San Francisco Employee Salaries

August 4, 2025

1 San Francisco Employee Salaries Data Analysis

1.1 Project Overview

This project focuses on performing Exploratory Data Analysis (EDA) on a dataset containing more than 300,000 employee compensation records from the city of San Francisco between 2011 and 2018. The data is provided by the Nevada Policy Research Institute to promote transparency and public understanding of government compensation. The primary goal is to use Python for a detailed analysis and visualization of the data to uncover key insights and relationships within employee pay structures.

1.2 Dataset Overview

- **EmployeeName:** Name of the employee.
- **JobTitle:** Title of the job.
- **BasePay:** Base salary pay.
- **OvertimePay:** Pay for overtime work.
- **OtherPay:** Any other types of compensation.
- **Benefits:** Benefits provided to the employee.
- **TotalPay:** The total pay without benefits.
- **TotalPayBenefits:** Total pay with benefits included.
- **Year:** The year of the payroll record.

1.3 Key Analytical Goals

- Analyze the statistical distributions of BasePay, OvertimePay, OtherPay, and Benefits to understand their typical values and ranges.
- Investigate how total compensation and its components have changed over the years from 2011 to 2018. This includes looking at year-over-year changes in average pay.
- Determine the correlations and relationships between different pay components and their collective impact on the final TotalPayBenefits.
- Identify the job titles that receive the highest and lowest total compensation and analyze how different pay components contribute to their overall earnings. This will help uncover disparities across different roles.

- Detect and investigate any unusual patterns or outliers in the compensation data. This could include extremely high or low values in specific pay components that might represent data entry errors or unique circumstances.

1.3.1 Load Required libraries

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

1.3.2 Load Data From CSV File

[3]:
```
# Load csv file
df = pd.read_csv("Salaries.csv",low_memory=False)
df.head(20)
```

[3]:

	EmployeeName	JobTitle	BasePay	OvertimePay	OtherPay	Benefits	TotalPay
0	NATHANIEL FORD	GENERAL MANAGER-METROPOLITAN TRANSIT AUTHORITY	167411.18	0.00	400184.25	Not Provided	567595.43
1	GARY JIMENEZ	CAPTAIN III (POLICE DEPARTMENT)	155966.02	245131.88	137811.38	Not Provided	538909.28
2	ALBERT PARDINI	CAPTAIN III (POLICE DEPARTMENT)	212739.13	106088.18	16452.60	Not Provided	335279.91
3	CHRISTOPHER CHONG	WIRE ROPE CABLE MAINTENANCE MECHANIC	77916.00	56120.71	198306.90	Not Provided	332343.61
4	PATRICK GARDNER	DEPUTY CHIEF OF DEPARTMENT,(FIRE DEPARTMENT)	134401.60	9737.00	182234.59	Not Provided	326373.19
5	DAVID SULLIVAN	ASSISTANT DEPUTY CHIEF II	118602.00	8601.00	189082.74	Not Provided	316285.74
6	ALSON LEE	BATTALION CHIEF, (FIRE DEPARTMENT)	92492.01	89062.90	134426.14	Not Provided	315981.05
7	DAVID KUSHNER	DEPUTY DIRECTOR OF INVESTMENTS					
8	MICHAEL MORRIS	BATTALION CHIEF, (FIRE DEPARTMENT)					
9	JOANNE HAYES-WHITE	CHIEF OF DEPARTMENT, (FIRE DEPARTMENT)					
10	ARTHUR KENNEY	ASSISTANT CHIEF OF DEPARTMENT, (FIRE DEPARTMENT)					
11	PATRICIA JACKSON	CAPTAIN III (POLICE DEPARTMENT)					
12	EDWARD HARRINGTON	EXECUTIVE CONTRACT EMPLOYEE					
13	JOHN MARTIN	DEPARTMENT HEAD V					
14	DAVID FRANKLIN	BATTALION CHIEF, (FIRE DEPARTMENT)					
15	RICHARD CORRIEA	COMMANDER III, (POLICE DEPARTMENT)					
16	AMY HART	DEPARTMENT HEAD V					
17	SEBASTIAN WONG	CAPTAIN, EMERGENCYCY MEDICAL SERVICES					
18	MARTY ROSS	BATTALION CHIEF, (FIRE DEPARTMENT)					
19	ELLEN MOFFATT	ASSISTANT MEDICAL EXAMINER					

```
7   256576.96        0.00   51322.50  Not Provided  307899.46
8   176932.64    86362.68   40132.23  Not Provided  303427.55
9   285262.00        0.00   17115.73  Not Provided  302377.73
10  194999.39    71344.88   33149.90  Not Provided  299494.17
11   99722.00    87082.62  110804.30  Not Provided  297608.92
12  294580.02        0.00       0.00  Not Provided  294580.02
13  271329.03        0.00   21342.59  Not Provided  292671.62
14  174872.64    74050.30   37424.11  Not Provided  286347.05
15  198778.01    73478.20   13957.65  Not Provided  286213.86
16  268604.57        0.00   16115.86  Not Provided  284720.43
17  140546.87   119397.26   18625.08  Not Provided  278569.21
18  168692.63    69626.12   38115.47  Not Provided  276434.22
19  257510.59      880.16   16159.50  Not Provided  274550.25
```

```
    TotalPayBenefits  Year
0          567595.43  2011
1          538909.28  2011
2          335279.91  2011
3          332343.61  2011
4          326373.19  2011
5          316285.74  2011
6          315981.05  2011
7          307899.46  2011
8          303427.55  2011
9          302377.73  2011
10         299494.17  2011
11         297608.92  2011
12         294580.02  2011
13         292671.62  2011
14         286347.05  2011
15         286213.86  2011
16         284720.43  2011
17         278569.21  2011
18         276434.22  2011
19         274550.25  2011
```

1.3.3 Data Inspection

[4]:
```
# Get all columns name
df.columns
```

[4]:
```
Index(['EmployeeName', 'JobTitle', 'BasePay', 'OvertimePay', 'OtherPay',
       'Benefits', 'TotalPay', 'TotalPayBenefits', 'Year'],
      dtype='object')
```

[5]:
```
# get dataframe infomations like data type, counts and non-null
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 312882 entries, 0 to 312881
Data columns (total 9 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   EmployeeName      312882 non-null  object
 1   JobTitle          312882 non-null  object
 2   BasePay           312882 non-null  object
 3   OvertimePay       312882 non-null  object
 4   OtherPay          312882 non-null  object
 5   Benefits          312882 non-null  object
 6   TotalPay          312882 non-null  float64
 7   TotalPayBenefits  312882 non-null  float64
 8   Year              312882 non-null  int64
dtypes: float64(2), int64(1), object(6)
memory usage: 21.5+ MB
```

[6]:
```
# Check null value in dataframe
df.isnull().sum()
```

[6]:
```
EmployeeName        0
JobTitle            0
BasePay             0
OvertimePay         0
OtherPay            0
Benefits            0
TotalPay            0
TotalPayBenefits    0
Year                0
dtype: int64
```

