Module 04: Lab 01

Visual Reporting and Storytelling

Your Name

November 21, 2024

# Objectives

By the end of this lab, you will: 1. Load and analyze the **Lightcast dataset** in **Spark DataFrame**. 2. Create **five easy and three medium-complexity visualizations** using **Plotly**. 3. Explore **salary distributions, employment trends, and job postings**. 4. Analyze **skills in relation to NAICS/SOC/ONET codes and salaries**. 5. Customize **colors, fonts, and styles** in all visualizations (**default themes result in a 2.5-point deduction**). 6. Follow **best practices for reporting on data communication**.

# Step 1: Load the Dataset

import pandas as pd  
import plotly.express as px  
import plotly.io as pio  
pio.renderers.default = "svg"  
from pyspark.sql import SparkSession  
import re  
import plotly.graph\_objects as go  
from pyspark.sql.functions import col, split, explode, regexp\_replace, transform  
from pyspark.sql import functions as F  
  
# Initialize Spark Session  
spark = SparkSession.builder.appName("LightcastData").getOrCreate()  
  
# Load Data  
df = spark.read.option("header", "true").option("inferSchema", "true").option("multiLine","true").option("escape", "\"").csv("./data/lightcast\_job\_postings.csv")  
df.createOrReplaceTempView("jobs")  
# Show Schema and Sample Data  
df.printSchema()  
df.show(5)

