Assignment 03

Makenzie Howard

September 22, 2025

Load the Dataset

```
import pandas as pd
import plotly.express as px
import plotly.io as pio
from pyspark.sql import SparkSession
import re
import numpy as np
import plotly.graph_objects as go
from pyspark.sql.functions import col, split, explode, regexp_replace, transform, when
from pyspark.sql import functions as F
from pyspark.sql.functions import col, monotonically_increasing_id

np.random.seed(42)

pio.renderers.default = "notebook"

# Initialize Spark Session
spark = SparkSession.builder.appName("LightcastData").getOrCreate()

# Load Data
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","t
df.createOrReplaceTempView("job_postings")

# Show Schema and Sample Data
print("---This is Diagnostic check, No need to print it in the final doc---")

df.printSchema() # comment this line when rendering the submission
df.show(5)
```

```
[Stage 46:>                                                         (0 + 1) / 1]
```

```
---This is Diagnostic check, No need to print it in the final doc---
root
 |-- ID: string (nullable = true)
 |-- LAST_UPDATED_DATE: string (nullable = true)
 |-- LAST_UPDATED_TIMESTAMP: timestamp (nullable = true)
 |-- DUPLICATES: integer (nullable = true)
 |-- POSTED: string (nullable = true)
 |-- EXPIRED: string (nullable = true)
 |-- DURATION: integer (nullable = true)
 |-- SOURCE_TYPES: string (nullable = true)
 |-- SOURCES: string (nullable = true)
 |-- URL: string (nullable = true)
 |-- ACTIVE_URLS: string (nullable = true)
 |-- ACTIVE_SOURCES_INFO: string (nullable = true)
 |-- TITLE_RAW: string (nullable = true)
 |-- BODY: string (nullable = true)
 |-- MODELED_EXPIRED: string (nullable = true)
 |-- MODELED_DURATION: integer (nullable = true)
 |-- COMPANY: integer (nullable = true)
 |-- COMPANY_NAME: string (nullable = true)
 |-- COMPANY_RAW: string (nullable = true)
 |-- COMPANY_IS_STAFFING: boolean (nullable = true)
 |-- EDUCATION_LEVELS: string (nullable = true)
 |-- EDUCATION_LEVELS_NAME: string (nullable = true)
 |-- MIN_EDULEVELS: integer (nullable = true)
 |-- MIN_EDULEVELS_NAME: string (nullable = true)
 |-- MAX_EDULEVELS: integer (nullable = true)
 |-- MAX_EDULEVELS_NAME: string (nullable = true)
 |-- EMPLOYMENT_TYPE: integer (nullable = true)
 |-- EMPLOYMENT_TYPE_NAME: string (nullable = true)
 |-- MIN_YEARS_EXPERIENCE: integer (nullable = true)
 |-- MAX_YEARS_EXPERIENCE: integer (nullable = true)
 |-- IS_INTERNSHIP: boolean (nullable = true)
 |-- SALARY: integer (nullable = true)
 |-- REMOTE_TYPE: integer (nullable = true)
 |-- REMOTE_TYPE_NAME: string (nullable = true)
 |-- ORIGINAL_PAY_PERIOD: string (nullable = true)
 |-- SALARY_TO: integer (nullable = true)
 |-- SALARY_FROM: integer (nullable = true)
 |-- LOCATION: string (nullable = true)
 |-- CITY: string (nullable = true)
 |-- CITY_NAME: string (nullable = true)
 |-- COUNTY: integer (nullable = true)
```

```
|-- COUNTY_NAME: string (nullable = true)
|-- MSA: integer (nullable = true)
|-- MSA_NAME: string (nullable = true)
|-- STATE: integer (nullable = true)
|-- STATE_NAME: string (nullable = true)
|-- COUNTY_OUTGOING: integer (nullable = true)
|-- COUNTY_NAME_OUTGOING: string (nullable = true)
|-- COUNTY_INCOMING: integer (nullable = true)
|-- COUNTY_NAME_INCOMING: string (nullable = true)
|-- MSA_OUTGOING: integer (nullable = true)
|-- MSA_NAME_OUTGOING: string (nullable = true)
|-- MSA_INCOMING: integer (nullable = true)
|-- MSA_NAME_INCOMING: string (nullable = true)
|-- NAICS2: integer (nullable = true)
|-- NAICS2_NAME: string (nullable = true)
|-- NAICS3: integer (nullable = true)
|-- NAICS3_NAME: string (nullable = true)
|-- NAICS4: integer (nullable = true)
|-- NAICS4_NAME: string (nullable = true)
|-- NAICS5: integer (nullable = true)
|-- NAICS5_NAME: string (nullable = true)
|-- NAICS6: integer (nullable = true)
|-- NAICS6_NAME: string (nullable = true)
|-- TITLE: string (nullable = true)
|-- TITLE_NAME: string (nullable = true)
|-- TITLE_CLEAN: string (nullable = true)
|-- SKILLS: string (nullable = true)
|-- SKILLS_NAME: string (nullable = true)
|-- SPECIALIZED_SKILLS: string (nullable = true)
|-- SPECIALIZED_SKILLS_NAME: string (nullable = true)
|-- CERTIFICATIONS: string (nullable = true)
|-- CERTIFICATIONS_NAME: string (nullable = true)
|-- COMMON_SKILLS: string (nullable = true)
|-- COMMON_SKILLS_NAME: string (nullable = true)
|-- SOFTWARE_SKILLS: string (nullable = true)
|-- SOFTWARE_SKILLS_NAME: string (nullable = true)
|-- ONET: string (nullable = true)
|-- ONET_NAME: string (nullable = true)
|-- ONET_2019: string (nullable = true)
|-- ONET_2019_NAME: string (nullable = true)
|-- CIP6: string (nullable = true)
|-- CIP6_NAME: string (nullable = true)
|-- CIP4: string (nullable = true)
```

```
|-- CIP4_NAME: string (nullable = true)
|-- CIP2: string (nullable = true)
|-- CIP2_NAME: string (nullable = true)
|-- SOC_2021_2: string (nullable = true)
|-- SOC_2021_2_NAME: string (nullable = true)
|-- SOC_2021_3: string (nullable = true)
|-- SOC_2021_3_NAME: string (nullable = true)
|-- SOC_2021_4: string (nullable = true)
|-- SOC_2021_4_NAME: string (nullable = true)
|-- SOC_2021_5: string (nullable = true)
|-- SOC_2021_5_NAME: string (nullable = true)
|-- LOT_CAREER_AREA: integer (nullable = true)
|-- LOT_CAREER_AREA_NAME: string (nullable = true)
|-- LOT_OCCUPATION: integer (nullable = true)
|-- LOT_OCCUPATION_NAME: string (nullable = true)
|-- LOT_SPECIALIZED_OCCUPATION: integer (nullable = true)
|-- LOT_SPECIALIZED_OCCUPATION_NAME: string (nullable = true)
|-- LOT_OCCUPATION_GROUP: integer (nullable = true)
|-- LOT_OCCUPATION_GROUP_NAME: string (nullable = true)
|-- LOT_V6_SPECIALIZED_OCCUPATION: integer (nullable = true)
|-- LOT_V6_SPECIALIZED_OCCUPATION_NAME: string (nullable = true)
|-- LOT_V6_OCCUPATION: integer (nullable = true)
|-- LOT_V6_OCCUPATION_NAME: string (nullable = true)
|-- LOT_V6_OCCUPATION_GROUP: integer (nullable = true)