[17]:
```
# Convert compensation columns to a numeric type.
# will convert non-numeric values (like 'Not Provided') to NaN.
for col in ['BasePay', 'OvertimePay', 'OtherPay', 'Benefits', 'TotalPay',␣
 ↪'TotalPayBenefits']:

    # Convert columns to numeric, forcing errors to NaN and then filling with 0
    df[col] = pd.to_numeric(df[col], errors='coerce').fillna(0)

    # convert any negative values to positive
    df[col] = np.abs(df[col])

# Standardize the 'JobTitle' column.
df['JobTitle'] = df['JobTitle'].str.lower().str.strip()

# get dataframe infomations like data type, counts and non-null
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 312882 entries, 0 to 312881
Data columns (total 9 columns):
 #   Column            Non-Null Count   Dtype
---  ------            --------------   -----
 0   EmployeeName      312882 non-null  object
 1   JobTitle          312882 non-null  object
 2   BasePay           312882 non-null  float64
 3   OvertimePay       312882 non-null  float64
 4   OtherPay          312882 non-null  float64
 5   Benefits          312882 non-null  float64
 6   TotalPay          312882 non-null  float64
 7   TotalPayBenefits  312882 non-null  float64
 8   Year              312882 non-null  int64
dtypes: float64(6), int64(1), object(2)
memory usage: 21.5+ MB
```

[18]:
```
# Check duplicate value in dataframe
df.duplicated().sum()
```

[18]:
```
np.int64(0)
```

[19]:
```
# get descriptive statistics of a dataframe like the central tendency,␣
  ↪dispersion, and shape of a distribution
df.describe().T
```

[19]:

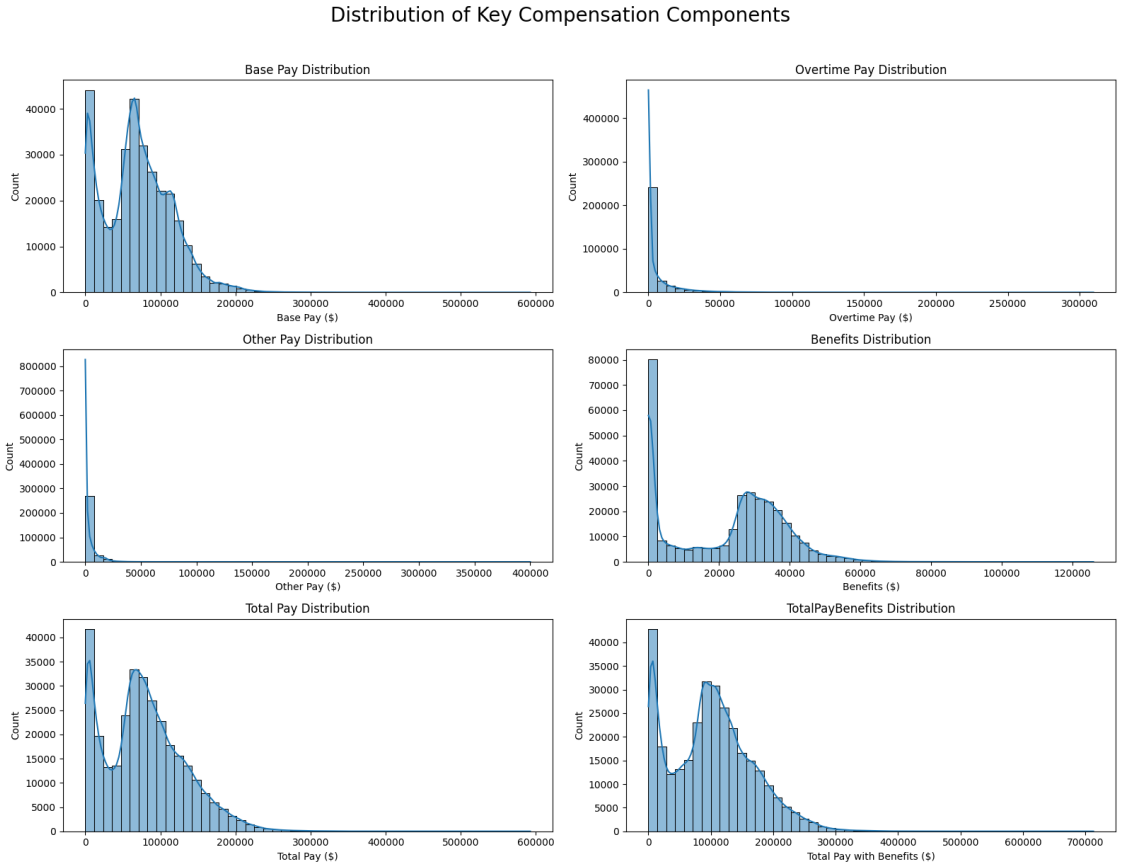
	count	mean	std	min	25%	\
BasePay	312882.0	69673.061646	45436.729791	0.0	35342.1250	
OvertimePay	312882.0	5668.913146	12745.638137	0.0	0.0000	
OtherPay	312882.0	3460.736250	7387.229388	0.0	0.0000	
Benefits	312882.0	22125.945884	16288.841942	0.0	2079.8100	
TotalPay	312882.0	78802.655176	53230.744644	0.0	38803.0000	
TotalPayBenefits	312882.0	100928.373287	66485.135626	0.0	48955.0725	
Year	312882.0	2014.625303	2.290899	2011.0	2013.0000	

	50%	75%	max
BasePay	67645.970	99236.2025	592394.34
OvertimePay	0.000	5223.1150	309481.03
OtherPay	728.000	3958.6800	400184.25
Benefits	26771.565	34288.8475	125891.73
TotalPay	74908.790	111386.8975	592394.34
TotalPayBenefits	100011.290	142376.3000	712802.36
Year	2015.000	2017.0000	2018.00

1.4 Exploratory Analysis

1.4.1 Distribution of Key Compensation Components

[26]:
```
# set chart style properties
fig, axes = plt.subplots(3, 2, figsize=(16, 12))
fig.suptitle('Distribution of Key Compensation Components', fontsize=20, y=1.02)

# Create Histplot For all Key Compensation Components
sns.histplot(df['BasePay'], bins=50, kde=True, ax=axes[0, 0])
axes[0, 0].set_title('Base Pay Distribution')
axes[0, 0].set_xlabel('Base Pay ($)')

sns.histplot(df['OvertimePay'], bins=50, kde=True, ax=axes[0, 1])
axes[0, 1].set_title('Overtime Pay Distribution')
axes[0, 1].set_xlabel('Overtime Pay ($)')

sns.histplot(df['OtherPay'], bins=50, kde=True, ax=axes[1, 0])
axes[1, 0].set_title('Other Pay Distribution')
axes[1, 0].set_xlabel('Other Pay ($)')

sns.histplot(df['Benefits'], bins=50, kde=True, ax=axes[1, 1])
axes[1, 1].set_title('Benefits Distribution')
axes[1, 1].set_xlabel('Benefits ($)')

sns.histplot(df['TotalPay'], bins=50, kde=True, ax=axes[2, 0])
axes[2, 0].set_title('Total Pay Distribution')
axes[2, 0].set_xlabel('Total Pay ($)')

sns.histplot(df['TotalPayBenefits'], bins=50, kde=True, ax=axes[2, 1])
axes[2, 1].set_title('Total Pay Benefits Distribution')
axes[2, 1].set_xlabel('Total Pay with Benefits ($)')

plt.tight_layout()
plt.show()
```



