Assignment 03

Julio Vargas

September 21, 2025

from pyspark.sql import SparkSession  
from pyspark.sql import functions as F  
from pyspark.sql.functions import col, split, explode, regexp\_replace, transform, when  
from pyspark.sql.functions import col, monotonically\_increasing\_id  
from pyspark.sql.types import StructType # to/from JSON  
  
import json  
import re  
import numpy as np  
import pandas as pd  
  
import plotly.express as px  
import plotly.io as pio  
import plotly.graph\_objects as go  
  
  
np.random.seed(30) # set a fixed seed for reproducibility  
pio.renderers.default = "vscode+notebook" #  
# Initialize Spark Session  
spark = SparkSession.builder.appName("JobPostingsAnalysis").getOrCreate()  
# Load schema from JSON file  
with open("data/schema\_lightcast.json") as f:  
 schema = StructType.fromJson(json.load(f))  
  
# Load Data  
df = (spark.read  
 .option("header", "true")  
 .option("inferSchema", "false")  
 .schema(schema) # saved schema  
 .option("multiLine", "true")  
 .option("escape", "\"")  
 .csv("data/lightcast\_job\_postings.csv")  
 .limit(5000))  
# Show Schema and Sample Data  
#df.printSchema()   
#df.show(5)

# Histogram of SALARY distribution (cast + filter)  
salary\_df = (  
 df.select(col("SALARY").cast("float"))  
 .filter(col("SALARY").isNotNull() & (col("SALARY") > 0))  
)  
  
fig = px.histogram(  
 salary\_df.toPandas(),  
 x="SALARY",  
 nbins=50,  
 title="Salary Distribution"  
)  
fig.update\_layout(bargap=0.1)  
fig

Unable to display output for mime type(s): text/html

Unable to display output for mime type(s): application/vnd.plotly.v1+json, text/html

# 1. Data Preparation

# Step 1: Casting salary and experience columns  
from pyspark.sql.functions import col  
  
df = df.withColumn("SALARY", col("SALARY").cast("float")) \  
 .withColumn("SALARY\_FROM", col("SALARY\_FROM").cast("float")) \  
 .withColumn("SALARY\_TO", col("SALARY\_TO").cast("float")) \  
 .withColumn("MIN\_YEARS\_EXPERIENCE", col("MIN\_YEARS\_EXPERIENCE").cast("float"))\  
 .withColumn("MAX\_YEARS\_EXPERIENCE", col("MAX\_YEARS\_EXPERIENCE").cast("float"))  
  
# Step 2: Computing medians for salary columns  
def compute\_median(sdf, col\_name):  
 q = sdf.approxQuantile(col\_name, [0.5], 0.01)  
 return q[0] if q else None  
  
median\_from = compute\_median(df, "SALARY\_FROM")  
median\_to = compute\_median(df, "SALARY\_TO")  
median\_salary = compute\_median(df, "SALARY")  
  
print("Medians:", median\_from, median\_to)  
  
# Step 3: Imputing missing salaries, but not experience  
df = df.fillna({  
 "SALARY\_FROM": median\_from,  
 "SALARY\_TO": median\_to,  
 "SALARY": median\_salary  
})  
  
# Step 5: Computing average salary  
df = df.withColumn("Average\_Salary",  
 (col("SALARY\_FROM") + col("SALARY\_TO")) / 2)  
  
# Step 6: Selecting required columns  
export\_cols = [  
 "EDUCATION\_LEVELS\_NAME",  
 "REMOTE\_TYPE\_NAME",  
 "MAX\_YEARS\_EXPERIENCE",  
 "Average\_Salary",  
 "SALARY",  
 "LOT\_V6\_SPECIALIZED\_OCCUPATION\_NAME"  
]  
df\_selected = df.select(\*export\_cols)  
  
# Step 7: Saving to CSV  
pdf = df\_selected.toPandas()  
pdf.to\_csv("data/lightcast\_cleaned.csv", index=False)  
  
print("Data cleaning complete. Rows retained:", len(pdf))

Medians: 89565.0 131400.0  
Data cleaning complete. Rows retained: 5000

## 1.1 Salary Distribution by Industry and Employment Type

* Compare salary variations across industries.