|-- LOT_V6_OCCUPATION_GROUP_NAME: string (nullable = true)
|-- LOT_V6_CAREER_AREA: integer (nullable = true)
|-- LOT_V6_CAREER_AREA_NAME: string (nullable = true)
|-- SOC_2: string (nullable = true)
|-- SOC_2_NAME: string (nullable = true)
|-- SOC_3: string (nullable = true)
|-- SOC_3_NAME: string (nullable = true)
|-- SOC_4: string (nullable = true)
|-- SOC_4_NAME: string (nullable = true)
|-- SOC_5: string (nullable = true)
|-- SOC_5_NAME: string (nullable = true)
|-- LIGHTCAST_SECTORS: string (nullable = true)
|-- LIGHTCAST_SECTORS_NAME: string (nullable = true)
|-- NAICS_2022_2: integer (nullable = true)
|-- NAICS_2022_2_NAME: string (nullable = true)
|-- NAICS_2022_3: integer (nullable = true)
|-- NAICS_2022_3_NAME: string (nullable = true)
|-- NAICS_2022_4: integer (nullable = true)
|-- NAICS_2022_4_NAME: string (nullable = true)
```

```
 |-- NAICS_2022_5: integer (nullable = true)
 |-- NAICS_2022_5_NAME: string (nullable = true)
 |-- NAICS_2022_6: integer (nullable = true)
 |-- NAICS_2022_6_NAME: string (nullable = true)

+--------------------+----------------+----------------------+----------
+--------+---------+--------+--------------------+--------------------+--------------------
+----------+------------------+--------------------+--------------------
+---------------+---------------+--------+--------------------+----------
+-------------------+---------------+--------------------+-------------
+-------------------+-------------+------------------+--------------+--------------------
+--------------------+--------------------+-------------+------+----------
+----------------+------------------+---------+----------+--------------------
+--------------------+-------------+------+--------------+-----+--------------------
+-----+----------+--------------+--------------------+--------------+--------------------
+------------+--------------------+------------+--------------------+------
+--------------------+------+--------------------+------+--------------------
+------+--------------------+------+--------------------+------------------
+-------------------+--------------------+--------------------+--------------------
+-------------------+----------------------+--------------------+--------------------
+--------------------+--------------------+--------------------+--------------------
+----------+--------------------+----------+--------------------+--------------------
+--------------------+--------------------+--------------------+--------------------
+--------------------+----------+--------------------+----------+--------------------
+----------+---------------+----------+---------------+---------------+--------------------
+--------------+--------------------+--------------------------+----------------------------
+--------------------+--------------------------+-----------------------------
+-----------------------------------+-----------------+----------------------
+------------------------+-----------------------------+------------------
+------------------------+-------+--------------------+-------+--------------------
+-------+---------------+-------+---------------+-----------------+----------------------
+------------+--------------------+------------+--------------------+------------
+--------------------+------------+--------------------+------------+--------------------
+
|                  ID|LAST_UPDATED_DATE|LAST_UPDATED_TIMESTAMP|DUPLICATES|  POSTED|  EXPIRED
+--------------------+-----------------+----------------------+----------
+--------+---------+--------+--------------------+--------------------+--------------------
+----------+------------------+--------------------+--------------------
+---------------+---------------+--------+--------------------+----------
+-------------------+---------------+--------------------+-------------
+-------------------+-------------+------------------+--------------+--------------------
+--------------------+--------------------+-------------+------+----------
+----------------+------------------+---------+----------+--------------------
```

```
+--------------------+------------+------+-------------+-----+--------------------
+-----+----------+---------------+--------------------+---------------+--------------------
+-----------+--------------------+-----------+--------------------+------
+--------------------+------+--------------------+------+--------------------
+------+--------------------+------+--------------------+-----------------
+--------------------+--------------------+--------------------+--------------------
+--------------------+----------------------+--------------------+--------------------
+--------------------+--------------------+--------------------+--------------------
+----------+--------------------+----------+--------------------+--------------------
+--------------------+-------------------+--------------------+--------------------
+--------------------+---------+--------------------+---------+--------------------
+----------+---------------+----------+---------------+---------------+--------------------
+-------------+--------------------+--------------------------+----------------------------
+--------------------+--------------------------+-----------------------------
+----------------------------------+-----------------+----------------------
+------------------------+----------------------------+------------------
+-------+--------------------+-------+--------------------+-------+--------------------
+-------+--------------+-------+--------------+-----------------+----------------------
+-----------+--------------------+-----------+--------------------+------------
+--------------------+-----------+--------------------+-----------+--------------------
+
|1f57d95acf4dc67ed...|         9/6/2024|  2024-09-06 20:32:...|         0|6/2/2024| 6/8/2024