- **BasePay:** The chart shows a right-skewed distribution, meaning the majority of employees have a base pay in the lower to mid-range, and a smaller number of employees have very high base salaries. The most common salary range appears to be between $50,000 and $100,000. There is also a significant number of individuals with very low base pay, likely indicating part-time or hourly workers.
- **OvertimePay:** This distribution is extremely right-skewed and looks heavily concentrated near zero. This indicates that most employees in the dataset receive little to no overtime pay, while a small number of employees earn a significant amount.
- **OtherPay:** Similar to overtime pay, the distribution of OtherPay is also heavily skewed to the right. The vast majority of employees receive a minimal amount of other pay, with a long tail of a few individuals receiving large amounts.
- **Benefits:** The distribution of benefits is right-skewed, but less so than OvertimePay or OtherPay. There is a noticeable peak around the $25,000 to $35,000 range, and another spike at or near zero. This suggests that a portion of employees receive no benefits, while others receive a substantial amount.
- **TotalPay:** The total pay distribution is right-skewed, showing that a large number of employees fall into the lower pay brackets. The shape is similar to BasePay, but shifted slightly to the right due to the inclusion of OvertimePay, OtherPay, and other components.

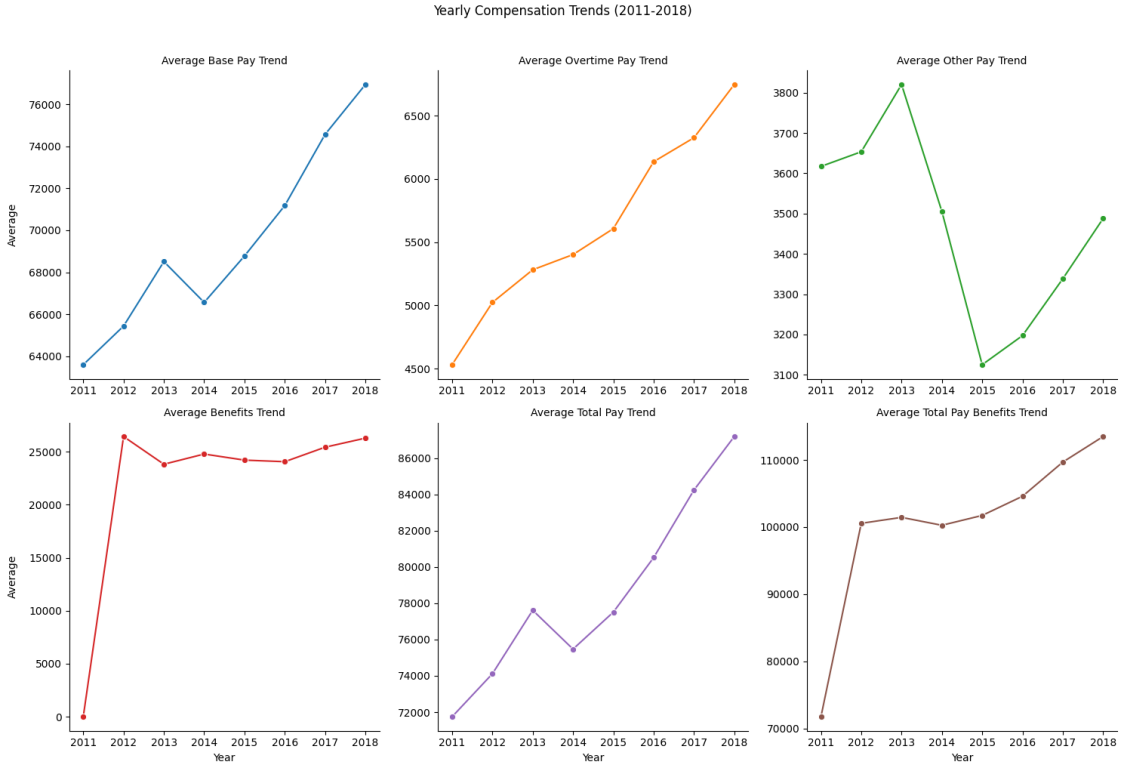
- **TotalPayBenefits:** This distribution, which includes all forms of compensation, mirrors the shape of the TotalPay distribution but is shifted further to the right. This confirms a strong relationship between total pay and total compensation with benefits, and also shows the impact of benefits in increasing the overall compensation for employees.

The histograms show that the compensation of San Francisco employees is not uniformly distributed. All pay components exhibit a right-skewed distribution. This indicates that a significant number of employees are at the lower end of the pay scale, while a smaller number of individuals receive exceptionally high compensation. The strong similarity in the shapes of the TotalPay and TotalPayBenefits distributions suggests that BasePay and Benefits are the primary drivers of total compensation, with OvertimePay and OtherPay having a lesser, though still notable, impact on the overall distribution.

1.4.2 Yearly Compensation Trends (2011-2018)

[49]:
```
# Group by 'Year' and calculate the mean of all compensation columns
yearly_trends = df.groupby('Year').agg(Base_Pay=('BasePay', 'mean'),
                                       Overtime_Pay=('OvertimePay', 'mean'),
                                       Other_Pay=('OtherPay', 'mean'),
                                       Benefits=('Benefits', 'mean'),
                                       Total_Pay=('TotalPay', 'mean'),
                                       Total_Pay_Benefits=('TotalPayBenefits',␣
 ↪'mean')
                                      ).reset_index().round(2)

# Re-arrange dataframe
yearly_trends = yearly_trends.melt(id_vars='Year',
                                   var_name='type',
                                   value_name='value')

# Replace _ with Space
yearly_trends['type'] = yearly_trends['type'].str.replace('_', ' ')

# Create relplot
ax = sns.relplot(data=yearly_trends,
                 x="Year",
                 y="value",
                 hue='type',
                 col='type',
                 col_wrap=3,
                 kind="line",
                 legend=None,
                 marker='o',
                 facet_kws={'sharey': False, 'sharex': False}
)

ax.fig.suptitle("Yearly Compensation Trends (2011-2018)",y=1.02)
ax.set_titles(col_template="Average {col_name} Trend")
```

```
ax.set_ylabels("Average",labelpad=10)

plt.tight_layout()
plt.show()
```



- **BasePay:** The line plot for BasePay shows a consistent and steady increase from 2011 to 2018. This indicates that, on average, the base salaries for employees have been rising each year.
- **OvertimePay:** This line plot shows a more fluctuating trend. OvertimePay saw a significant rise from 2011, peaking around 2014-2015, after which it slightly decreased before stabilizing.
- **OtherPay:** The trend for OtherPay is relatively flat. It was at its highest in 2011 and has remained fairly stable, with only minor fluctuations, throughout the period.
- **Benefits:** The average value of benefits shows a strong and continuous upward trend from 2011 to 2014, followed by a more gradual increase from 2015 to 2018. This suggests that employee benefits have been consistently improving over the years.
- **TotalPayBenefits:** This plot, representing the total compensation including salary and benefits, shows a clear and strong upward trend from 2011 to 2018. This is the most consistent positive growth trend among all the components, which is expected as it is the sum of other components that are also trending upwards.

