | | | |
| Gerencia / Alta Gerencia / Dirección | 0 | 0 | 0 | |
| Jefe / Supervisor / Responsable | 5 | 0 | 5 | |
| Junior | 14 | 23 | 56 | |
| Otro | 16 | 2 | 5 | |
| Senior / Semi-Senior | 72 | 65 | 137 | |

```
nombre_area                          Tecnologia / Sistemas  \
nivel_laboral
Gerencia / Alta Gerencia / Dirección                     4
Jefe / Supervisor / Responsable                         27
Junior                                                  27
Otro                                                    21
Senior / Semi-Senior                                   309

nombre_area                          Telecomunicaciones  Telemarketing  \
nivel_laboral
Gerencia / Alta Gerencia / Dirección                   0              1
Jefe / Supervisor / Responsable                        3              0
Junior                                                11             48
Otro                                                   7             15
Senior / Semi-Senior                                  65             74

nombre_area                          Tesorería  Testing / QA / QC  \
nivel_laboral
Gerencia / Alta Gerencia / Dirección          0                  0
Jefe / Supervisor / Responsable               2                  2
Junior                                       33                  2
Otro                                          7                  2
Senior / Semi-Senior                         95                 79

nombre_area                          Transporte  Ventas
nivel_laboral
Gerencia / Alta Gerencia / Dirección           1      30
Jefe / Supervisor / Responsable                6      72
Junior                                        10     253
Otro                                          27     140
Senior / Semi-Senior                         104    1164

[5 rows x 40 columns]
```

In [35]:
```
plt.figure(figsize = (30,3))
avisos_heatMap = sns.heatmap(cant_x_nivel, cmap='Greens', linewidths=0.5)
avisos_heatMap.set_ylabel("Nivel", fontsize = 20)
avisos_heatMap.set_xlabel("Area", fontsize = 20)
avisos_heatMap.set_title("Avisos publicados por Areas y Nivel", fontsize = 35)
avisos_heatMap.set_xticklabels(labels = cant_x_nivel.columns.values, rotation=75);
```



## 3.1 Conclusión

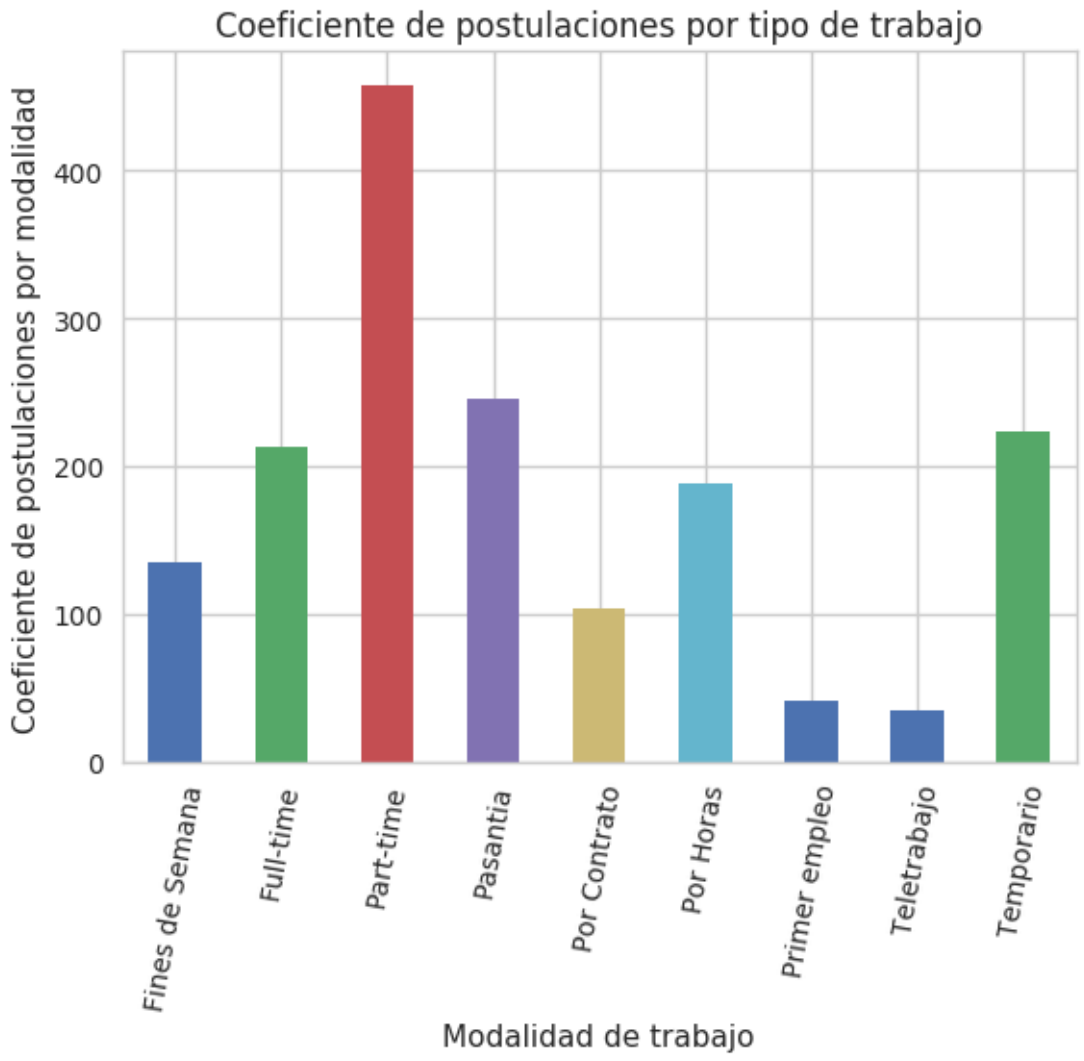
En este caso, se puede observar una distribucion mas pareja, asi como tamabien aquellas areas donde aumenta la exigencia en el nivel. La Diferencia entre las distintas áreas se muestra similar, salvo en el caso de las Ventas, donde se solicita un nivel mucho mayo que en el resto.