May-2024\n\nEn...|       6/8/2024|               6|  894731|          Murphy USA| Murphy USA
time (> 32 h...|                   2|                   2|        false|  NULL|          0|
2051.01|Business Intellig...|15-2051.01|Business Intellig...|[\n  "45.0601",\n...|[\n  "Econ
0000|Computer and Math...|   15-2000|Mathematical Scie...|   15-2050|Data Scientists|   15-
2051|Data Scientists|             23|Information Techn...|        231010|Business Intellig..
0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-
2051|Data Scientists|        [\n  7\n]|  [\n  "Artificial ...|          44|        Retail Tra
|0cb072af26757b6c4...|         8/2/2024|  2024-08-02 17:08:...|         0|6/2/2024| 8/1/2024
time (> 32 h...|                   3|                   3|        false|  NULL|          1|
Watervill...|   23|     Maine|          23011|        Kennebec, ME|          23011|        K
Watervill...|       12300|Augusta-Watervill...|    56|Administrative an...|   561|Administrat
2051.01|Business Intellig...|15-2051.01|Business Intellig...|                  []|
0000|Computer and Math...|   15-2000|Mathematical Scie...|   15-2050|Data Scientists|   15-
2051|Data Scientists|             23|Information Techn...|        231010|Business Intellig..
0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-
2051|Data Scientists|             NULL|                  NULL|          56|Administrative an
|85318b12b3331fa49...|         9/6/2024|  2024-09-06 20:32:...|         1|6/2/2024| 7/7/2024
time (> 32 h...|                   5|                NULL|        false|  NULL|          0|
Fort Worth...|   48|     Texas|          48113|          Dallas, TX|          48113|
Fort Worth...|       19100|Dallas-Fort Worth...|    52|Finance and Insur...|   524|Insurance
2051.01|Business Intellig...|15-2051.01|Business Intellig...|                  []|
```

```
0000|Computer and Math...|   15-2000|Mathematical Scie...|   15-2050|Data Scientists|   15-
2051|Data Scientists|             23|Information Techn...|        231113|Data / Data Minin..
0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-
2051|Data Scientists|             NULL|                  NULL|          52|Finance and Insur
|1b5c3941e54a1889e...|         9/6/2024|  2024-09-06 20:32:...|         1|6/2/2024|7/20/2024
time (> 32 h...|                   3|                NULL|        false|  NULL|          0|
Mesa-Chan...|    4|   Arizona|           4013|        Maricopa, AZ|           4013|        Ma
Mesa-Chan...|       38060|Phoenix-Mesa-Chan...|    52|Finance and Insur...|   522|Credit Inte
2051.01|Business Intellig...|15-2051.01|Business Intellig...|                  []|
0000|Computer and Math...|   15-2000|Mathematical Scie...|   15-2050|Data Scientists|   15-
2051|Data Scientists|             23|Information Techn...|        231113|Data / Data Minin..
0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-
2051|Data Scientists|        [\n  6\n]|  [\n  "Data Privac...|          52|Finance and Insur
|cb5ca25f02bdf25c1...|        6/19/2024|   2024-06-19 07:00:00|         0|6/2/2024|6/17/2024
time / full-...|                NULL|                NULL|        false| 92500|          0|
2051.01|Business Intellig...|15-2051.01|Business Intellig...|                  []|
0000|Computer and Math...|   15-2000|Mathematical Scie...|   15-2050|Data Scientists|   15-
2051|Data Scientists|             23|Information Techn...|        231010|Business Intellig..
0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-
2051|Data Scientists|             NULL|                  NULL|          99|Unclassified Indu
+--------------------+-----------------+----------------------+----------
+--------+---------+--------+-------------------+--------------------+--------------------
+----------+------------------+--------------------+--------------------
+--------------+---------------+--------+--------------------+----------
+------------------+--------------+--------------------+-------------
+------------------+-------------+------------------+--------------+--------------------
+--------------------+--------------------+-------------+------+----------
+----------------+------------------+---------+----------+--------------------
+--------------------+-------------+------+-------------+-----+--------------------
+-----+----------+--------------+--------------------+--------------+--------------------
+-----------+--------------------+------------+--------------------+------
+--------------------+------+--------------------+------+--------------------
+------+--------------------+------+--------------------+------------------
+------------------+--------------------+--------------------+--------------------
+--------------------+----------------------+--------------------+--------------------
+--------------------+--------------------+--------------------+--------------------
+----------+--------------------+----------+--------------------+--------------------
+--------------------+--------------------+--------------------+--------------------
+--------------------+----------+--------------------+----------+--------------------
+----------+---------------+----------+---------------+---------------+--------------------
+-------------+--------------------+--------------------------+-----------------------------
+--------------------+-------------------------+----------------------------
+-----------------------------------+-----------------+----------------------
```

```
+----------------------+---------------------------+-----------------
+----------------------+-------+--------------------+-------+--------------------
+-------+--------------+-------+--------------+----------------+----------------------
+-----------+--------------------+-----------+--------------------+------------
+--------------------+-----------+--------------------+-----------+--------------------
+
only showing top 5 rows
```