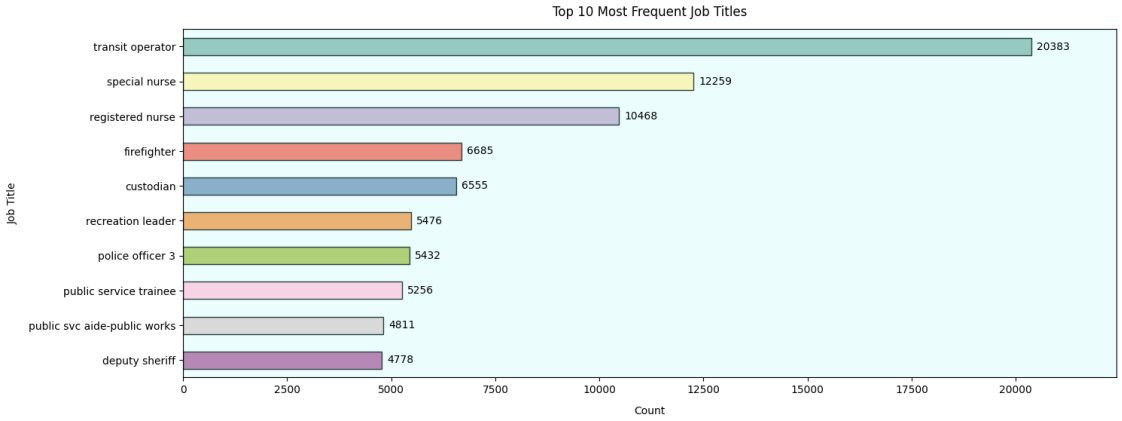
The line plots reveal that the total compensation for San Francisco employees has been on a strong

and steady upward trajectory from 2011 to 2018. This growth is primarily driven by consistent increases in BasePay and Benefits, both of which show clear positive trends over the years. While OvertimePay and OtherPay show more volatility and have not contributed as consistently to the growth in total compensation, their inclusion still adds to the overall increase. The most significant period of growth for benefits occurred in the early years (2011-2014), which contributed significantly to the overall rise in total employee compensation.

[119]:
```
# Count Most Frequent Job Titles
top_jobs = df['JobTitle'].value_counts().reset_index().head(10)
```

1.4.3 Top 10 Most Frequent Job Titles

[164]:
```
# set chart style properties
plt.figure(figsize=(16, 6))

# create barplot
ax = sns.barplot(data=top_jobs,
            y='JobTitle',
            x='count',
            hue='JobTitle',
            legend=False,
            palette='Set3',
            errorbar = None,
            width = 0.5,
            edgecolor="#2b4141",
           )

# set barplot properties
ax.margins(x=0.10)
ax.set_facecolor("#ebfdfd")
ax.set_title('Top 10 Most Frequent Job Titles',y=1.02)
ax.set_xlabel('Count',labelpad=10)
ax.set_ylabel('Job Title',labelpad=10)

# set value on lagend
for container in ax.containers:
    ax.bar_label(container,
                 padding=5)
```



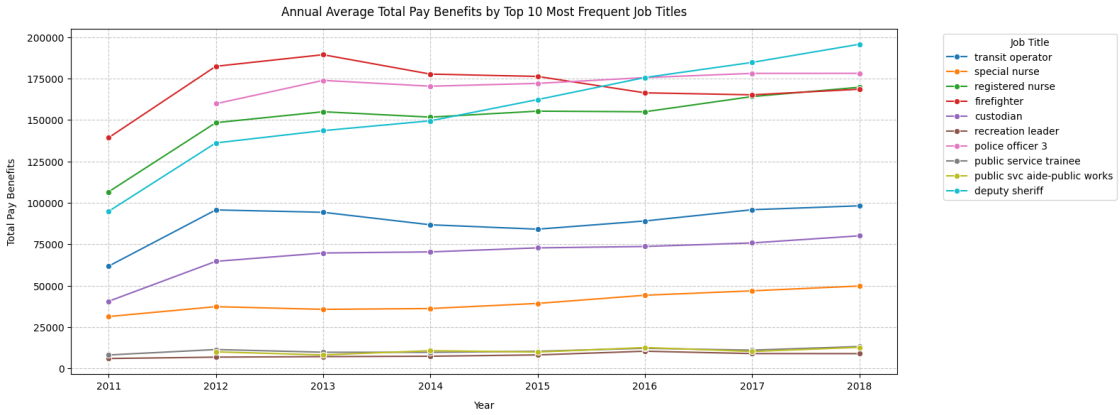
- **Transit Operator** is by far the most frequent job title, with its bar being significantly longer than any other.
- **Special Nurse and Registered Nurse** are also very common, highlighting a strong representation of healthcare professionals.
- Jobs like **Police Officer 3, Police Officer, and Firefighter** are among the top 10, indicating a large number of employees in public safety roles.
- **Public Svc Aide-Public Works and Custodian** also make the list, showing a considerable workforce in public works and maintenance.

The bar chart reveals that the San Francisco workforce is heavily concentrated in essential public services. The dominant position of **Transit Operator** suggests a strong focus on public transportation. The high frequency of jobs in healthcare and public safety further reinforces that the city's employee base is centered on providing critical services to its residents. This chart offers a clear visual breakdown of the most common roles, providing a valuable snapshot of the city's employment priorities.

1.4.4 Annual Average Total Pay Benefits by Top 10 Most Frequent Job Titles

[163]:
```
# calculate yearly total pay benefits by job title
Yearly_paid = df[df['JobTitle'].isin(top_jobs['JobTitle'])].
 ↪groupby(['Year','JobTitle'])['TotalPayBenefits'].mean().reset_index().
 ↪round(2)

# set chart style properties
plt.figure(figsize=(16, 6))

# create lineplot
ax = sns.lineplot(data=Yearly_paid,
             x='Year',
             y='TotalPayBenefits',
             hue_order=top_jobs['JobTitle'],
```

```
            hue='JobTitle',
            marker='o')

# set lineplot property
ax.set_title('Annual Average Total Pay Benefits by Top 10 Most Frequent Job␣
 ↪Titles',y=1.02)
ax.set_xlabel('Year',labelpad=10)
ax.set_ylabel('Total Pay Benefits',labelpad=10)

# Set legend title and position
ax.legend(title='Job Title',
          bbox_to_anchor=(1.05, 1),
          loc='upper left')

ax.grid(True, linestyle='--', alpha=0.7)

plt.tight_layout()
plt.show()
```