## 3.2 Relación entre publicaciones y postulaciones

```
In [36]: df_postulaciones_avisos = pd.merge(df_avisos_detalle,df_postulaciones, on="idaviso", h

In [37]: postulaciones_promedio_tipo_trabajo=df_postulaciones_avisos.groupby(["tipo_de_trabajo"
```

### 3.2.1 Relacion segun tipo de trabajo

```
In [69]: title = 'Coeficiente de postulaciones por tipo de trabajo'
         ax = postulaciones_promedio_tipo_trabajo.plot(kind='bar',rot=80,title=title)
         ax = plt.gca()
         ax.set_ylabel('Coeficiente de postulaciones por modalidad');
         ax.set_xlabel('Modalidad de trabajo');
```

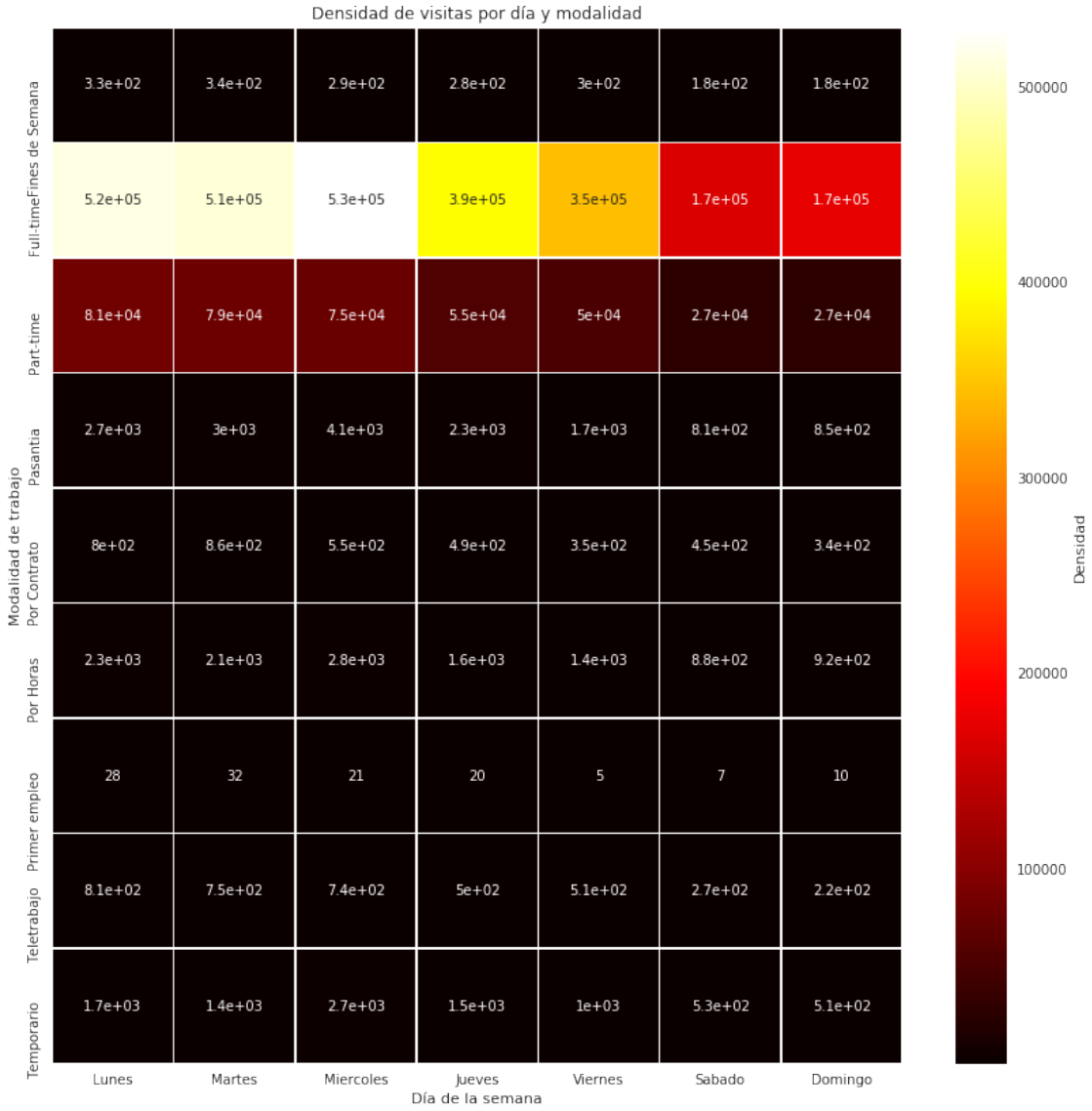



La modalidad con mayor coeficiente de postulantes es la modalidad *"Part-time"*, el segundo puesto pertenece a *"Temporario"* con casi la mitad del anterior. Las modalidades que presentarn un menor coeficiente son *"Teletrabajo"* y *"Primer empleo"*

```
In [39]: df_postulaciones_avisos["diadelasemana"] = df_postulaciones_avisos.fechapostulacion.d
         aux = df_postulaciones_avisos.groupby(["tipo_de_trabajo","diadelasemana"]).size().to_
         aux.columns = ['tipo_de_trabajo', 'dia_de_la_semana', 'cantidad']

In [40]: fig, ax = plt.subplots(figsize=(14,14))

         weekday_map= ["Lunes", "Martes", "Miercoles", "Jueves", "Viernes", "Sabado", "Domingo
         graph = sns.heatmap(aux.pivot_table(index='tipo_de_trabajo',columns='dia_de_la_semana
                    linewidths=.5,cmap="hot", ax=ax, xticklabels=weekday_map, cbar_kws={'label
         ax.set_xlabel('Día de la semana');
         ax.set_ylabel('Modalidad de trabajo');
```