1 Data Preparation

```
# Step 1: Casting salary and experience columns
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, median_salary)

# Step 3: Imputing missing salaries, but not experience
df = df.fillna({
    "SALARY_FROM" : median_from,
    "SALARY_TO" : median_to,
    "SALARY" : median_salary
})
df_filtered = df.filter(
    (F.col("SALARY") > 0) &
    F.col("EMPLOYMENT_TYPE_NAME").isNotNull() &
```

```
    (F.trim(F.col("EMPLOYMENT_TYPE_NAME")) != F.lit("")) &
    (F.lower(F.trim(F.col("EMPLOYMENT_TYPE_NAME"))) != F.lit("none"))
)

pdf = (
    df_filtered
      .select("EMPLOYMENT_TYPE_NAME", "SALARY")
      .toPandas()
)


# 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",
  "median_salary"
  "LOT_V6_SPECIALIZED_OCCUPATION_NAME",
]

df_selected = df.select(*[c for c in export_cols if c in df.columns])

# Step 7: Saving to CSV
pdf = df_selected.toPandas()   # OK for small/medium data
pdf.to_csv("./data/lightcast_cleaned.csv", index=False)

print("Data Cleaning Complete. Rows retained:", len(pdf))
```

```
[Stage 48:>                                                         (0 + 1) / 1]

Medians: 87295.0 130042.0 115024.0

[Stage 51:>                                                         (0 + 1) / 1]

Data Cleaning Complete. Rows retained: 72498
```