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|DURATION| SOURCE\_TYPES| SOURCES| URL|ACTIVE\_URLS|ACTIVE\_SOURCES\_INFO| TITLE\_RAW| BODY|MODELED\_EXPIRED|MODELED\_DURATION| COMPANY| COMPANY\_NAME|COMPANY\_RAW|COMPANY\_IS\_STAFFING|EDUCATION\_LEVELS|EDUCATION\_LEVELS\_NAME|MIN\_EDULEVELS| MIN\_EDULEVELS\_NAME|MAX\_EDULEVELS|MAX\_EDULEVELS\_NAME|EMPLOYMENT\_TYPE|EMPLOYMENT\_TYPE\_NAME|MIN\_YEARS\_EXPERIENCE|MAX\_YEARS\_EXPERIENCE|IS\_INTERNSHIP|SALARY|REMOTE\_TYPE|REMOTE\_TYPE\_NAME|ORIGINAL\_PAY\_PERIOD|SALARY\_TO|SALARY\_FROM| LOCATION| CITY| CITY\_NAME|COUNTY| COUNTY\_NAME| MSA| MSA\_NAME|STATE|STATE\_NAME|COUNTY\_OUTGOING|COUNTY\_NAME\_OUTGOING|COUNTY\_INCOMING|COUNTY\_NAME\_INCOMING|MSA\_OUTGOING| MSA\_NAME\_OUTGOING|MSA\_INCOMING| MSA\_NAME\_INCOMING|NAICS2| NAICS2\_NAME|NAICS3| NAICS3\_NAME|NAICS4| NAICS4\_NAME|NAICS5| NAICS5\_NAME|NAICS6| NAICS6\_NAME| TITLE| TITLE\_NAME| TITLE\_CLEAN| SKILLS| SKILLS\_NAME| SPECIALIZED\_SKILLS|SPECIALIZED\_SKILLS\_NAME| CERTIFICATIONS| CERTIFICATIONS\_NAME| COMMON\_SKILLS| COMMON\_SKILLS\_NAME| SOFTWARE\_SKILLS|SOFTWARE\_SKILLS\_NAME| ONET| ONET\_NAME| ONET\_2019| ONET\_2019\_NAME| CIP6| CIP6\_NAME| CIP4| CIP4\_NAME| CIP2| CIP2\_NAME|SOC\_2021\_2| SOC\_2021\_2\_NAME|SOC\_2021\_3| SOC\_2021\_3\_NAME|SOC\_2021\_4|SOC\_2021\_4\_NAME|SOC\_2021\_5|SOC\_2021\_5\_NAME|LOT\_CAREER\_AREA|LOT\_CAREER\_AREA\_NAME|LOT\_OCCUPATION| LOT\_OCCUPATION\_NAME|LOT\_SPECIALIZED\_OCCUPATION|LOT\_SPECIALIZED\_OCCUPATION\_NAME|LOT\_OCCUPATION\_GROUP|LOT\_OCCUPATION\_GROUP\_NAME|LOT\_V6\_SPECIALIZED\_OCCUPATION|LOT\_V6\_SPECIALIZED\_OCCUPATION\_NAME|LOT\_V6\_OCCUPATION|LOT\_V6\_OCCUPATION\_NAME|LOT\_V6\_OCCUPATION\_GROUP|LOT\_V6\_OCCUPATION\_GROUP\_NAME|LOT\_V6\_CAREER\_AREA|LOT\_V6\_CAREER\_AREA\_NAME| SOC\_2| SOC\_2\_NAME| SOC\_3| SOC\_3\_NAME| SOC\_4| SOC\_4\_NAME| SOC\_5| SOC\_5\_NAME|LIGHTCAST\_SECTORS|LIGHTCAST\_SECTORS\_NAME|NAICS\_2022\_2| NAICS\_2022\_2\_NAME|NAICS\_2022\_3| NAICS\_2022\_3\_NAME|NAICS\_2022\_4| NAICS\_2022\_4\_NAME|NAICS\_2022\_5| NAICS\_2022\_5\_NAME|NAICS\_2022\_6| NAICS\_2022\_6\_NAME|  
+--------------------+-----------------+----------------------+----------+--------+---------+--------+--------------------+--------------------+--------------------+-----------+-------------------+--------------------+--------------------+---------------+----------------+--------+--------------------+-----------+-------------------+----------------+---------------------+-------------+-------------------+-------------+------------------+---------------+--------------------+--------------------+--------------------+-------------+------+-----------+----------------+-------------------+---------+-----------+--------------------+--------------------+-------------+------+--------------+-----+--------------------+-----+----------+---------------+--------------------+---------------+--------------------+------------+--------------------+------------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------------------+-------------------+--------------------+--------------------+--------------------+--------------------+-----------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+----------+---------------+----------+---------------+---------------+--------------------+--------------+--------------------+--------------------------+-------------------------------+--------------------+-------------------------+-----------------------------+----------------------------------+-----------------+----------------------+-----------------------+----------------------------+------------------+-----------------------+-------+--------------------+-------+--------------------+-------+---------------+-------+---------------+-----------------+----------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+  
|1f57d95acf4dc67ed...| 9/6/2024| 2024-09-06 20:32:...| 0|6/2/2024| 6/8/2024| 6| [\n "Company"\n]|[\n "brassring.c...|[\n "https://sjo...| []| NULL|Enterprise Analys...|31-May-2024\n\nEn...| 6/8/2024| 6| 894731| Murphy USA| Murphy USA| false| [\n 2\n]| [\n "Bachelor's ...| 2| Bachelor's degree| NULL| NULL| 1|Full-time (> 32 h...| 2| 2| false| NULL| 0| [None]| NULL| NULL| NULL|{\n "lat": 33.20...|RWwgRG9yYWRvLCBBUg==|El Dorado, AR| 5139| Union, AR|20980| El Dorado, AR| 5| Arkansas| 5139| Union, AR| 5139| Union, AR| 20980| El Dorado, AR| 20980| El Dorado, AR| 44| Retail Trade| 441|Motor Vehicle and...| 4413|Automotive Parts,...| 44133|Automotive Parts ...|441330|Automotive Parts ...|ET29C073C03D1F86B4|Enterprise Analysts|enterprise analys...|[\n "KS126DB6T06...|[\n "Merchandisi...|[\n "KS126DB6T06...| [\n "Merchandisi...| []| []|[\n "KS126706DPF...|[\n "Mathematics...|[\n "KS440W865GC...|[\n "SQL (Progra...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...|[\n "45.0601",\n...|[\n "Economics, ...|[\n "45.06",\n ...|[\n "Economics",...|[\n "45",\n "27...|[\n "Social Scie...| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231010|Business Intellig...| 23101011| General ERP Analy...| 2310| Business Intellig...| 23101011| General ERP Analy...| 231010| Business Intellig...| 2310| Business Intellig...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| [\n 7\n]| [\n "Artificial ...| 44| Retail Trade| 441|Motor Vehicle and...| 4413|Automotive Parts,...| 44133|Automotive Parts ...| 441330|Automotive Parts ...|  
|0cb072af26757b6c4...| 8/2/2024| 2024-08-02 17:08:...| 0|6/2/2024| 8/1/2024| NULL| [\n "Job Board"\n]| [\n "maine.gov"\n]|[\n "https://job...| []| NULL|Oracle Consultant...|Oracle Consultant...| 8/1/2024| NULL| 133098|Smx Corporation L...| SMX| true| [\n 99\n]| [\n "No Educatio...| 99|No Education Listed| NULL| NULL| 1|Full-time (> 32 h...| 3| 3| false| NULL| 1| Remote| NULL| NULL| NULL|{\n "lat": 44.31...| QXVndXN0YSwgTUU=| Augusta, ME| 23011| Kennebec, ME|12300|Augusta-Watervill...| 23| Maine| 23011| Kennebec, ME| 23011| Kennebec, ME| 12300|Augusta-Watervill...| 12300|Augusta-Watervill...| 56|Administrative an...| 561|Administrative an...| 5613| Employment Services| 56132|Temporary Help Se...|561320|Temporary Help Se...|ET21DDA63780A7DC09| Oracle Consultants|oracle consultant...|[\n "KS122626T55...|[\n "Procurement...|[\n "KS122626T55...| [\n "Procurement...| []| []| []| []|[\n "BGSBF3F508F...|[\n "Oracle Busi...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231010|Business Intellig...| 23101012| Oracle Consultant...| 2310| Business Intellig...| 23101012| Oracle Consultant...| 231010| Business Intellig...| 2310| Business Intellig...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| NULL| NULL| 56|Administrative an...| 561|Administrative an...| 5613| Employment Services| 56132|Temporary Help Se...| 561320|Temporary Help Se...|  