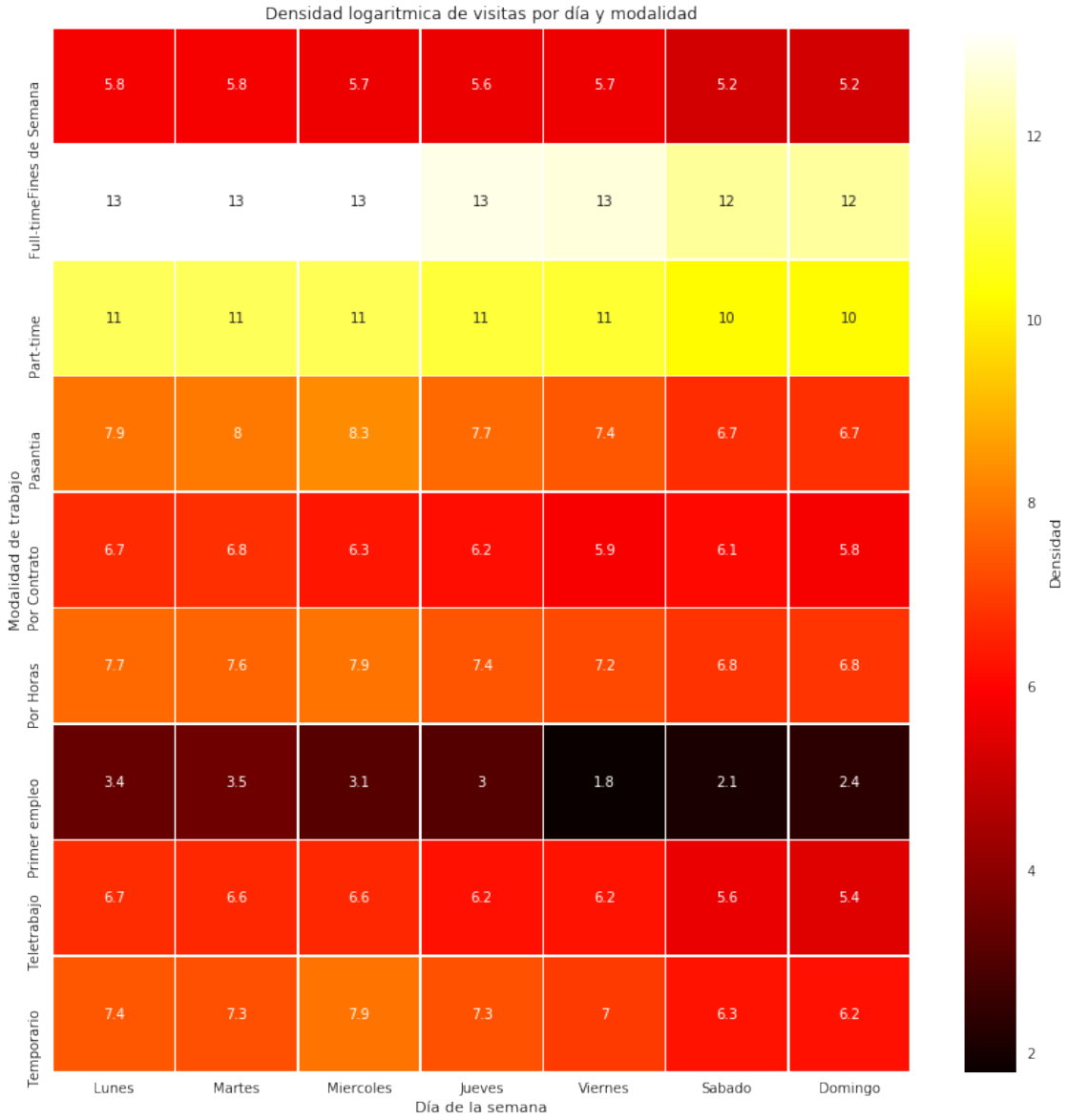
```
ax.set_title("Densidad de visitas por día y modalidad")
graph.plot()
```

Out[40]: []




```
In [41]: aux["log_cantidad"] = np.log(aux.cantidad+1)
         fig, ax = plt.subplots(figsize=(14,14))

         graph = sns.heatmap(aux.pivot_table(index='tipo_de_trabajo',columns='dia_de_la_semana
                     linewidths=.5,cmap="hot", ax=ax, xticklabels=weekday_map, cbar_kws={'label
         ax.set_xlabel('Día de la semana');
         ax.set_ylabel('Modalidad de trabajo');
         ax.set_title("Densidad logaritmica de visitas por día y modalidad")
```

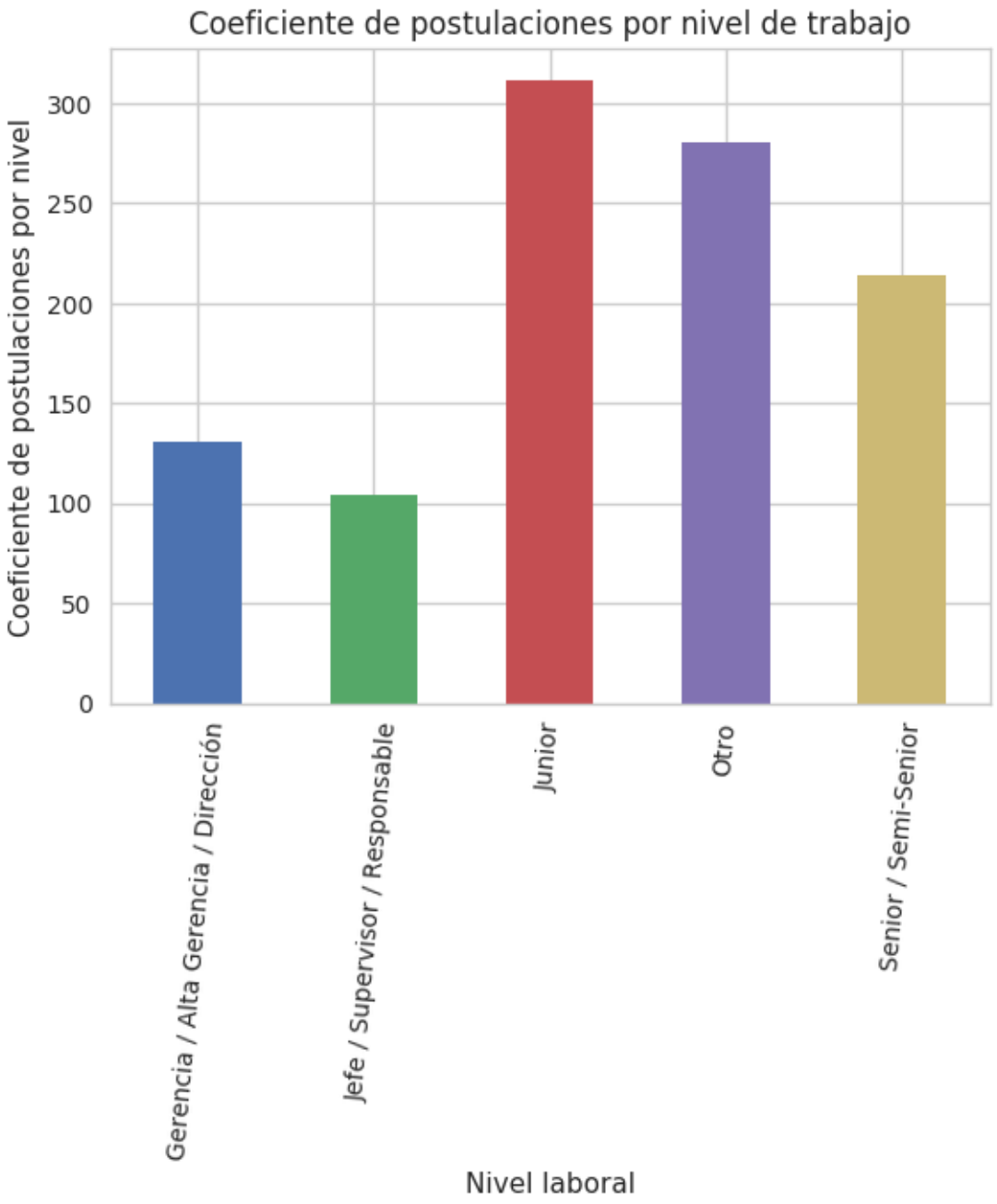
Out[41]: Text(0.5,1,'Densidad logaritmica de visitas por día y modalidad')