2 Salary Distribution by Employment Type

```
# Salary Distribution by Industry and Employment Type

import os, re
import pandas as pd
import plotly.express as px
from IPython.display import display

# Ensure output folder exists
os.makedirs("output", exist_ok=True)

# 1) Filter out missing or zero salary values and bring to pandas
pdf = (
    df.filter(df["SALARY"] > 0)    # PySpark filter (assumes SALARY is numeric)
      .select("EMPLOYMENT_TYPE_NAME", "SALARY")
      .toPandas()
)

# 2) Clean employment type names
pdf["EMPLOYMENT_TYPE_NAME"] = (
    pdf["EMPLOYMENT_TYPE_NAME"]
      .astype(str)
      .apply(lambda x: re.sub(r"[^\x00-\x7F]+", "", x))
)

# Make sure salary is numeric
pdf["SALARY"] = pd.to_numeric(pdf["SALARY"], errors="coerce")

# If a stray pluralized column exists from earlier runs
if "EMPLOYMENT_TYPE_NAMES" in pdf.columns:
    pdf = pdf.drop(columns=["EMPLOYMENT_TYPE_NAMES"])

# 3) Compute median salary for sorting
median_salaries = pdf.groupby("EMPLOYMENT_TYPE_NAME", dropna=False)["SALARY"].median()

# 4) Sort employment types based on median salary (descending)
sorted_employment_types = median_salaries.sort_values(ascending=False).index

# 5) Apply sorted categories
pdf["EMPLOYMENT_TYPE_NAME"] = pd.Categorical(
```

```
    pdf["EMPLOYMENT_TYPE_NAME"],
    categories=sorted_employment_types,
    ordered=True
)

# 6) Create box plot
fig = px.box(
    pdf,
    x="EMPLOYMENT_TYPE_NAME",
    y="SALARY",
    title="Salary Distribution by Employment Type",
    points="all",
    boxmode="group",

)
# Force black for markers and box lines
fig.update_traces(marker_color="red", line_color="black")

# 7) Improve layout, font styles, and axis labels
fig.update_layout(
    title=dict(text="Salary Distribution by Employment Type",
               font=dict(size=30, family="Helvetica", color="black")),
    xaxis=dict(
        showline=True, linewidth=2, linecolor="black", mirror=True, showgrid=False,
        categoryorder="array", categoryarray=list(sorted_employment_types)
    ),
    yaxis=dict(
        title=dict(text="Salary (K $)", font=dict(size=24, family="Helvetica", color="black"
        tickvals=[0, 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450000, 5
        ticktext=["0","50K","100K","150K","200K","250K","300K","350K","400K","450K","500K"],
        tickfont=dict(size=18, family="Helvetica", color="black"),
        showline=True, linewidth=2, linecolor="black", mirror=True,
        showgrid=True, gridcolor="lightgray", gridwidth=0.5
    ),
    font=dict(family="Helvetica", size=16, color="black"),
    boxgap=0.7, boxmode="group",
    plot_bgcolor="white", paper_bgcolor="white",
    showlegend=False, height=500, width=850,
)

display(fig)
```

```
# Save SVG
try:
    # pip install kaleido
    fig.write_image("output/Q1.svg", width=850, height=500, scale=1)
except Exception:
    fig.write_html("output/Q1.html", include_plotlyjs="cdn")
```

```
[Stage 53:>                                                         (0 + 1) / 1]
```