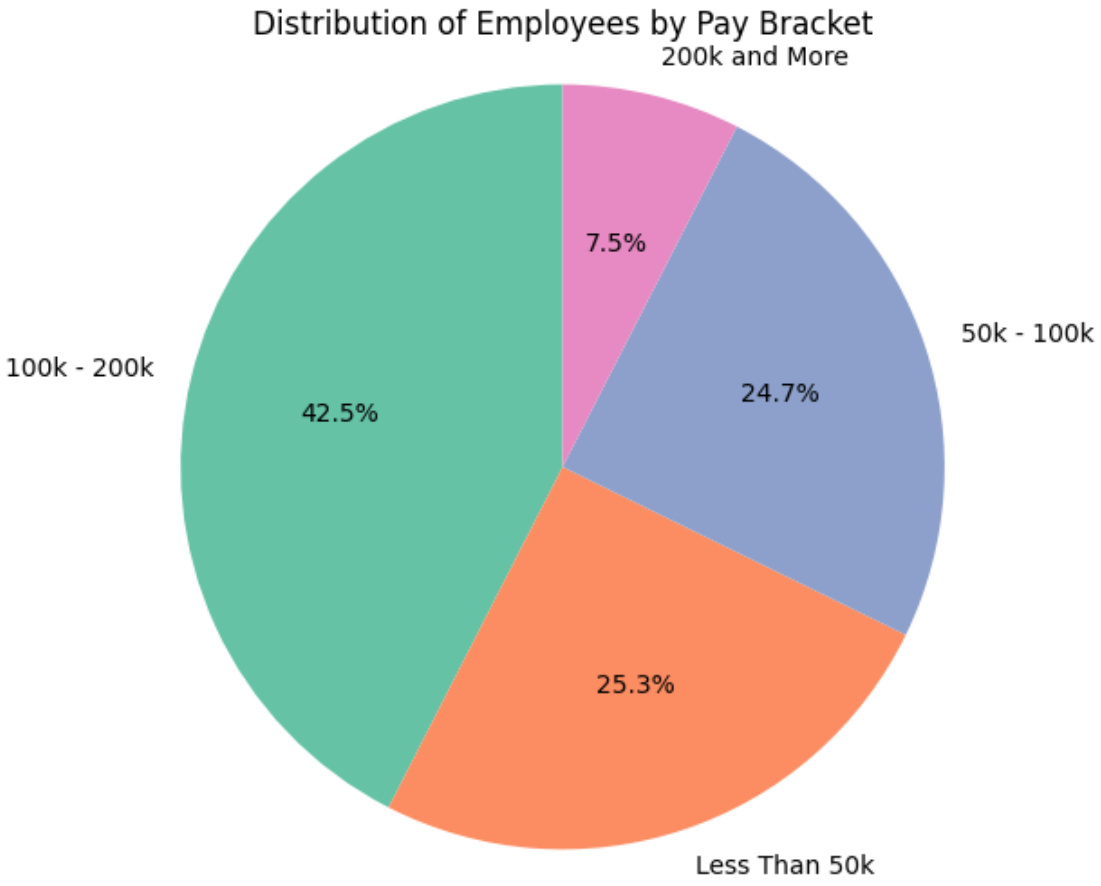
- The **deputy sheriff and firefighter** job titles consistently had the highest average total pay benefits, with the deputy sheriff seeing a particularly sharp increase from 2016 to 2018.
- The **registered nurse and police officer 3** job titles are in a similar range, with benefits fluctuating around the mid-150,000 to 175,000 range.
- Job titles like **custodian, recreation leader, and public service aide trainee** consistently had the lowest average pay benefits, staying below 25,000 throughout the entire period.
- The **deputy sheriff** shows the most significant growth in pay benefits, starting at around 95,000 in 2011 and rising to almost 200,000 by 2018.
- The average pay benefits for **custodian, recreation leader, and public service aide trainee** remained relatively flat from 2011 to 2018, with very little change.

This line chart provides a clear picture of the disparity in average total pay benefits across various

job titles from 2011 to 2018. It highlights that public safety roles like **deputy sheriff and firefighter** command the highest benefits, with the former experiencing the most substantial growth over the period. Conversely, support roles like **custodian and recreation leader** have significantly lower and stagnant benefits. The chart underscores the different compensation trajectories for various professions, with some seeing steady increases while others remain largely unchanged. This data could be valuable for a variety of purposes, including career planning, salary benchmarking, or public policy analysis.

1.4.5 Distribution of Employees by Pay Bracket

[161]:
```
# Define salary bins to categorize TotalPayBenefits into pay brackets
bins = [0, 50000, 100000, 200000, 500000]

# Define labels corresponding to each bin range
labels = ['Less Than 50k', '50k - 100k', '100k - 200k', '200k and More']

# Create a new 'PayBracket' by segmenting 'TotalPayBenefits' using the defined␣
 ↪bins and labels
pay_bracket = pd.cut(df['TotalPayBenefits'], bins=bins, labels=labels).
 ↪value_counts()

# Create Pie Chart
pay_bracket.plot(kind='pie',
                 autopct='%1.1f%%',
                 startangle=90,
                 colors=sns.color_palette("Set2"),
                 ylabel='',
                 figsize=(6, 6))

# set pie chart properties
plt.title('Distribution of Employees by Pay Bracket')
plt.axis('equal')
plt.show()
```



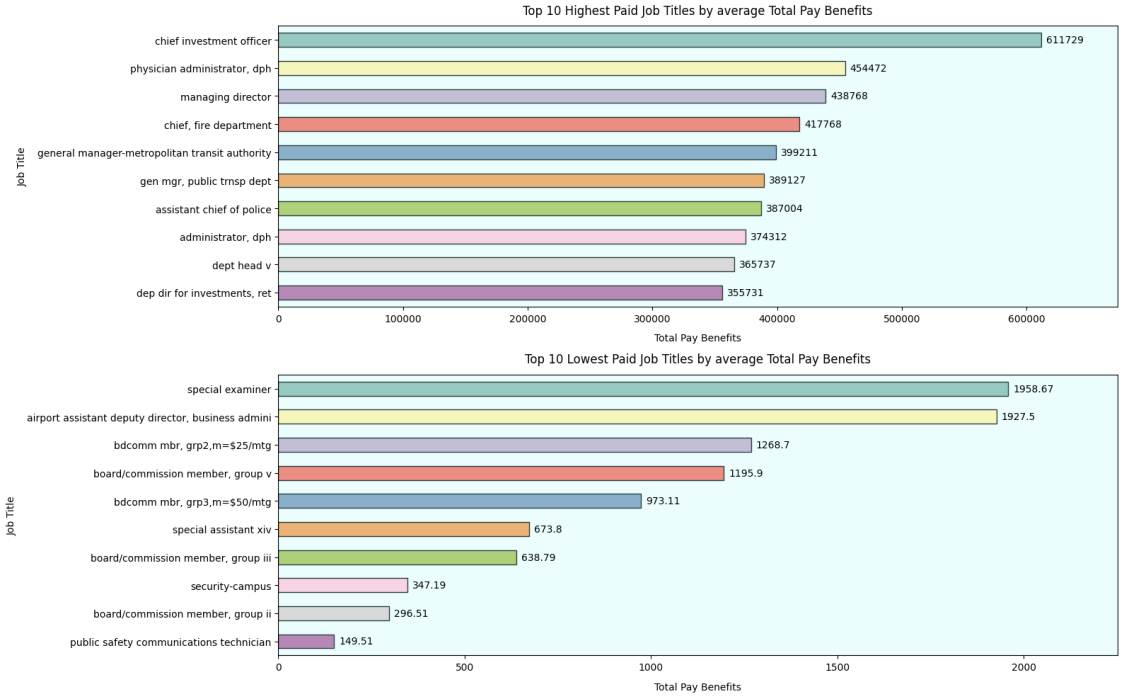
- **100k-200k:** This is the largest segment, representing 42.5% of the total employees. It signifies that nearly half of the employees are in this mid-to-high salary range.
- **Less Than 50k:** This segment is the second largest, accounting for 25.3% of the employees. This indicates a significant portion of the workforce earns less than $50,000 annually.
- **50k-100k:** This segment is very close in size to the lowest-earning group, representing 24.7% of the total employees.
- **200k and More:** This is the smallest segment, with only 7.5% of the employees earning more than $200,000. It represents the highest-paid employees in the dataset.