### 3.3 Relacion segun nivel laboral

In [42]:
```
postulaciones_promedio_nivel_laboral=df_postulaciones_avisos.groupby(["nivel_laboral"]
```

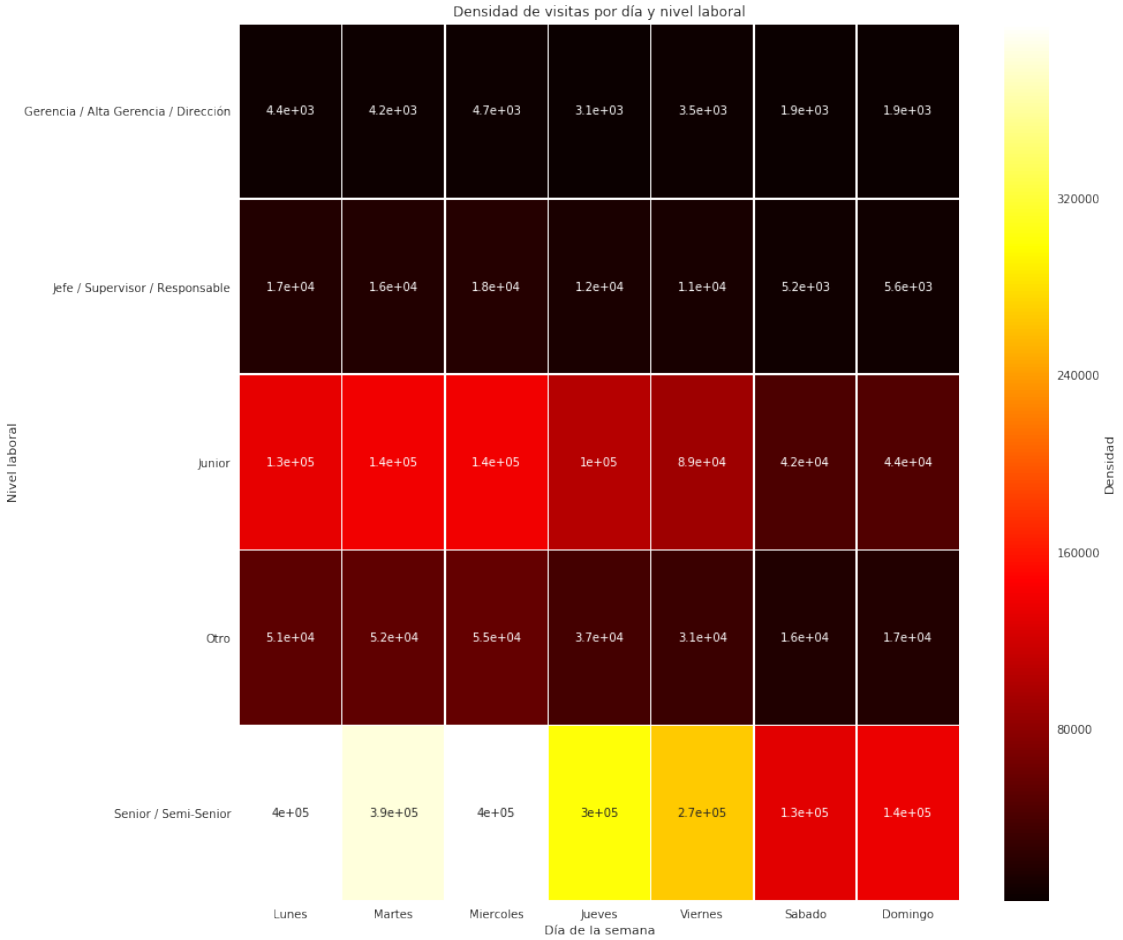
In [70]:
```
title = 'Coeficiente de postulaciones por nivel de trabajo'
postulaciones_promedio_nivel_laboral.plot(kind="bar", rot = 85, title=title)
ax = plt.gca()
ax.set_ylabel('Coeficiente de postulaciones por nivel');
ax.set_xlabel('Nivel laboral');
```




```
In [44]: aux = df_postulaciones_avisos.groupby(["nivel_laboral","diadelasemana"]).size().to_fra
         aux.columns = ['nivel_laboral', 'dia_de_la_semana', 'cantidad']

In [45]: fig, ax = plt.subplots(figsize=(14,14))

         graph = sns.heatmap(aux.pivot_table(index='nivel_laboral',columns='dia_de_la_semana',v
                   linewidths=.5,cmap="hot", ax=ax, xticklabels=weekday_map, cbar_kws={'label
         ax.set_xlabel('Día de la semana');
```