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

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

Explanation This table represents salary distributions by employment type: full time, part-time, or a hybrid of both. According to the table, full-time employees make the highest salaries with part-time employees making the least. Due to its hybrid nature, full-time/part-time employees experience the broadest range of salaries.

3 Salary Distribution by Industry

```
# Show table + interactive box chart

import plotly.express as px
from IPython.display import display, HTML
import pandas as pd, os

# 1) Data to pandas
pdf = df.select("NAICS2_NAME", "SALARY").toPandas()

# 1a) Force a visible HTML table preview
display(HTML(pdf.head(15).to_html(index=False)))

# 2) Plotly box chart
fig = px.box(
    pdf,
    x="NAICS2_NAME",
    y="SALARY",
    title="Salary Distribution by Industry",
```

```
    color_discrete_sequence=["red"]
)
fig.update_layout(font_family="Helvetica", title_font_size=16)
fig.update_xaxes(tickangle=45, tickfont=dict(size=12))

fig.show()

os.makedirs("output", exist_ok=True)
fig.write_html("output/naics_salary_box.html", include_plotlyjs="cdn")
```

```
[Stage 54:>                                                         (0 + 1) / 1]
```

NAICS2_NAME	SALARY
Retail Trade	115024.0
Administrative and Support and Waste Management and Remediation Services	115024.0
Finance and Insurance	115024.0
Finance and Insurance	115024.0
Unclassified Industry	92500.0
Information	110155.0
Manufacturing	115024.0
Finance and Insurance	115024.0
Unclassified Industry	115024.0
Professional, Scientific, and Technical Services	92962.0
Wholesale Trade	107645.0
Administrative and Support and Waste Management and Remediation Services	115024.0
Finance and Insurance	115024.0
Professional, Scientific, and Technical Services	192800.0
Finance and Insurance	81286.0

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

Explanation This table represents salary distribution by industry. According to the table, Healthcare Services, Waste Management Services, and Information Services have the largest ranges of salaries while also claiming the highest salaries.

4 Salary Analysis by ONET Occupation Type (Bubble Chart)

```
#Step 1: spark SQL - Median Salary and job count per TITLE_NAME
salary_analysis = spark.sql("""
  SELECT
    LOT_OCCUPATION_NAME AS Occupation_name,
    PERCENTILE(SALARY, 0.5) AS Median_Salary,
    COUNT(*) AS Job_Postings
  FROM job_postings
  GROUP BY LOT_OCCUPATION_NAME
  ORDER BY Job_Postings DESC
  LIMIT 10
""")

#step 2: convert to pandas dataframe
salary_pd = salary_analysis.toPandas()
salary_pd.head()

#Step 3: Bubble chart using plotly
import plotly.express as px

fig = px.scatter(
    salary_pd,
    x="Occupation_name",
    y="Median_Salary",
    size="Job_Postings",
    title="Salary Analysis by Lot Occupation Type (Bubble Chart)",
    labels={
          "LOT_OCCUPATION_NAME": "Lot Occupation",
          "Median_Salary": "Median Salary",
          "Job_Postings": "Number of Job Postings",
    },
    hover_name="Occupation_name",
    size_max=60,
    width=1000,
    height=600,
    color="Job_Postings",
    color_continuous_scale="Plasma"
)

#Step 4: Layout customization
fig.update_layout(
    font_family="Helvetica",
```

```
    font_size=14,
    title_font_size=25,
    xaxis_title="Lot Occupation",
    yaxis_title="Median Salary",
    plot_bgcolor="white",
    xaxis=dict(
        tickangle=-45,
        showline=True,
        linecolor="red"
    ),
    yaxis=dict(
        showline=True,
        linecolor="red"
    )
)

fig.show()

# save artifact without kaleido
import os
os.makedirs("output", exist_ok=True)

try:
    fig.write_image("output/Q7.svg", width=1000, height=600, scale=1)
except Exception:
    # no kaleido? save interactive HTML instead
    fig.write_html("output/Q7.html", include_plotlyjs="cdn")
    print("Saved interactive HTML fallback to output/Q7.html (no kaleido).")
```

```
[Stage 55:>                                                         (0 + 1) / 1]
```

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

```
Saved interactive HTML fallback to output/Q7.html (no kaleido).
```

Explanation This bubble chart shows the top ten job titles by number of job postings. We can see that Business Intelligence Analysts and Data Mining Analysts, although the most abundant, do not have the highest salaries in comparison to more technical groups such as Computer System Engineers. The least amount of job postings lie in the business analyst and market research analyst positions, while also paying the lowest salaries.