|85318b12b3331fa49...| 9/6/2024| 2024-09-06 20:32:...| 1|6/2/2024| 7/7/2024| 35| [\n "Job Board"\n]|[\n "dejobs.org"\n]|[\n "https://dej...| []| NULL| Data Analyst|Taking care of pe...| 6/10/2024| 8|39063746| Sedgwick| Sedgwick| false| [\n 2\n]| [\n "Bachelor's ...| 2| Bachelor's degree| NULL| NULL| 1|Full-time (> 32 h...| 5| NULL| false| NULL| 0| [None]| NULL| NULL| NULL|{\n "lat": 32.77...| RGFsbGFzLCBUWA==| Dallas, TX| 48113| Dallas, TX|19100|Dallas-Fort Worth...| 48| Texas| 48113| Dallas, TX| 48113| Dallas, TX| 19100|Dallas-Fort Worth...| 19100|Dallas-Fort Worth...| 52|Finance and Insur...| 524|Insurance Carrier...| 5242|Agencies, Brokera...| 52429|Other Insurance R...|524291| Claims Adjusting|ET3037E0C947A02404| Data Analysts| data analyst|[\n "KS1218W78FG...|[\n "Management"...|[\n "ESF3939CE1F...| [\n "Exception R...|[\n "KS683TN76T7...|[\n "Security Cl...|[\n "KS1218W78FG...|[\n "Management"...|[\n "KS126HY6YLT...|[\n "Microsoft O...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231113|Data / Data Minin...| 23111310| Data Analyst| 2311| Data Analysis and...| 23111310| Data Analyst| 231113| Data / Data Minin...| 2311| Data Analysis and...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| NULL| NULL| 52|Finance and Insur...| 524|Insurance Carrier...| 5242|Agencies, Brokera...| 52429|Other Insurance R...| 524291| Claims Adjusting|  
|1b5c3941e54a1889e...| 9/6/2024| 2024-09-06 20:32:...| 1|6/2/2024|7/20/2024| 48| [\n "Job Board"\n]|[\n "disabledper...|[\n "https://www...| []| NULL|Sr. Lead Data Mgm...|About this role:\...| 6/12/2024| 10|37615159| Wells Fargo|Wells Fargo| false| [\n 99\n]| [\n "No Educatio...| 99|No Education Listed| NULL| NULL| 1|Full-time (> 32 h...| 3| NULL| false| NULL| 0| [None]| NULL| NULL| NULL|{\n "lat": 33.44...| UGhvZW5peCwgQVo=| Phoenix, AZ| 4013| Maricopa, AZ|38060|Phoenix-Mesa-Chan...| 4| Arizona| 4013| Maricopa, AZ| 4013| Maricopa, AZ| 38060|Phoenix-Mesa-Chan...| 38060|Phoenix-Mesa-Chan...| 52|Finance and Insur...| 522|Credit Intermedia...| 5221|Depository Credit...| 52211| Commercial Banking|522110| Commercial Banking|ET2114E0404BA30075|Management Analysts|sr lead data mgmt...|[\n "KS123QX62QY...|[\n "Exit Strate...|[\n "KS123QX62QY...| [\n "Exit Strate...| []| []|[\n "KS7G6NP6R6L...|[\n "Reliability...|[\n "KS4409D76NW...|[\n "SAS (Softwa...|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231113|Data / Data Minin...| 23111310| Data Analyst| 2311| Data Analysis and...| 23111310| Data Analyst| 231113| Data / Data Minin...| 2311| Data Analysis and...| 23| Information Techn...|15-0000|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...| 522|Credit Intermedia...| 5221|Depository Credit...| 52211| Commercial Banking| 522110| Commercial Banking|  
|cb5ca25f02bdf25c1...| 6/19/2024| 2024-06-19 07:00:00| 0|6/2/2024|6/17/2024| 15|[\n "FreeJobBoar...|[\n "craigslist....|[\n "https://mod...| []| NULL|Comisiones de $10...|Comisiones de $10...| 6/17/2024| 15| 0| Unclassified| LH/GM| false| [\n 99\n]| [\n "No Educatio...| 99|No Education Listed| NULL| NULL| 3|Part-time / full-...| NULL| NULL| false| 92500| 0| [None]| year| 150000| 35000|{\n "lat": 37.63...| TW9kZXN0bywgQ0E=| Modesto, CA| 6099|Stanislaus, CA|33700| Modesto, CA| 6|California| 6099| Stanislaus, CA| 6099| Stanislaus, CA| 33700| Modesto, CA| 33700| Modesto, CA| 99|Unclassified Indu...| 999|Unclassified Indu...| 9999|Unclassified Indu...| 99999|Unclassified Indu...|999999|Unclassified Indu...|ET0000000000000000| Unclassified|comisiones de por...| []| []| []| []| []| []| []| []| []| []|15-2051.01|Business Intellig...|15-2051.01|Business Intellig...| []| []| []| []| []| []| 15-0000|Computer and Math...| 15-2000|Mathematical Scie...| 15-2050|Data Scientists| 15-2051|Data Scientists| 23|Information Techn...| 231010|Business Intellig...| 23101012| Oracle Consultant...| 2310| Business Intellig...| 23101012| Oracle Consultant...| 231010| Business Intellig...| 2310| Business Intellig...| 23| Information Techn...|15-0000|Computer and Math...|15-2000|Mathematical Scie...|15-2050|Data Scientists|15-2051|Data Scientists| NULL| NULL| 99|Unclassified Indu...| 999|Unclassified Indu...| 9999|Unclassified Indu...| 99999|Unclassified Indu...| 999999|Unclassified Indu...|  
+--------------------+-----------------+----------------------+----------+--------+---------+--------+--------------------+--------------------+--------------------+-----------+-------------------+--------------------+--------------------+---------------+----------------+--------+--------------------+-----------+-------------------+----------------+---------------------+-------------+-------------------+-------------+------------------+---------------+--------------------+--------------------+--------------------+-------------+------+-----------+----------------+-------------------+---------+-----------+--------------------+--------------------+-------------+------+--------------+-----+--------------------+-----+----------+---------------+--------------------+---------------+--------------------+------------+--------------------+------------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------+--------------------+------------------+-------------------+--------------------+--------------------+--------------------+--------------------+-----------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------+----------+--------------------+----------+--------------------+----------+---------------+----------+---------------+---------------+--------------------+--------------+--------------------+--------------------------+-------------------------------+--------------------+-------------------------+-----------------------------+----------------------------------+-----------------+----------------------+-----------------------+----------------------------+------------------+-----------------------+-------+--------------------+-------+--------------------+-------+---------------+-------+---------------+-----------------+----------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+------------+--------------------+  
only showing top 5 rows