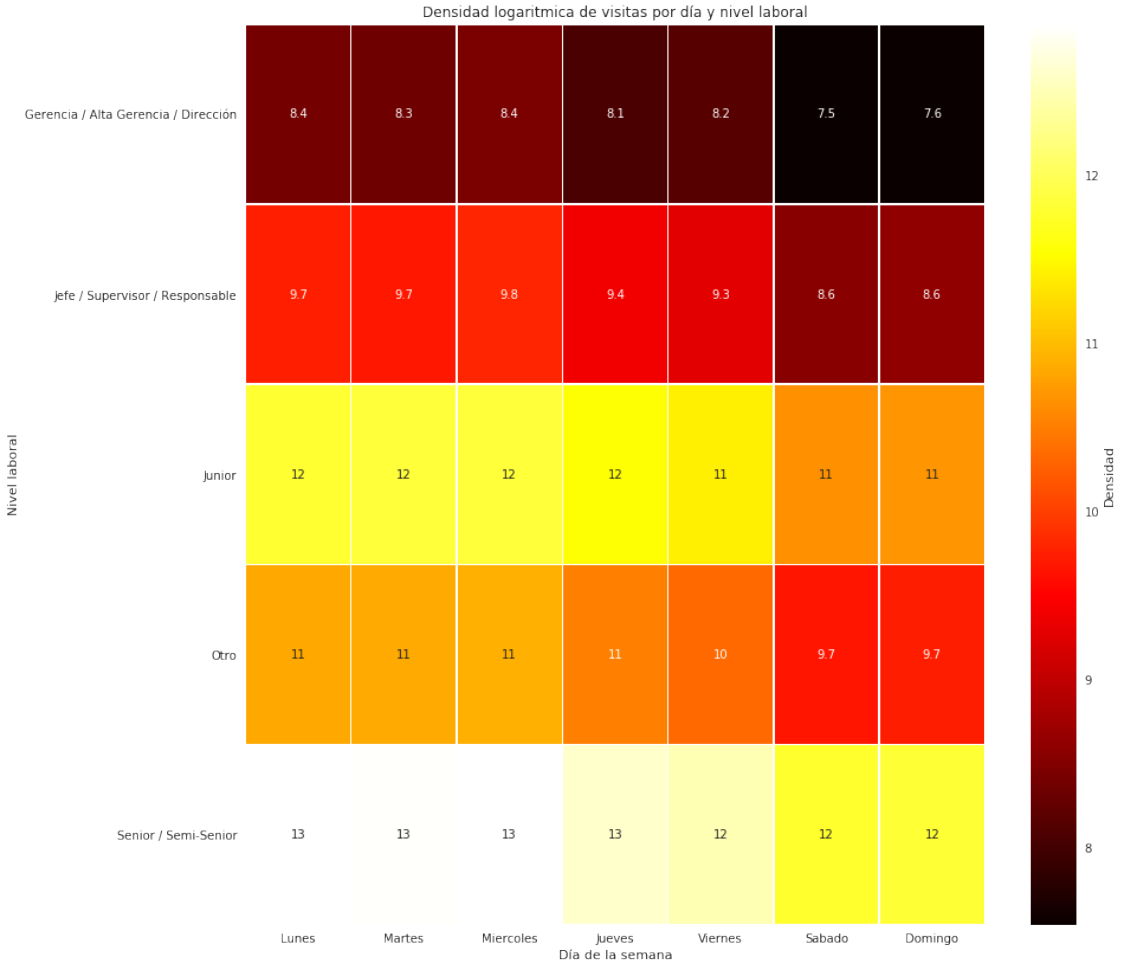
```
        ax.set_ylabel('Nivel laboral');
        ax.set_title("Densidad de visitas por día y nivel laboral")
```

Out[45]: Text(0.5,1,'Densidad de visitas por día y nivel laboral')




```
In [46]: aux["log_cantidad"] = np.log(aux.cantidad+1)
         fig, ax = plt.subplots(figsize=(14,14))
         graph = sns.heatmap(aux.pivot_table(index='nivel_laboral',columns='dia_de_la_semana',v
                    linewidths=.5,cmap="hot", ax=ax, xticklabels=weekday_map, cbar_kws={'label
         ax.set_xlabel('Día de la semana');
         ax.set_ylabel('Nivel laboral');
         ax.set_title("Densidad logaritmica de visitas por día y nivel laboral")
```

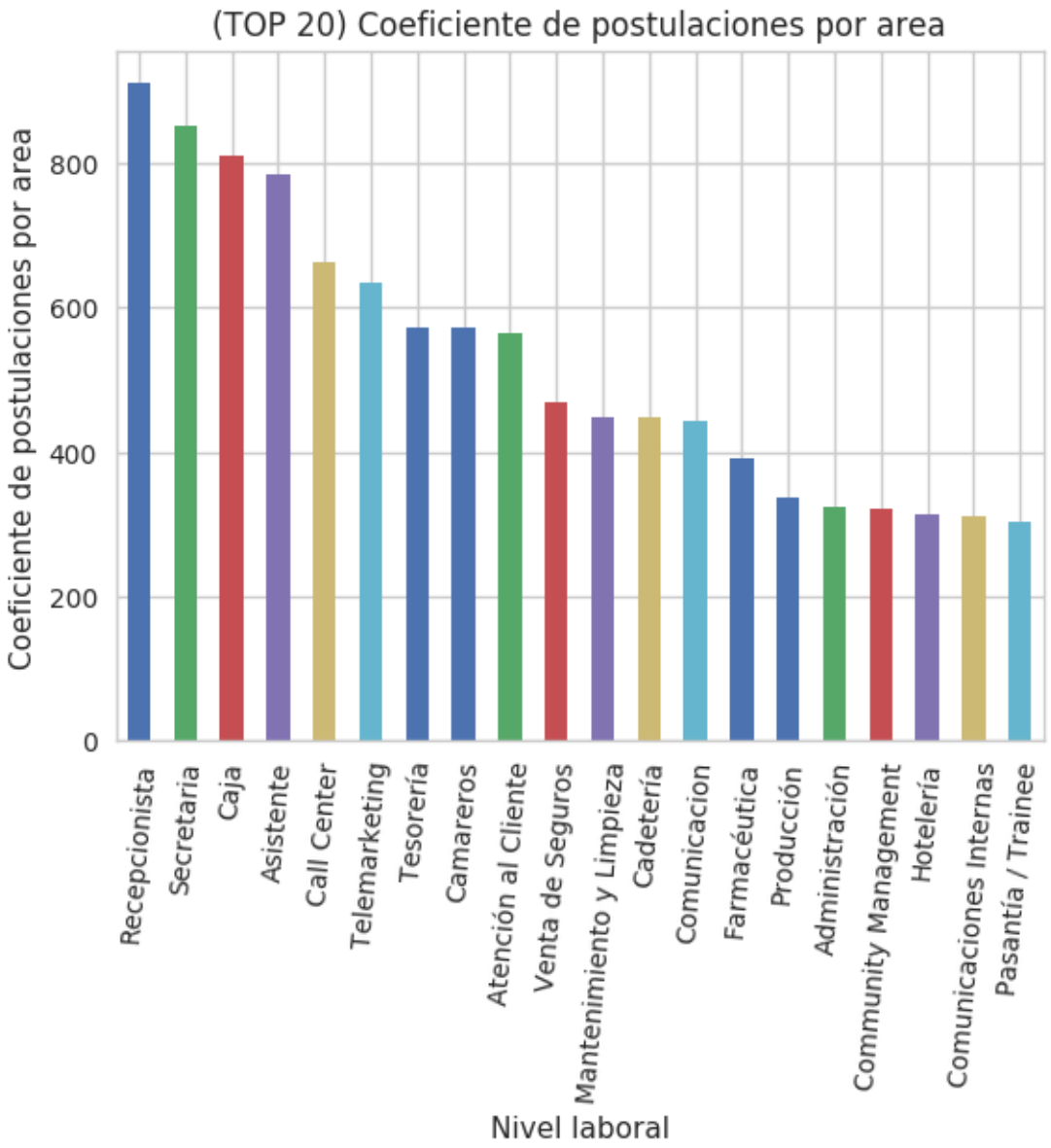
Out[46]: Text(0.5,1,'Densidad logaritmica de visitas por día y nivel laboral')