**Filter the dataset** - Remove records where **salary is missing or zero**.

**Aggregate Data** - Group by **NAICS industry codes** (e.g., NAICS2\_NAME). - Group by **employment type** (EMPLOYMENT\_TYPE\_NAME) and compute salary distribution. - Calculate **salary percentiles** (25th, 50th, 75th) for each group.

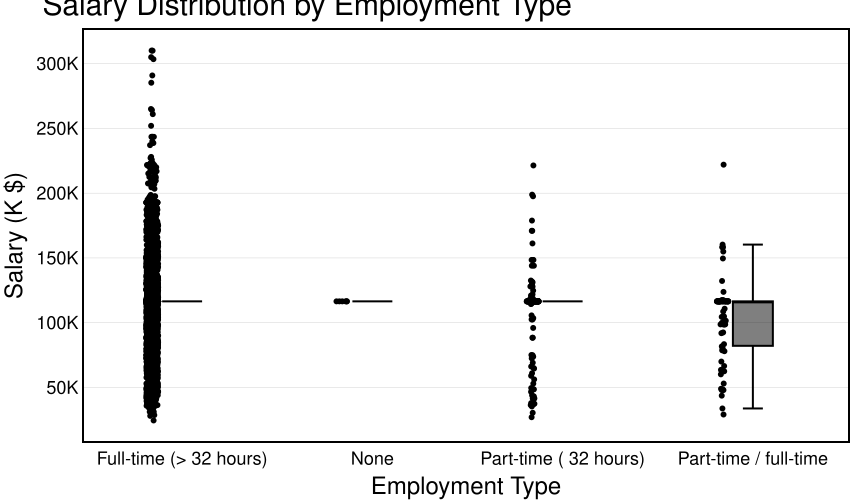
**Visualize results** - Create a **box plot** where: - **X-axis** = NAICS2\_NAME - **Y-axis** = SALARY\_FROM, or SALARY\_TO, or SALARY - **Group/color** = EMPLOYMENT\_TYPE\_NAME - Customize colors, fonts, and styles.

**Explanation:** Write two sentences about what the graph reveals (e.g., median differences across industries and dispersion by employment type).

#your code for first query  
import pandas as pd  
import polars as pl  
from IPython.display import display, HTML  
  
# Filter out missing or zero salary values  
pdf = df.filter(df["SALARY"] > 0).select("EMPLOYMENT\_TYPE\_NAME", "SALARY").toPandas()  
  
  
# Clean employment type names for better readability  
pdf["EMPLOYMENT\_TYPE\_NAME"] = (  
 pdf["EMPLOYMENT\_TYPE\_NAME"]  
 .astype(str)  
 .str.replace(r"[^\x00-\x7F]+", "", regex=True)  
)  
  
#display(HTML(f"<div style='height:300px; overflow:auto'>{pdf.iloc[:10].to\_html(index=False)}</div>"))  
  
# Compute median salary for sorting  
median\_salaries = pdf.groupby("EMPLOYMENT\_TYPE\_NAME")["SALARY"].median()  
display(median\_salaries.to\_frame().head())  
  
  
# Sort employment types based on median salary in descending order  
sorted\_employment\_types = median\_salaries.sort\_values(ascending=False).index  
  
# Apply sorted categories  
pdf["EMPLOYMENT\_TYPE\_NAME"] = pd.Categorical(  
 pdf["EMPLOYMENT\_TYPE\_NAME"],  
 categories=sorted\_employment\_types,  
 ordered=True  
)  
  