# 1. Salary Distribution by Employment Type

* Identify salary trends across different employment types.
* **Filter the dataset**
  + Remove records where **salary is missing or zero**.
* **Aggregate Data**
  + Group by **employment type** and compute salary distribution.
* **Visualize results**
  + Create a **box plot** where:
    - **X-axis** = EMPLOYMENT\_TYPE\_NAME
    - **Y-axis** = SALARY\_FROM
  + Customize **colors, fonts, and styles** to avoid a **2.5-point deduction**.
* **Explanation:** Write two sentences about what the graph reveals.

# Your Code for 1st question here  
  
# 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"].apply(lambda x: re.sub(r"[^\x00-\x7F]+", "", x))  
  
# Compute median salary for sorting  
median\_salaries = pdf.groupby("EMPLOYMENT\_TYPE\_NAME")["SALARY"].median()  
  
# 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=["black"], # Single neutral color  
 boxmode="group",  
 points="all", # Show all outliers  
)  
  
# Improve layout, font styles, and axis labels  
fig.update\_layout(  
 title=dict(  
 text="Salary Distribution by Employment Type",   
 font=dict(size=30, family="Arial", color="black", weight="bold") # Bigger & Bold Title  
 ),  
 xaxis=dict(  
 title=dict(text="Employment Type", font=dict(size=24, family="Arial", color="black", weight="bold")), # Bigger X-label  
 tickangle=0, # Rotate X-axis labels for readability  
 tickfont=dict(size=18, family="Arial", color="black", weight="bold"), # 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 (K $)", font=dict(size=24, family="Arial", color="black", weight="bold")), # Bigger Y-label  
 tickvals=[0, 50000, 100000, 150000, 200000, 250000, 300000, 350000, 400000, 450000, 500000],  
 ticktext=["0", "50K", "100K", "150K", "200K", "250K", "300K", "350K", "400K", "450K", "500K"],  
 tickfont=dict(size=18, family="Arial", color="black", weight="bold"), # 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="Arial", size=16, color="black"),  
 boxgap=0.7,   
 plot\_bgcolor="white",  
 paper\_bgcolor="white",  
 showlegend=False,   
 height=500,   
 width=850,   
)  
  
# Show the figure  
fig.show()  
fig.write\_image("output/Q1.svg", width=850, height=500, scale=1)

[Salary Distribution by Employment Type](_output/Q1.svg)

**Salary Distribution:** *Full-time employees working more than 32 hours have a higher median salary compared to part-time and mixed employment types, indicating better earning potential for full-time roles.*

**Variability in Salaries:** *The full-time category has a wider spread of salaries and a higher concentration of outliers, suggesting greater variability in earnings, possibly due to differences in experience, job roles, or industries.*

**Outliers and Salary Extremes:** *All employment types have salary outliers, but full-time employees exhibit the highest number of extreme salaries, indicating that while most employees earn within a standard range, a few earn significantly higher wages.*

# 2. Salary Distribution by Industry

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

# Your Code for 2nd question here  
  
# Load Data (Ensure df is pre-loaded with "NAICS2\_NAME" and "SALARY\_FROM")  
pdf = df.select("NAICS2\_NAME", "SALARY\_FROM").toPandas()  
  
# Convert salary to thousands (K) for better readability  
pdf["SALARY\_FROM"] = pdf["SALARY\_FROM"] / 1000  
  
# Shorten industry names for readability  
industry\_mapping = {  
 "Retail Trade": "Retail",  
 "Administrative and Support and Waste Management and Remediation Services": "Admin & Waste",  
 "Finance and Insurance": "Finance & Ins.",  
 "Unclassified Industry": "Unclassified",  
 "Information": "Info",  
 "Manufacturing": "Manuf.",  
 "Professional, Scientific, and Technical Services": "Pro. & Sci.",  
 "Wholesale Trade": "Wholesale",  
 "Educational Services": "Edu. Services",  
 "Health Care and Social Assistance": "Health & Social",  
 "Public Administration": "Public Admin.",  
 "Construction": "Constr.",  
 "Transportation and Warehousing": "Trans. & Waren.",  
 "Real Estate and Rental and Leasing": "Real Estate & Lease",  
 "Accommodation and Food Services": "Accom. & Food",  
 "Other Services (except Public Administration)": "Other Services",  
 "Management of Companies and Enterprises": "Management",  
 "Mining, Quarrying, and Oil and Gas Extraction": "Mining & Oil",  
 "Agriculture, Forestry, Fishing and Hunting": "Agri. & Fish",  
 "Utilities": "Utilities",  
 "Arts, Entertainment, and Recreation": "Arts & Rec."  
}  
  
# Apply renaming  
pdf["NAICS2\_NAME"] = pdf["NAICS2\_NAME"].map(industry\_mapping)  
  
# Compute median salary for each industry  
median\_salaries = pdf.groupby("NAICS2\_NAME")["SALARY\_FROM"].median()  
  
# Sort industries based on median salary in descending order  
sorted\_industries = median\_salaries.sort\_values(ascending=False).index  
  
# Apply sorted categories  
pdf["NAICS2\_NAME"] = pd.Categorical(  
 pdf["NAICS2\_NAME"],   
 categories=sorted\_industries,   
 ordered=True  
)  
  
# Create the ordered box plot with contrasting theme  
fig = px.box(  
 pdf,   
 x="NAICS2\_NAME",   
 y="SALARY\_FROM",   
 title="Salary Distribution by Industry (Ordered by Median Salary)",  
 labels={"NAICS2\_NAME": "Industry", "SALARY\_FROM": "Salary (K $)"},  
 width=850,   
 height=500,   
 points="outliers",   
 template="plotly\_dark",   
 color\_discrete\_sequence=["#1f77b4"]   
)  
  