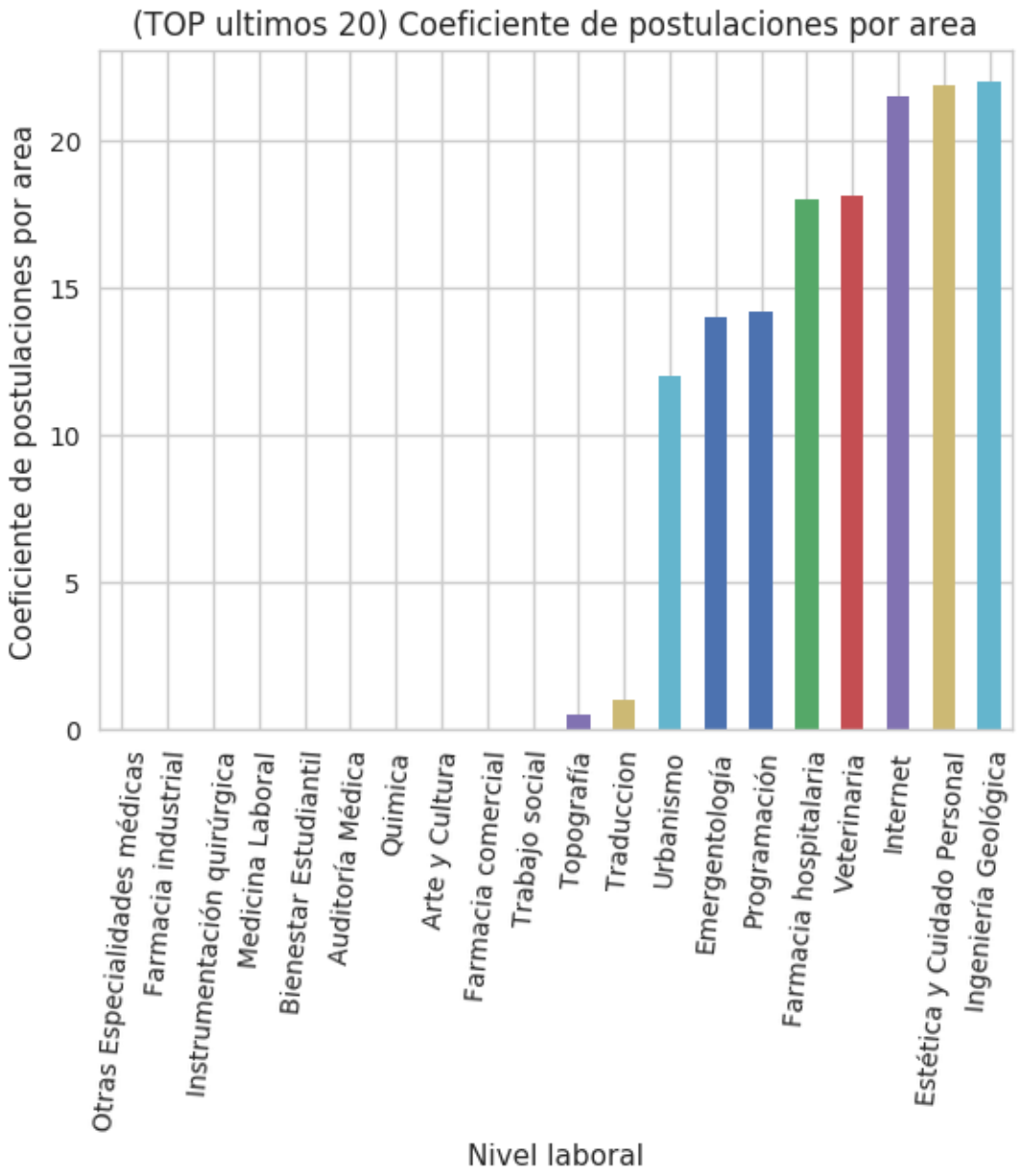

## 3.4 Relacion segun area

```
In [47]: postulaciones_promedio_area=df_postulaciones_avisos.groupby(["nombre_area"]).size()/d
```

```
In [71]: title = '(TOP 20) Coeficiente de postulaciones por area'
         postulaciones_promedio_area.sort_values(ascending=False).head(20)\
             .plot(kind="bar", rot = 85, title=title)
         ax = plt.gca()
         ax.set_ylabel('Coeficiente de postulaciones por area');
         ax.set_xlabel('Nivel laboral');
```