5 Salary by Education Level

```
from pyspark.sql.functions import when, col
import plotly.express as px
import pandas as pd
import os, re
from IPython.display import display, HTML

SHOW_PREVIEW = True

# Build groups
lower_deg  = ["Bachelor's", "Associate", "GED", "No Education Listed", "High School"]
higher_deg = ["Master's degree", "PHD or professional degree"]
lower_pat  = "(?i)(" + "|".join(map(re.escape, lower_deg)) + ")"
higher_pat = "(?i)(" + "|".join(map(re.escape, higher_deg)) + ")"

df2 = (
    df.withColumn(
        "EDU_GROUP",
        when(col("EDUCATION_LEVELS_NAME").rlike(lower_pat),  "Bachelor's or lower")
        .when(col("EDUCATION_LEVELS_NAME").rlike(higher_pat), "Master's or PhD")
        .otherwise("Other")
    )
    .withColumn("MAX_YEARS_EXPERIENCE", col("MAX_YEARS_EXPERIENCE").cast("float"))
    .withColumn("Average_Salary",       col("Average_Salary").cast("float"))
    .filter(
        col("MAX_YEARS_EXPERIENCE").isNotNull() &
        col("Average_Salary").isNotNull() &
        (col("MAX_YEARS_EXPERIENCE") > 0) &
        (col("Average_Salary") > 0)
    )
)

df_filtered = df2.filter(col("EDU_GROUP").isin("Bachelor's or lower", "Master's or PhD"))

# To pandas
df_pd = df_filtered.toPandas()

# Optional preview table
if SHOW_PREVIEW:
    cols_to_show = [c for c in ["EDU_GROUP","MAX_YEARS_EXPERIENCE","Average_Salary",
```

```
                                    "LOT_V6_SPECIALIZED_OCCUPATION_NAME"] if c in df_pd.columns]
    display(HTML(df_pd.loc[:, cols_to_show].head(10).to_html(index=False)))

# Scatter plot
hover_cols = [c for c in ["LOT_V6_SPECIALIZED_OCCUPATION_NAME"] if c in df_pd.columns]
fig1 = px.scatter(
    df_pd,
    x="MAX_YEARS_EXPERIENCE",
    y="Average_Salary",
    color="EDU_GROUP",
    hover_data=hover_cols,
    title="<b>Experience vs Salary by Education Level</b>",
    opacity=0.7,
    color_discrete_sequence=["blue", "red"]
)
fig1.update_traces(marker=dict(size=7, line=dict(width=1, color="black")))
fig1.update_layout(
    plot_bgcolor="#f9f9f9",
    paper_bgcolor="#FFF5DC",
    font=dict(family="Helvetica", size=14),
    title_font=dict(size=22),
    xaxis_title="Years of Experience",
    yaxis_title="Average Salary (USD)",
    legend_title="Education Group",
    hoverlabel=dict(bgcolor="white", font_size=13, font_family="Helvetica"),
    margin=dict(t=70, b=60, l=60, r=60),
    xaxis=dict(gridcolor="lightgrey", tickmode="linear", dtick=1),
    yaxis=dict(gridcolor="lightgrey")
)

#  Return fig1 so Quarto embeds THIS chart (don't call fig.show())
fig1.show()

# Save interactive artifact
os.makedirs("output", exist_ok=True)
fig1.write_html("output/q_1a_Experience_vs_Salary_by_Education_Level.html", include_plotlyjs=
```

```
[Stage 58:>                                                         (0 + 1) / 1]
```

EDU_GROUP	MAX_YEARS_EXPERIENCE	Average_Salary	LOT_V6_SPECIALIZED_OCCUP
Bachelor's or lower	2.0	108668.5	General ERP Analyst / Consultant
Bachelor's or lower	3.0	108668.5	Oracle Consultant / Analyst
Bachelor's or lower	7.0	108668.5	General ERP Analyst / Consultant
Bachelor's or lower	2.0	92962.0	Data Analyst
Bachelor's or lower	5.0	108668.5	Data Analyst
Bachelor's or lower	3.0	108668.5	Oracle Consultant / Analyst
Bachelor's or lower	8.0	165000.0	Enterprise Architect
Bachelor's or lower	2.0	108668.5	Data Analyst
Bachelor's or lower	2.0	75026.0	Oracle Consultant / Analyst
Bachelor's or lower	1.0	108668.5	Data Analyst

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

Explanation This table analyzes salary by education level. It indicates that there is not an extreme discrepancy surrounding bachelor's degree holders versus higher education holders, but those with higher education beyond a bachelor's degree do tend to have higher salaries. However, years of service does not seem to generate a higher salary. Alternatively, those who have the highest salaries are earning it during the 5-10 year mark of experience in their careers. Years of experience does not necessarily translate to higher salaries, but higher education can.