# Create box plot with horizontal grid lines  
fig = px.box(  
 pdf,  
 x="EMPLOYMENT\_TYPE\_NAME",  
 y="SALARY",  
 title="Salary Distribution by Employment Type",  
 color\_discrete\_sequence=["#CC0000"], # Single neutral color  
 boxmode="group",  
 points="all" # Show all outliers  
)  
fig  
  
  
  
# Improve layout, font styles, and axis labels  
fig.update\_layout(  
 title=dict(  
 text="Salary Distribution by Employment Type",  
 font=dict(size=16, family="Helvetica", color="black") # Bigger & Bold Title  
 ),  
 xaxis=dict(  
 title=dict(text="Employment Type", font=dict(size=14, family="Helvetica", color="black",weight="bold")), # Bigger X-label  
 tickangle=0, # Rotate X-axis labels for readability  
 tickfont=dict(size=12, family="Helvetica", color="black"), # Bigger & Bold X-ticks  
 showline=True, # Show axis lines  
 linewidth=2, # Thicker axis lines  
 linecolor="black",  
 mirror=True,  
 showgrid=False, # Remove vertical grid lines  
 categoryorder="array",  
 categoryarray=sorted\_employment\_types.tolist()  
 ),  
 yaxis=dict(  
 title=dict(text="Salary (in $1000)", font=dict(size=14, family="Helvetica", color="black",weight="bold")), # Bigger Y-label  
 tickvals=[0, 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450000, 500000],  
 ticktext=["0", "50", "100", "150", "200", "250", "300", "350", "400", "450", "500"],  
 tickfont=dict(size=12, family="Helvetica", color="black"), # Bigger & Bold Y-ticks  
 showline=True,  
 linewidth=2,  
 linecolor="black",  
 mirror=True,  
 showgrid=True, # Enable light horizontal grid lines  
 gridcolor="lightgray", # Light shade for the horizontal grid  
 gridwidth=0.5 # Thin grid lines  
 ),  
 font=dict(family="Helvetica", size=12, color="black"),  
 boxgap=0.7,  
 plot\_bgcolor="white",  
 paper\_bgcolor="white",  
 showlegend=False,  
 height=500,  
 width=850  
)  
  
# Show & export  
fig.show()  
fig.write\_html("output/Q1.html")  
fig.write\_image("output/Q1.svg", width=850, height=500, scale=1)

|  | SALARY |
| --- | --- |
| EMPLOYMENT\_TYPE\_NAME |  |
| Full-time (> 32 hours) | 116490.0 |
| None | 116490.0 |
| Part-time ( 32 hours) | 116490.0 |
| Part-time / full-time | 116490.0 |

Unable to display output for mime type(s): application/vnd.plotly.v1+json, text/html



# 2. Salary Distribution by Industry and Employment Type

* Compare salary variations across industries.
* **Filter the dataset**
  + Remove records where **Salary is missing or zero**.
* **Aggregate Data**
  + Group by **NAICS industry codes**.
  + Group by **employment type** and compute salary-distribution.
* **Visualize results**
  + Create a **box plot** where:
    - **X-axis** = NAICS2\_NAME.
    - **Y-axis** = SALARY\_FROM.
    - Group by EMPLOYMENT\_TYPE\_NAME.
  + Customize colors, fonts, and styles.
* **Explanation:** Write two sentences about what the graph reveals.

pdf = df.select("NAICS2\_NAME", "SALARY").toPandas()  
fig = px.box(pdf, x="NAICS2\_NAME", y="SALARY", title="Salary Distribution by Industry",  
 color\_discrete\_sequence=["#EF553B"])  
fig.update\_layout(font\_family="Arial", title\_font\_size=16,  
 height=900,  
 width=1100)  
# rotate x-axis label for readability  
fig.update\_xaxes(tickangle=45, tickfont=dict(size=12))  
fig.show()

Unable to display output for mime type(s): application/vnd.plotly.v1+json, text/html