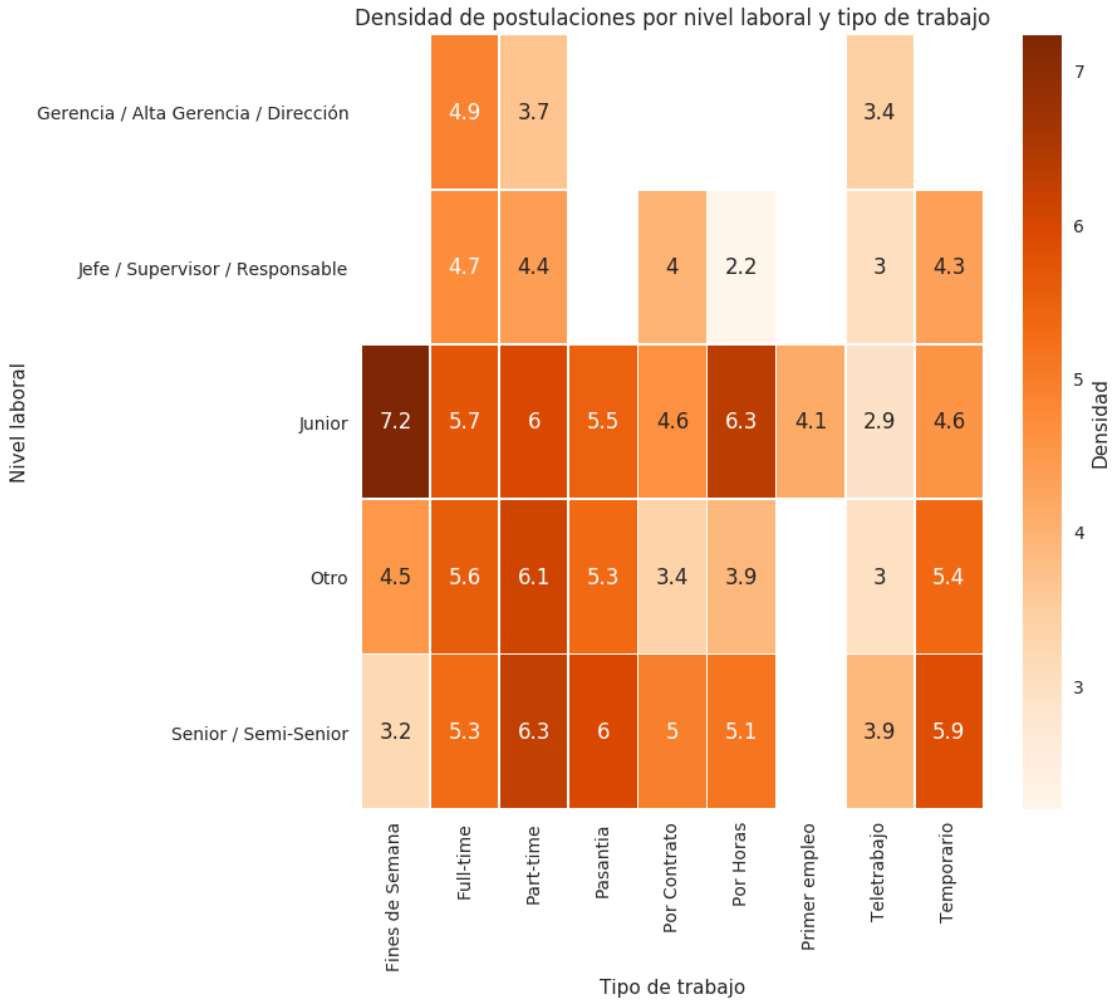

```
In [72]: title = '(TOP ultimos 20) Coeficiente de postulaciones por area'
         postulaciones_promedio_area.sort_values().head(20)\
             .plot(kind="bar", rot = 85, title=title)
         ax = plt.gca()
         ax.set_ylabel('Coeficiente de postulaciones por area');
         ax.set_xlabel('Nivel laboral');
```




```
In [75]: aux = df_postulaciones_avisos.groupby(["nivel_laboral", "tipo_de_trabajo"]).size() \
             .divide(df_avisos_detalle.groupby(["nivel_laboral", "tipo_de_trabajo"]).size())
         aux = aux.to_frame().reset_index(level=[0,1])
         aux.columns = ['nivel_laboral', 'tipo_de_trabajo', 'cantidad']
         aux["log_cantidad"] = np.log(aux.cantidad+1)

In [76]: fig, ax = plt.subplots(figsize=(8,8))

         graph = sns.heatmap(aux.pivot_table(index='nivel_laboral',columns='tipo_de_trabajo',va
                     linewidths=.5,cmap="Oranges", ax=ax, cbar_kws={'label': 'Densidad'}, annot=
```

```
        ax.set_xlabel('Tipo de trabajo');
        ax.set_ylabel('Nivel laboral');
        ax.set_title("Densidad de postulaciones por nivel laboral y tipo de trabajo")
```

```
Out[76]: Text(0.5,1,'Densidad de postulaciones por nivel laboral y tipo de trabajo')
```




## 4 Análisis del tiempo entre la vista y la postulación

A partir del siguiente análisis se propone realizar un análisis del tiempo que transcurre entre que un postulante ve por primera vez un aviso, y el que pasa hasta que se postula al mismo.

```
In [52]: df_postulaciones['fechapostulacion'] = pd.to_datetime(df_postulaciones['fechapostulaci
```

Unifico para que tanto las vistas como las postulaciones esten en el mismo tipo de datos, para poder calcular el delta entre el momento en que se postula, con la primera vez que vio el aviso

```
In [53]: df_vistas_general['timestamp1'] = pd.to_datetime(df_vistas_general['timestamp'], erro
```