This pie chart provides a clear visual breakdown of employee distribution across different salary brackets. The data shows that the majority of employees fall into the **100k-200k bracket (42.5%),** indicating a substantial concentration of middle to upper-middle-class income earners. The remaining half of the employees are fairly evenly split between the lower-earning brackets of **Less Than 50k (25.3%) and 50k-100k (24.7%)**. The highest-earning group, those making over **$200,000,** constitutes a small minority at only **7.5%**. Overall, the chart suggests a salary structure where a large portion of the workforce is in the middle-to-high-income range, with a significant segment earning lower salaries and a small, elite group at the very top.

1.4.6 Top 10 Highest/Lowest Paid Job Titles by average Total Pay Benefits

[121]:
```
# Group by JobTitle and calculate the average TotalPayBenefits
jobtitle_totalpaybenefits = df.groupby('JobTitle')['TotalPayBenefits'].mean().
 ↪sort_values(ascending=False).reset_index().round(2)

# set chart style properties
fig, axes = plt.subplots(2, 1, figsize=(16, 10))

# create barplot
sns.barplot(data=jobtitle_totalpaybenefits.head(10),
            y='JobTitle',
            x='TotalPayBenefits',
            hue='JobTitle',
            legend=False,
            palette='Set3',
            errorbar = None,
            width = 0.5,
            edgecolor="#2b4141",
            ax=axes[0]
           )

# set barplot properties
axes[0].margins(x=0.10)
axes[0].set_facecolor("#ebfdfd")
axes[0].set_title('Top 10 Highest Paid Job Titles by average Total Pay␣
 ↪Benefits',y=1.02)
axes[0].set_xlabel('Total Pay Benefits',labelpad=10)
axes[0].set_ylabel('Job Title',labelpad=10)

# set value on lagend
for container in axes[0].containers:
    axes[0].bar_label(container,
                      padding=5)

# create barplot
sns.barplot(data=jobtitle_totalpaybenefits.tail(10),
            y='JobTitle',
            x='TotalPayBenefits',
            hue='JobTitle',
            legend=False,
            palette='Set3',
            errorbar = None,
            width = 0.5,
            edgecolor="#2b4141",
            ax=axes[1]
           )
```

```
# set barplot properties
axes[1].margins(x=0.15)
axes[1].set_facecolor("#ebfdfd")
axes[1].set_title('Top 10 Lowest Paid Job Titles by average Total Pay␣
 ↪Benefits',y=1.02)
axes[1].set_xlabel('Total Pay Benefits',labelpad=10)
axes[1].set_ylabel('Job Title',labelpad=10)

# set value on lagend
for container in axes[1].containers:
    axes[1].bar_label(container,
                      padding=5)

plt.tight_layout()
plt.show()
```



Top 10 Highest Paid Job Titles This chart displays the average total pay and benefits for the 10 most highly compensated job titles.

- The highest-paid position is **Chief of Police,** with an average total compensation exceeding **$400,000**.
- Leadership roles in the Fire Department, such as **Chief of Department and Assistant Chief of Department,** also rank among the top earners.

- This indicates that senior public safety and leadership positions are the most lucrative within the dataset.

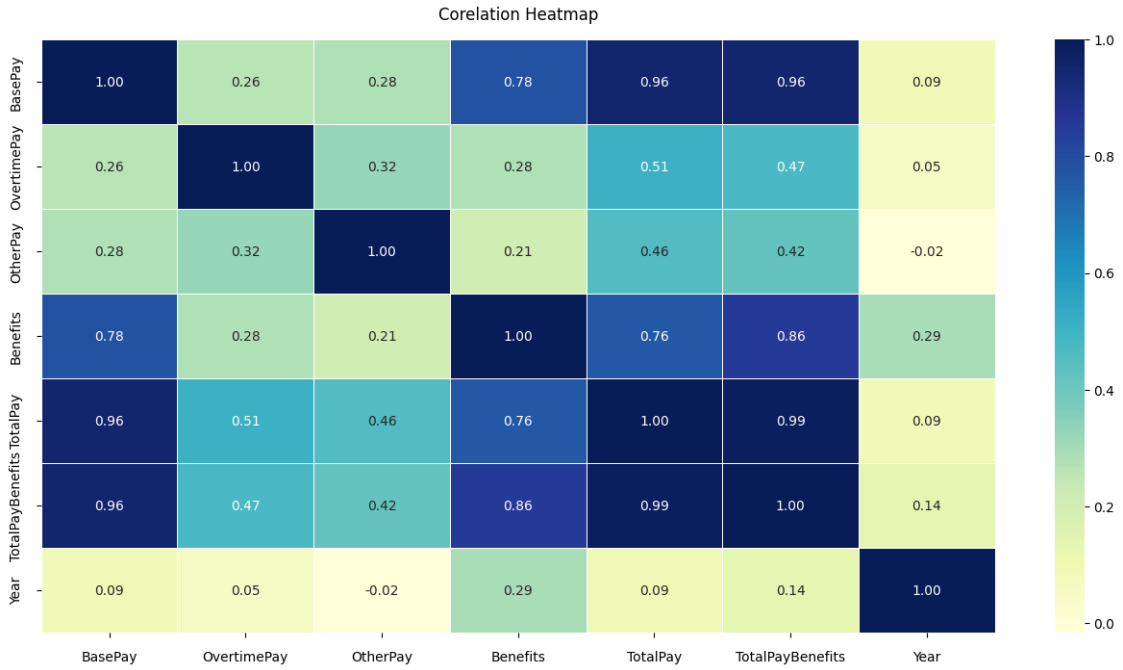
Top 10 Lowest Paid Job Titles This chart shows the average total pay and benefits for the 10 lowest-compensated job titles.

- The average total pay and benefits for these roles are all below **$10,000**, which is significantly lower than the highest-paid jobs.
- The positions on this list often include auxiliary or entry-level roles, such as **Special Nurse, Public Svc Aide-Public Works, and Junior Clerk**.
- This demonstrates a stark contrast in compensation, with these roles receiving minimal average pay and benefits.

These two charts effectively highlight the significant pay disparity within the San Francisco employee dataset. The top-paid positions, which are senior leadership roles in public safety, command average total compensation packages in the hundreds of thousands of dollars. In contrast, the lowest-paid jobs are predominantly entry-level or part-time roles with average total pay and benefits of less than **$10,000**. This stark difference underscores a highly stratified compensation structure where a small number of top-tier positions receive disproportionately high compensation, while many others are at the lower end of the pay scale.