# Improve layout with bigger and bold fonts, and force Y-axis to start at 0  
fig.update\_layout(  
 title=dict(  
 text="Salary Distribution by Industry (Ordered by Median Salary)",   
 font=dict(size=22, family="Arial", color="white", weight="bold")  
 ),  
 font=dict(family="Arial", size=12, color="white"),  
 xaxis=dict(  
 title=dict(text="Industry", font=dict(size=20, family="Arial", color="white", weight="bold")),   
 tickangle=-45,   
 tickfont=dict(size=12, family="Arial", color="white", weight="bold"),  
 showline=True,   
 linewidth=2,  
 linecolor="white",  
 categoryorder="array",  
 categoryarray=sorted\_industries.tolist()  
 ),  
 yaxis=dict(  
 title=dict(text="Salary (K $)", font=dict(size=20, family="Arial", color="white", weight="bold")),   
 tickfont=dict(size=12, family="Arial", color="white", weight="bold"),  
 showline=True,   
 linewidth=2,  
 linecolor="white",  
 tickformat=",",   
 gridcolor="gray",  
 zeroline=True, # Ensure the zero line is drawn  
 zerolinewidth=3, # Make it thicker  
 zerolinecolor="white", # Make zero line visible  
 range=[0, pdf["SALARY\_FROM"].max() + 50] # Force Y-axis to start at 0  
 ),  
 plot\_bgcolor="#222222",   
 paper\_bgcolor="#111111",   
 boxmode="group"   
)  
  
# Show the plot  
fig.show()  
fig.write\_image("output/Q2.svg", width=850, height=500, scale=1)

[Salary Distribution by Industry](_output/Q2.svg)

**Industry Salary Variation:** *The Information (Info) and Accommodation & Food industries have the highest median salaries, while Educational Services (Edu. Services) and Agriculture & Fish show the lowest median salaries.This suggests that jobs in tech and hospitality industries tend to offer higher pay compared to education and agriculture sectors.*

**Outliers in Salaries:** *Some industries, such as Information, Professional & Scientific Services, and Mining & Oil, have significant salary outliers, indicating that a few jobs within these sectors offer extremely high salaries compared to the median.*

**Salary Distribution Trends:** *The middle 50% salary range (interquartile range, IQR) is wider in high-paying industries, indicating a broader variation in salaries. Conversely, industries with lower median salaries tend to have a narrower IQR, suggesting more consistent pay within those fields.*

# 3. Job Posting Trends Over Time

* Analyze how job postings fluctuate over time.
* **Aggregate Data**
  + Count job postings per **posted date (POSTED)**.
* **Visualize results**
  + Create a **line chart** where:
    - **X-axis** = POSTED
    - **Y-axis** = Number of Job Postings
  + Apply custom colors and font styles.
* **Explanation:** Write two sentences about what the graph reveals.

# Your code for 3rd question here  
posted\_count = spark.sql("""  
 SELECT POSTED, COUNT(\*) AS posted\_count  
 FROM jobs  
 GROUP BY POSTED  
 ORDER BY POSTED  
""")  
# Convert to Pandas for Plotly  
pdf = posted\_count.toPandas()  
  
# Ensure POSTED is datetime type for proper x-axis formatting  
pdf["POSTED"] = pd.to\_datetime(pdf["POSTED"])  
  
# Sort just in case (if not sorted already)  
pdf = pdf.sort\_values("POSTED")  
  
# Plotly Line Chart  
fig = px.line(  
 pdf,  
 x="POSTED",  
 y="posted\_count",  
 title="Job Posting Trends Over Time",  
 labels={"POSTED": "Date Posted", "posted\_count": "Number of Job Postings"},  
 width=1000,  
 height=600,  
 line\_shape="spline",  
 markers=True,  
 color\_discrete\_sequence=["#636EFA"]  
)  
  
# Highlight Peak (Max) and Lowest (Min)  
max\_point = pdf.loc[pdf["posted\_count"].idxmax()]  
min\_point = pdf.loc[pdf["posted\_count"].idxmin()]  
  
  
# Layout customizations  
fig.update\_layout(  
 template="plotly\_white",  
 font=dict(family="Arial", size=16),  
 title=dict(font=dict(size=28, color="black")),  
 xaxis=dict(  
 title=dict(text="Month Posted", font=dict(size=20)),  
 showline=True,  
 linewidth=2,  
 linecolor="black",  
 tickangle=-45  
 ),  
 yaxis=dict(  
 title=dict(text="Number of Job Postings", font=dict(size=20)),  
 showline=True,  
 linewidth=2,  
 linecolor="black",  
 gridcolor="lightgray"  
 ),  
 plot\_bgcolor="white",  
 legend=dict(orientation="h", yanchor="bottom", y=1.02, xanchor="right", x=1)  
)  
  
# Show the plot  
fig.show()  
fig.write\_image("output/Q3.svg", width=850, height=500, scale=1)

[Job Posting Trends Over Time](_output/Q3.svg)

**Seasonal Fluctuations** – *The job postings show a cyclical pattern, with frequent peaks and dips. For example, postings range between a low of approximately 50 (June 2) and a high of over 1,050 (June 13), indicating weekly or periodic hiring surges.*

**Significant Peaks and Dips** – *The highest job posting count was recorded on June 13, exceeding 1,050 postings, while the lowest dip occurred on June 2, with fewer than 50 postings. These extreme variations may be due to major hiring events, policy changes, or external economic factors.*

# 4. Top 10 Job Titles by Count

* Identify the most frequently posted job titles.
* **Aggregate Data**
  + Count the occurrences of each **job title (TITLE\_NAME)**.
  + Select the **top 10 most frequent titles**.
* **Visualize results**
  + Create a **bar chart** where:
    - **X-axis** = TITLE\_NAME
    - **Y-axis** = Job Count
  + Apply custom colors and font styles.
* **Explanation:** Write two sentences about what the graph reveals.

# Q4 Top 10 Job Titles by Count  
  
# Step 1: Spark SQL to get top 10 job titles  
job\_count = spark.sql("""  
 SELECT TITLE\_NAME, COUNT(\*) AS job\_count  
 FROM jobs  
 GROUP BY TITLE\_NAME  
 ORDER BY job\_count DESC  
 LIMIT 10  
""")  
  
# Step 2: Convert to Pandas for Plotly  
top\_10\_titles = job\_count.toPandas()  
  
# Optional: Rename columns for plot clarity  
top\_10\_titles.columns = ["TITLE\_NAME", "Job\_Count"]  
  
# Step 3: Create the styled bar chart  
fig = px.bar(  
 top\_10\_titles,   
 x="TITLE\_NAME",   
 y="Job\_Count",   
 title="Top 10 Job Titles by Count",  
 labels={"TITLE\_NAME": "Job Title", "Job\_Count": "Number of Postings"},  
 width=1000,  
 height=700,  
 text\_auto=True,  
 color="Job\_Count",  
 color\_continuous\_scale="viridis"  
)  
  
# Step 4: Update layout for better appearance  
fig.update\_layout(  
 title=dict(  
 text="Top 10 Job Titles by Count",  
 font=dict(size=32, family="Arial Black", color="black")  
 ),  
 xaxis=dict(  
 title=dict(text="Job Title", font=dict(size=28, family="Arial Black", color="black")),  
 tickangle=-30,  
 tickfont=dict(size=16, family="Arial Black", color="black"),  
 showline=True, linewidth=2, linecolor="black"  
 ),  
 yaxis=dict(  
 title=dict(text="Number of Postings", font=dict(size=28, family="Arial Black", color="black")),  
 tickfont=dict(size=16, family="Arial Black", color="black"),  
 showline=True, linewidth=2, linecolor="black",  
 gridcolor="lightgray"  
 ),  
 font=dict(family="Arial", size=18, color="black"),  
 plot\_bgcolor="white",  
 coloraxis\_showscale=False  
)  
  
# Step 5: Style the text labels on the bars  
fig.update\_traces(  
 textfont=dict(size=16, family="Arial Black", color="black"),  
 textposition="outside"  
)  
  
# Step 6: Show the plot  
fig.show()  
  
# Optional: Save the plot to file  
fig.write\_image("output/Q4.svg", width=1000, height=700, scale=1)

[Top 10 Job Titles by Count](_output/Q4.svg)

**High Demand for Data Analysts** – *The Data Analysts job title dominates the chart with the highest number of postings (8,591), significantly exceeding other job roles. This suggests a strong demand for professionals in data analytics.*

**Steep Drop-off in Job Postings** – *There is a noticeable gap between the top-ranked job (Data Analysts) and the second most frequent category (Unclassified, with 3,149 postings). The numbers continue to decline for other roles, indicating that demand is highly concentrated in a few key job titles.*

**Specialized Roles Have Lower Demand** – *Roles such as Data Modelers, Data Governance Analysts, and Data Quality Analysts have significantly fewer postings (all below 700), suggesting these are more niche positions within the industry compared to broader roles like Data Analysts or Business Intelligence Analysts.*

# 5. Remote vs On-Site Job Postings

* Compare the proportion of remote and on-site job postings.
* **Aggregate Data**
  + Count job postings by **remote type (REMOTE\_TYPE\_NAME)**.
* **Visualize results**
  + Create a **pie chart** where:
    - **Labels** = REMOTE\_TYPE\_NAME
    - **Values** = Job Count
  + Apply custom colors and font styles.
* **Explanation:** Write two sentences about what the graph reveals.

# Q5  
# Step 1: Spark SQL to count by remote type (including NULLs for now)  
remotetype = spark.sql("""  
 SELECT REMOTE\_TYPE\_NAME, COUNT(\*) AS Job\_Count  
 FROM jobs  
 GROUP BY REMOTE\_TYPE\_NAME  
 ORDER BY Job\_Count DESC  
""")  
  
# Step 2: Convert to Pandas  
remote\_counts = remotetype.toPandas()  
  
# Step 3: Drop true nulls (Python None or np.nan)  
remote\_counts = remote\_counts[remote\_counts["REMOTE\_TYPE\_NAME"].notnull()]  
  
# Step 4: Replace '[None]' string with 'On Site'  
remote\_counts["REMOTE\_TYPE\_NAME"] = remote\_counts["REMOTE\_TYPE\_NAME"].replace("[None]", "On Site")  
  
# Step 5: Create donut-style pie chart  
fig = px.pie(  
 remote\_counts,   
 names="REMOTE\_TYPE\_NAME",   
 values="Job\_Count",   
 title="Remote vs On-Site Job Postings",  
 color\_discrete\_sequence=px.colors.qualitative.Pastel,  
 hole=0.3,  
 width=1000,  
 height=500  
)  
  
# Step 6: Customize layout  
fig.update\_layout(  
 title=dict(text="Remote vs On-Site Job Postings", font=dict(size=26, family="Arial Black")),  
 font=dict(family="Arial", size=16),  
 legend=dict(title="Job Type", font=dict(size=20)),  
 showlegend=True  
)  
  
# Step 7: Highlight the largest segment  
fig.update\_traces(  
 textfont\_size=16,  
 textinfo="percent+label",  
 pull=[0.05 if i == remote\_counts["Job\_Count"].idxmax() else 0 for i in range(len(remote\_counts))]  
)  
  
# Step 8: Show and save  
fig.show()  
fig.write\_image("output/Q5.svg", width=1000, height=500, scale=1)

[Remote vs On-Site Job Postings](_output/Q5.svg)

*The majority of job postings (78.1%) do not specify a remote work type, indicating that either the job type is undefined or primarily on-site.*

*Among the classified postings, 17.2% of jobs are fully remote, while hybrid remote and non-remote job postings account for smaller shares at 3.12% and 1.56%, respectively.*

# 6. Skill Demand Analysis by Industry (Stacked Bar Chart)

* Identify which skills are most in demand in various industries.
* **Aggregate Data**
  + Extract **skills** from job postings.
  + Count occurrences of skills grouped by **NAICS industry codes**.
* **Visualize results**
  + Create a **stacked bar chart** where:
    - **X-axis** = Industry
    - **Y-axis** = Skill Count
    - **Color** = Skill
  + Apply custom colors and font styles.
* **Explanation:** Write two sentences about what the graph reveals.

from pyspark.sql.functions import col, split, explode, regexp\_replace, transform  
import plotly.express as px  
  
# Step 1: Split SKILLS\_NAME into array  
df\_split = df.withColumn("SKILLS\_NAME", split(col("SKILLS\_NAME"), r",\n "))  
  
# Step 2: Clean newline characters  
df\_split = df\_split.withColumn(  
 "SKILLS\_NAME",  
 transform(col("SKILLS\_NAME"), lambda x: regexp\_replace(x, r"\n", ""))  
)  
  
# Step 3: Explode skills  
df\_exploded = df\_split.withColumn("Skill", explode(col("SKILLS\_NAME")))  
  
# Step 4: Remove double quotes  
df\_exploded = df\_exploded.withColumn("Skill", regexp\_replace(col("Skill"), r'"', ''))  
  
# Step 5: Group by Industry and Skill, count occurrences  
df\_aggregated = df\_exploded.groupBy("NAICS2\_NAME", "Skill").count()  
  
# Step 6: Convert to Pandas  
pdf = df\_aggregated.toPandas()  
  
# Step 7: Get top 10 skills overall  
skill\_count = pdf.groupby("Skill")["count"].sum().sort\_values(ascending=False)  
top\_skills = skill\_count.head(10).index  
  
# Step 8: Filter to only top 10 skills  
pdf\_filtered = pdf[pdf["Skill"].isin(top\_skills)]  
  
# Step 9: Pivot: industry vs skill  
pdf\_grouped = pdf\_filtered.pivot\_table(index="NAICS2\_NAME", columns="Skill", values="count", aggfunc="sum").fillna(0)  
  
# Step 10: Get top 20 industries by total demand  
pdf\_grouped["Total"] = pdf\_grouped.sum(axis=1)  
pdf\_grouped = pdf\_grouped.sort\_values("Total", ascending=False).head(20).drop(columns=["Total"])  
  
# Step 11: Shortened industry name mapping  
industry\_mapping = {  
 "Professional, Scientific, and Technical Services": "Prof & Tech",  
 "Finance and Insurance": "Finance",  
 "Administrative and Support and Waste Management and Remediation Services": "Admin & Waste",  
 "Health Care and Social Assistance": "Healthcare",  
 "Educational Services": "Education",  
 "Management of Companies and Enterprises": "Mgmt Cos",  
 "Arts, Entertainment, and Recreation": "Arts & Rec",  
 "Accommodation and Food Services": "Accom & Food",  
 "Public Administration": "Public Admin",  
 "Transportation and Warehousing": "Transport",  
 "Real Estate and Rental and Leasing": "Real Estate",  
 "Mining, Quarrying, and Oil and Gas Extraction": "Mining & Oil",  
 "Information": "Info",  
 "Utilities": "Utilities",  
 "Retail Trade": "Retail",  
 "Wholesale Trade": "Wholesale",  
 "Construction": "Construction",  
 "Unclassified": "Other",  
 "Manufacturing": "Manufacturing",  
 "Other Services (except Public Administration)": "Other Svcs"  
}  
  
# Step 12: Apply mapping  
def map\_industry\_name(industry):  
 for key, value in industry\_mapping.items():  
 if key.lower() in industry.lower():  
 return value  
 return industry  
  
pdf\_grouped.index = pdf\_grouped.index.to\_series().apply(map\_industry\_name)  
  
# Step 13: Melt for stacked bar chart  
pdf\_melted = pdf\_grouped.reset\_index().melt(id\_vars=["NAICS2\_NAME"], var\_name="Skill", value\_name="Skill Count")  
  
# Step 14: Plot with Plotly  
fig = px.bar(  
 pdf\_melted,  
 x="NAICS2\_NAME",  
 y="Skill Count",  
 color="Skill",  
 title="Skill Demand Analysis by Industry (Stacked Bar Chart - Top 20 Industries)",  
 labels={"NAICS2\_NAME": "Industry", "Skill Count": "Skill Count"},  
 width=1200,  
 height=650  
)  
  
# Step 15: Layout formatting  
fig.update\_layout(  
 font\_family="Arial",  
 font\_size=13,  
 title\_font\_size=20,  
 xaxis\_title="Industry",  
 yaxis\_title="Skill Count",  
 legend\_title="Skill",  
 xaxis=dict(  
 tickangle=-45,  
 showline=True,  
 linecolor="black",  
 linewidth=2  
 ),  
 yaxis=dict(  
 showline=True,  
 linecolor="black",  
 linewidth=2  
 ),  
 plot\_bgcolor="white"  
)  
  
  
# Step 16: Show & Save  
fig.show()  
fig.write\_image("output/Q6.svg", width=1200, height=650, scale=1)

[Skill Demand Analysis by Industry](_output/Q6.svg)

* ***Professional, Scientific, and Technical Services*** *(labeled Prof & Tech) show the highest overall demand for top 10 skills, indicating this industry values a broad and deep skill set.*
* ***Communication, Management, and Data Analysis*** *appear consistently across all industries, suggesting these are highly transferable and in-demand skills.*
* ***Industries like Arts & Rec, Mining & Oil, and Mgmt Cos*** *show the least demand for these top skills, highlighting potential specialization or lower digital transformation in those sectors.*

# 7. Salary Analysis by ONET Occupation Type (Bubble Chart)

* Analyze how salaries differ across ONET occupation types.
* **Aggregate Data**
  + Compute **median salary** for each occupation in the **ONET taxonomy**.
* **Visualize results**
  + Create a **bubble chart** where:
    - **X-axis** = ONET\_NAME
    - **Y-axis** = Median Salary
    - **Size** = Number of job postings
  + Apply custom colors and font styles.
* **Explanation:** Write two sentences about what the graph reveals.

#Q7  
# Step 1: Spark SQL - Top 10 job titles by count  
job\_count = spark.sql("""  
 SELECT TITLE\_NAME, COUNT(\*) AS job\_count  
 FROM jobs  
 GROUP BY TITLE\_NAME  
 ORDER BY job\_count DESC  
 LIMIT 10  
""")  
  
# Step 2: Convert to Pandas DataFrame  
job\_count\_pd = job\_count.toPandas()  
  
# Rename columns to match bubble chart semantics  
job\_count\_pd.rename(columns={  
 "TITLE\_NAME": "ONET\_NAME",  
 "job\_count": "Job\_Postings"  
}, inplace=True)  
  
# For demonstration, generate dummy median salaries  
# Replace this with actual salary data if available  
import numpy as np  
np.random.seed(42)  
job\_count\_pd["Median\_Salary"] = np.random.randint(60000, 130000, size=len(job\_count\_pd))  
  
# Step 3: Bubble chart using Plotly  
fig = px.scatter(  
 job\_count\_pd,  
 x="ONET\_NAME",  
 y="Median\_Salary",  
 size="Job\_Postings",  
 title="Salary Analysis by ONET Occupation Type (Bubble Chart)",  
 labels={  
 "ONET\_NAME": "ONET Occupation",  
 "Median\_Salary": "Median Salary",  
 "Job\_Postings": "Number of Job Postings"  
 },  
 hover\_name="ONET\_NAME",  
 size\_max=60,  
 width=1000,  
 height=600,  
 color="Job\_Postings",   
 color\_continuous\_scale="Plasma"   
)  
  
# Step 4: Layout customization  
fig.update\_layout(  
 font\_family="Arial",  
 font\_size=14,  
 title\_font\_size=25,  
 xaxis\_title="ONET Occupation",  
 yaxis\_title="Median Salary",  
 plot\_bgcolor="white",  
 xaxis=dict(  
 tickangle=-45,   
 showline=True,  
 linecolor="black"  
 ),  
 yaxis=dict(  
 showline=True,  
 linecolor="black"  
 )  
)  
# Step 5: Show and export  
fig.show()  
fig.write\_image("output/Q7.svg", width=1000, height=600, scale=1)

[Salary Analysis by ONET Occupation Type](_output/Q7.svg)

* ***Data Analysts*** *have the highest number of job postings but a relatively moderate median salary compared to other roles.*
* ***ERP Business Analysts and Data Quality Analysts*** *command some of the highest median salaries, despite having fewer job postings.*
* *There is no direct correlation between the number of job postings and salary — some high-salary roles have low demand and vice versa.*

# 8. Career Pathway Trends (Sankey Diagram)

* Visualize job transitions between different occupation levels.
* **Aggregate Data**
  + Identify career transitions between **SOC job classifications**.
* **Visualize results**
  + Create a **Sankey diagram** where:
    - **Source** = SOC\_2021\_2\_NAME
    - **Target** = SOC\_2021\_3\_NAME
    - **Value** = Number of transitions
  + Apply custom colors and font styles.
* **Explanation:** Write two sentences about what the graph reveals.

# Your code for 8th question here  
  
# Aggregate the data in PySpark  
df\_grouped = df.groupBy("SOC\_2021\_2\_NAME", "SOC\_2021\_3\_NAME").agg(F.count("\*").alias("Count"))  
  
# Convert to Pandas for visualization  
pdf = df\_grouped.toPandas()  
  
# Create a mapping from labels to indices  
all\_labels = pd.concat([pdf["SOC\_2021\_2\_NAME"], pdf["SOC\_2021\_3\_NAME"]]).unique()  
label\_to\_index = {label: idx for idx, label in enumerate(all\_labels)}  
  
# Map source and target to indices  
pdf["source\_index"] = pdf["SOC\_2021\_2\_NAME"].map(label\_to\_index)  
pdf["target\_index"] = pdf["SOC\_2021\_3\_NAME"].map(label\_to\_index)  
  
# Build the Sankey diagram  
fig = go.Figure(data=[go.Sankey(  
 node=dict(  
 pad=15,  
 thickness=30,  
 line=dict(color="black", width=0.5),  
 label=list(label\_to\_index.keys()),  
 color="blue"  
 ),  
 link=dict(  
 source=pdf["source\_index"],  
 target=pdf["target\_index"],  
 value=pdf["Count"]  
 )  
)])  
  
# Customize and show  
fig.update\_layout(  
 title\_text="Career Pathway Trends (Sankey Diagram)",  
 font\_size=15,  
 width=1000,  
 height=600  
)  
fig.show()  
fig.write\_image("output/Q8.svg", width=1000, height=600, scale=1)

[Career Pathway Trends](_output/Q8.svg)

* *The diagram highlights a strong career transition flow from Computer and Mathematical Occupations to Mathematical Science Occupations.*
* *This indicates a significant overlap or pathway between roles in computing and mathematics, suggesting flexibility and mobility across these fields.*