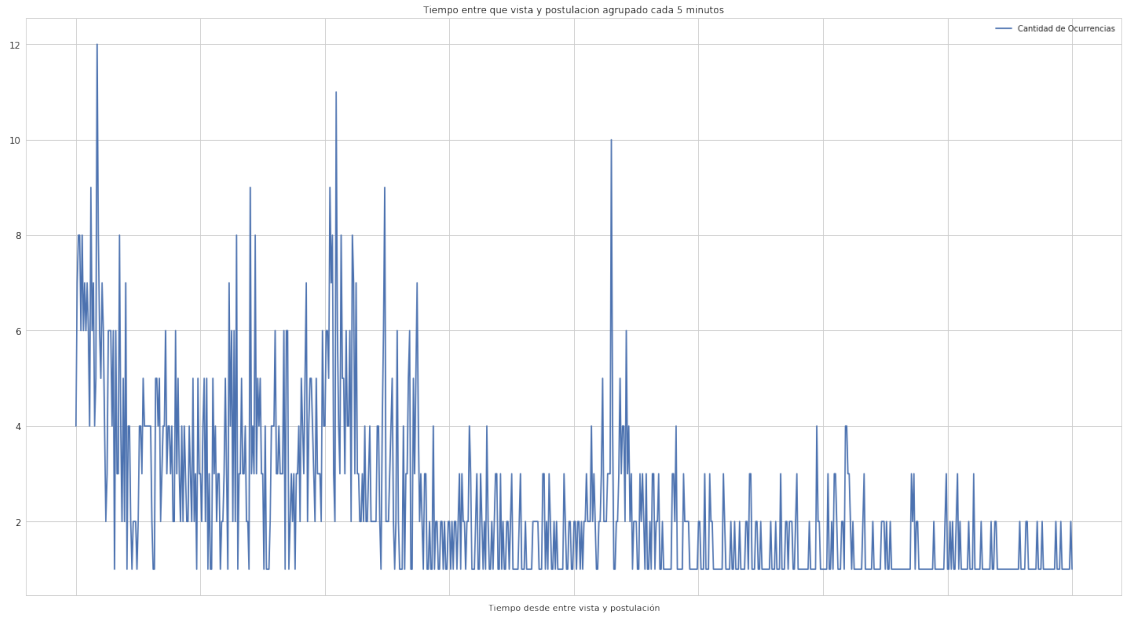
Remuevo aquellos en los que se accede a la vista una vez postulado

```
In [54]: aux = pd.merge(df_vistas_general,df_postulaciones,left_on=['idAviso','idpostulante'],
         temp = aux[aux["timestamp1"]<aux["fechapostulacion"]]
```

```
In [55]: temp["delta"] = (temp['fechapostulacion']-temp['timestamp1']).dt.round('5min')
         tmp = temp.drop_duplicates(['idpostulante','idaviso'],keep='first').groupby('delta').
         tmp = tmp.to_frame()
         tmp.reset_index(inplace=True)
```

```
/home/marcelo/.local/lib/python3.6/site-packages/ipykernel_launcher.py:1: SettingWithCopyWarni
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-docs/stable/indexing.html
  """Entry point for launching an IPython kernel.
```

```
In [67]: tmp.columns = [ 'Tiempo desde entre vista y postulación','Cantidad de Ocurrencias']
         tmp.plot(x="Tiempo desde entre vista y postulación",y="Cantidad de Ocurrencias", title
```




# 5 Conclusión

Se puede observar que inicialmente, el tiempo que se tarda entre el ver el aviso, y postularse es corto. Y conforme pasan los intervalos de tiempo, disminuye la cantidad de personas que esperaron tanto para postularse desde la primera vez que vio el aviso

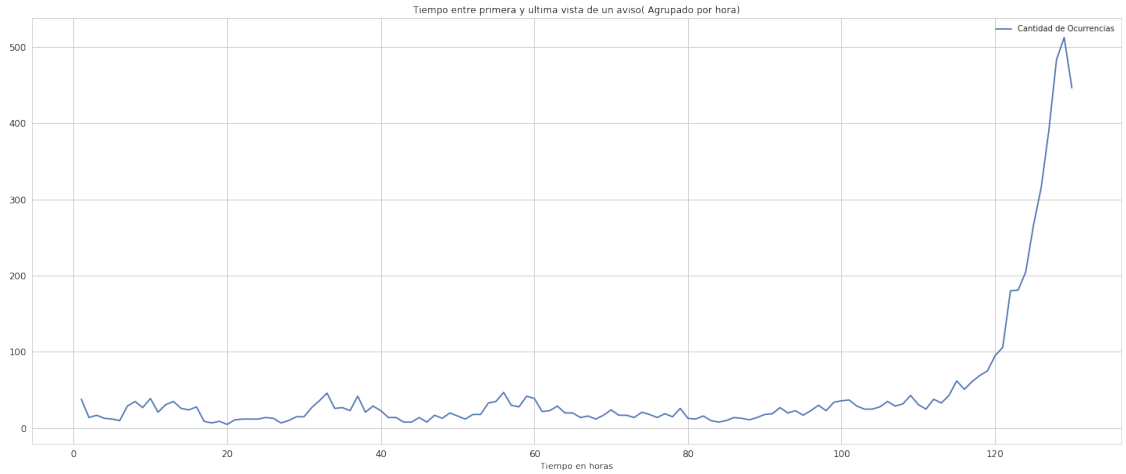
# 6 Tiempo de Vida de un aviso

Para evaluar el tiempo de publicación de un aviso, como no contamos con información sobre la fecha de publlicación y baja de la misma, lo que haremos es comparar la diferencia entre la primera y la ultima vez que se accedio a ver un aviso

```
In [57]: df_agrupar_vistas = df_vistas_general.groupby('idAviso').agg({'timestamp1' : [np.min,
         df_agrupar_vistas["delta"] = (df_agrupar_vistas.timestamp1.amax-df_agrupar_vistas.time
```

Como lo que estoy buscando evaluar es el tiempo entre la primer y ultima visita, retiro todos aquellos valores donde el delta me da 0(cero).

```
In [58]: df_vida_aviso = df_agrupar_vistas[df_agrupar_vistas["delta"].dt.total_seconds()>0].gr
         df_vida_aviso = df_vida_aviso.to_frame()
         df_vida_aviso.reset_index(inplace=True)

In [59]: df_vida_aviso["delta"]=df_vida_aviso["delta"].dt.total_seconds()/3600

In [66]: df_vida_aviso.columns = [ 'Tiempo en horas','Cantidad de Ocurrencias']
         df_vida_aviso.plot(x="Tiempo en horas",y="Cantidad de Ocurrencias", title ="Tiempo en

Out[66]: <matplotlib.axes._subplots.AxesSubplot at 0x7effcca6cef0>
```




## 6.1 Conclusión

A partir del grafico, se puede observar que la mayoria de los avisos presentan entre que accede el primer usuario, y la utima vez que es visto, tiempo mayor