1.4.7 Corelation Heatmap

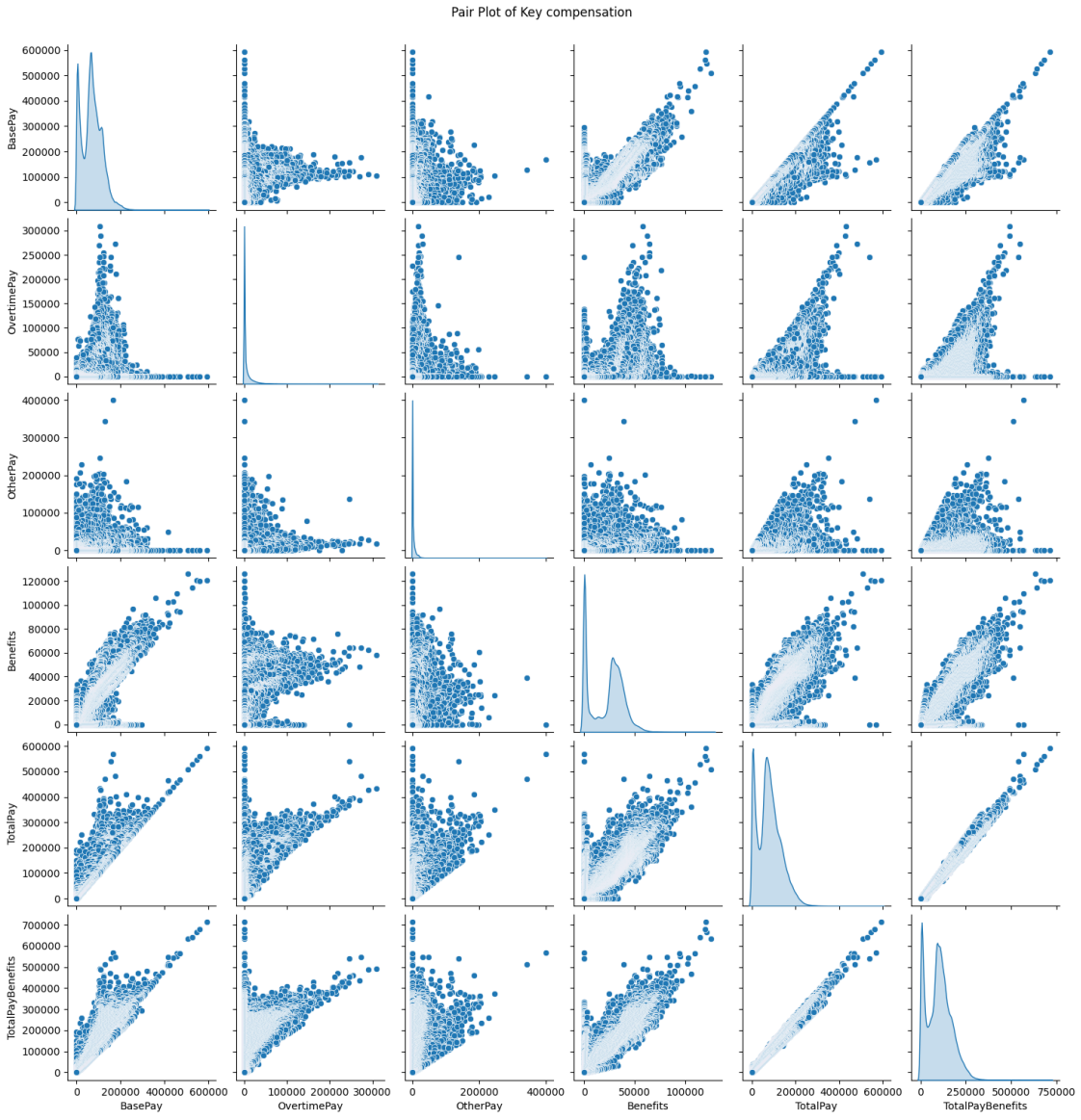
[50]:
```
# set chart style properties
plt.figure(figsize=(16,8))

# Create Heatmap
ax = sns.heatmap(df.corr(numeric_only=True),
            annot=True,
            fmt='.2f',
            linewidths=.5,
            cmap="YlGnBu")

# set Heatmap properties
plt.title('Corelation Heatmap', y=1.02)
ax.tick_params(axis='both',pad=10)
plt.show()
```



- **Strongest Positive Correlation:** The darkest squares on the heatmap show a very strong positive correlation (a value close to 1.0) between TotalPayBenefits and TotalPay. This means as an employee's total pay increases, their total pay with benefits also increases in a very predictable way.
- **Other High Correlations:** BasePay also has a very strong positive correlation with TotalPay and TotalPayBenefits. This indicates that the base salary is a major contributing factor to an employee's overall compensation.
- **Moderate Correlation:** There is a moderate positive correlation between Benefits and both TotalPay and BasePay. This suggests that employees with higher salaries tend to receive more benefits, but the relationship is not as strong as the one between TotalPay and TotalPayBenefits.
- **Weakest Correlation:** OvertimePay and OtherPay have the weakest correlation with the other variables. Their values are not strongly tied to BasePay, Benefits, or TotalPay, suggesting they are more variable and less predictable.

The heatmap provides a clear and concise visual summary of the relationships between the different compensation variables. The most significant finding is the extremely strong positive correlation between TotalPayBenefits, TotalPay, and BasePay, which highlights that an employee's base salary is the primary driver of their overall compensation package. In contrast, OvertimePay and OtherPay are more independent variables, having a weak relationship with the core compensation structure. This indicates that total pay is largely a function of an employee's base salary, with benefits contributing significantly and other pay types being less influential.

[113]:
```
sns.pairplot(data=df[['BasePay', 'OvertimePay', 'OtherPay', 'Benefits',␣
 ↪'TotalPay', 'TotalPayBenefits']],
             diag_kind="kde")
plt.suptitle('Pair Plot of Key compensation', y=1.02)
plt.show()
```