6 Salary by Remote Work Type

```
from pyspark.sql.functions import col, when, lower, trim
import plotly.express as px
import numpy as np, pandas as pd, os
from IPython.display import Markdown, display

os.makedirs("output", exist_ok=True)
np.random.seed(42)

# 1) Build groups: Remote / Hybrid / Onsite (Onsite includes None/blank)
df_remote = (
    df.withColumn(
        "REMOTE_GROUP",
        when(lower(trim(col("REMOTE_TYPE_NAME"))) == "remote", "Remote")
        .when(lower(trim(col("REMOTE_TYPE_NAME"))) == "hybrid", "Hybrid")
        .otherwise("Onsite")
```

```
    )
    .withColumn("MAX_YEARS_EXPERIENCE", col("MAX_YEARS_EXPERIENCE").cast("float"))
    .withColumn("Average_Salary",       col("Average_Salary").cast("float"))
    .filter(
        col("MAX_YEARS_EXPERIENCE").isNotNull() &
        col("Average_Salary").isNotNull() &
        (col("MAX_YEARS_EXPERIENCE") > 0) &
        (col("Average_Salary") > 0)
    )
    .select("REMOTE_GROUP", "MAX_YEARS_EXPERIENCE", "Average_Salary",
            "LOT_V6_SPECIALIZED_OCCUPATION_NAME")
)

# 2) To pandas
pdf = df_remote.toPandas()
pdf["EXP_JITTER"] = (
    pdf["MAX_YEARS_EXPERIENCE"].astype(float)
    + np.random.uniform(-0.15, 0.15, len(pdf))
).clip(lower=0)

# 3) y axis
def _round_up(x, base=50000):
    return int(base * np.ceil(max(x, 1) / base))
ymax = _round_up(float(pdf["Average_Salary"].max()) if len(pdf) else 50000, 50000)
yticks = list(range(0, ymax + 1, 50000))
yticktext = [("0" if v == 0 else f"{v//1000}k") for v in yticks]

# 4) colors
cmap = {"Onsite": "#4C6EF5", "Remote": "#E8590C", "Hybrid": "#2B8A3E"}  # blue, orange, gree

# 5) Single consolidated scatter
fig = px.scatter(
    pdf,
    x="EXP_JITTER",
    y="Average_Salary",
    color="REMOTE_GROUP",
    hover_data=["LOT_V6_SPECIALIZED_OCCUPATION_NAME", "MAX_YEARS_EXPERIENCE"],
    title="Experience vs Salary by Remote Work Type",
    opacity=0.7,
    color_discrete_map=cmap
)
```

```
# marker outlines + layout
fig.update_traces(marker=dict(size=8, line=dict(width=1, color="black")))
fig.update_layout(
    font=dict(family="Helvetica", size=16, color="#1c2a39"),
    title_font=dict(size=34),
    legend_title="Remote Work Type",
    legend=dict(x=1.02, y=1, bgcolor="rgba(255,255,255,0.7)"),
    xaxis_title="Years of Experience",
    yaxis_title="Average Salary (USD)",
    xaxis=dict(gridcolor="#e0e0e0", tickmode="linear", dtick=1),
    yaxis=dict(gridcolor="#e0e0e0", tickvals=yticks, ticktext=yticktext),
    plot_bgcolor="#F3F5F7",     # inner panel
    paper_bgcolor="#EAF6E8",    # soft green page
    margin=dict(t=90, r=140, b=70, l=80),
)

fig.show()

fig.write_html("output/remote_scatter_all.html", include_plotlyjs="cdn")
```

```
[Stage 59:>                                                         (0 + 1) / 1]
```

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

Explanation This table compares experience versus salary in onsite and remote workers. Onsite and remote workers do not have much difference in salary, however, there are more remote workers that have less experience than onsite workers with more experience. This is most likely caused by several different external factors and would require further analysis.