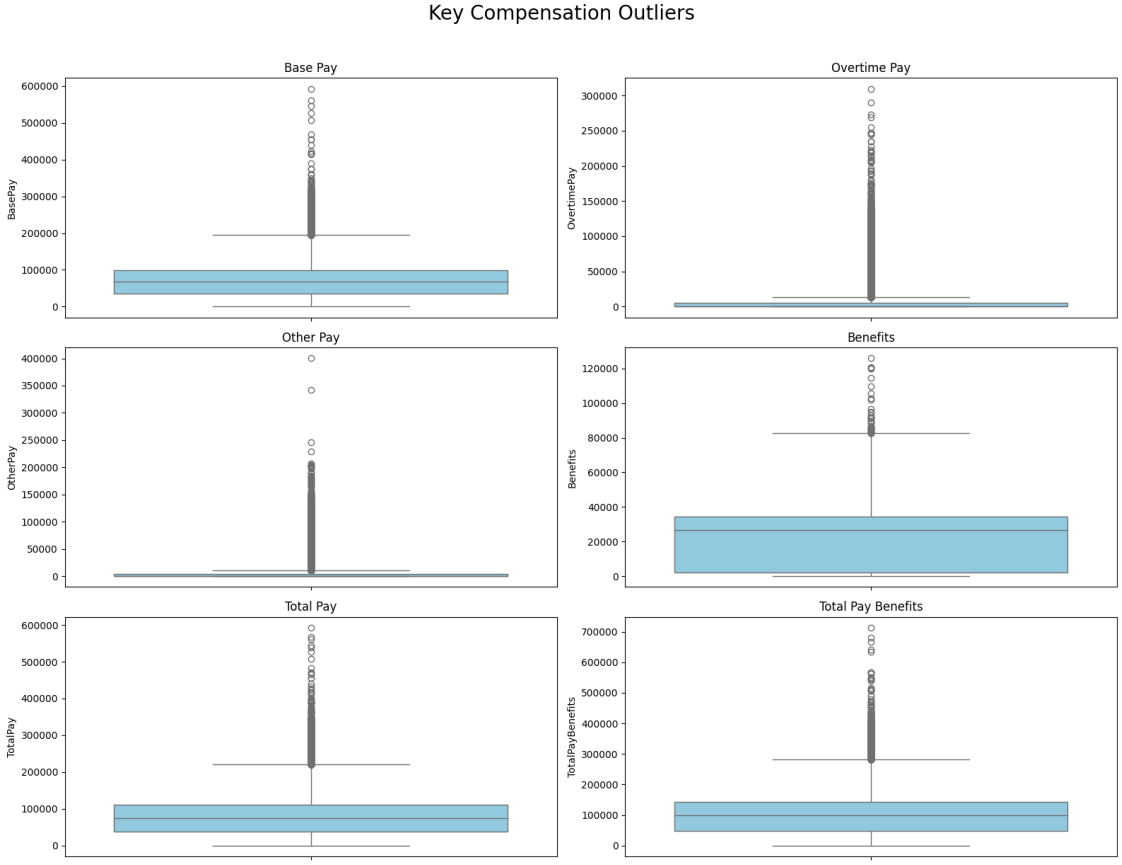
- **BasePay, TotalPay, Benefits, and TotalPayBenefits** all show a right-skewed distribution, meaning the majority of values are concentrated at the lower end, with a long tail of higher values.
- This indicates that most employees have lower compensation, while a smaller number of employees receive significantly higher pay and benefits.

- There is a **very strong positive linear relationship** between **TotalPayBenefits and both TotalPay and BasePay.** The scatter points form a tight, upward-sloping line, indicating that as one variable increases, the other increases proportionally.
- A **strong positive linear relationship** also exists between **TotalPay and BasePay,** suggesting that base salary is the primary component of total pay.
- Benefits show a positive, but **slightly weaker, linear relationship with TotalPayBenefits, TotalPay, and BasePay.** The scatter plots are a bit more spread out, indicating that while higher compensation generally comes with higher benefits, the relationship is not as perfectly linear.

The pair plot provides a comprehensive view of the relationships among the key compensation features. The diagonal histograms confirm that all pay components have a right-skewed distribution, with a few high-earners and many low-to-mid earners. The scatter plots reveal very strong positive linear correlations between TotalPayBenefits, TotalPay, and BasePay, demonstrating that these variables are highly interdependent. The analysis suggests that a higher base salary is the main driver of an employee's total compensation and total benefits package.

1.4.8 Identify Outliers

[116]:
```
# set chart style properties
fig, axes = plt.subplots(3, 2, figsize=(16, 12))
fig.suptitle('Key Compensation Outliers', fontsize=20, y=1.02)

# Create Boxplot For Key Compensation
sns.boxplot(y=df['BasePay'], data=df, ax=axes[0, 0], color='skyblue')
axes[0, 0].set_title('Base Pay')

sns.boxplot(y=df['OvertimePay'], data=df, ax=axes[0, 1], color='skyblue')
axes[0, 1].set_title('Overtime Pay')

sns.boxplot(y=df['OtherPay'], data=df, ax=axes[1, 0], color='skyblue')
axes[1, 0].set_title('Other Pay')

sns.boxplot(y=df['Benefits'], data=df, ax=axes[1, 1], color='skyblue')
axes[1, 1].set_title('Benefits')

sns.boxplot(y=df['TotalPay'], data=df, ax=axes[2, 0], color='skyblue')
axes[2, 0].set_title('Total Pay')

sns.boxplot(y=df['TotalPayBenefits'], data=df, ax=axes[2, 1], color='skyblue')
axes[2, 1].set_title('Total Pay Benefits')

plt.tight_layout()
plt.show()
```



- **BasePay:** The box is relatively wide, showing a significant spread in base salaries. The median is positioned towards the lower end of the box, indicating that more employees have lower base salaries. There is a large number of dots above the top whisker, indicating a substantial number of high-end outliers in base pay.
- **OvertimePay:** The box is very narrow and close to zero, suggesting that the vast majority of employees receive little to no overtime pay. The median is at zero. The plot shows a massive number of outliers extending far above the main distribution. This indicates that a small number of employees receive exceptionally high amounts of overtime pay.
- **OtherPay:** Similar to OvertimePay, the box is very narrow and centered near zero, which means most employees receive a small amount of "other" pay. The median is also at or near zero. Like the other plots, it features numerous outliers that extend to very high values, suggesting that a select group of employees receives significant additional compensation.
- **Benefits:** The box is wider than the pay-related boxes, showing a more significant spread in benefits for the middle 50% of employees. The median is positioned within the box, showing a more balanced distribution. There is a large collection of outliers above the top whisker, indicating that a minority of employees receive a much higher level of benefits.
- **TotalPay:** The box is relatively wide, showing a significant spread in total pay. The median is positioned towards the lower end of the box, indicating that more employees have lower

total pay. Similar to the BasePay plot, there are many dots above the top whisker, indicating a substantial number of high-end outliers in total pay.

- **TotalPayBenefits** The box for this plot is the widest of all, showing the largest spread in compensation when benefits are included. The median is positioned slightly higher than in the TotalPay plot, reflecting the addition of benefits. Like the others, this plot also has a large number of outliers, but they extend to the highest values, confirming that a small group of employees receives exceptionally high total compensation when all factors are included.

1.5 Recommendations

- Review salary distribution and address significant income gaps between senior leadership and lower-tier staff. Focus on reducing budget strain caused by excessively high compensation packages in roles like Chief of Police and Fire Department leadership.
- Prioritize financial strategies around BasePay and Benefits, which are the strongest contributors to total compensation. Adjustments here will have the most meaningful impact on overall salary expenditures.
- Investigate and control outliers in OvertimePay and OtherPay to prevent potential misuse and ensure fair distribution of additional compensation resources.
- Incorporate year-over-year growth patterns of BasePay and Benefits into long-term financial planning to better anticipate future payroll and benefits obligations.
- Reinforce compensation and retention initiatives for high-volume, public-serving job titles like Transit Operators, Nurses, Police Officers, and Firefighters, which form the operational backbone of the city.
- Implement stricter payroll data validation rules to eliminate entries like “Not Provided” and unify job title formats—improving data integrity and enabling more reliable financial analysis.

[ ]: