Test Quarto Julio Vargas

Julio Vargas Garcia

September 17, 2025

1 import data

Importing Dataset using Pyspark. The code shows the schema and first rows of the dataset. Schema is shown below but partial dataframe is muted.

```python
from pyspark.sql import SparkSession
from pyspark.sql.types import StructType
import json

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))

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))

df.createOrReplaceTempView("jobs")


# save schema to pretty JSON
import json
# convert schema to dict first
```

```
schema_dict = df.schema.jsonValue()
# write it nicely formatted (indent=2 gives pretty print)
with open("data/schema_lightcast.json", "w") as f:
    json.dump(schema_dict, f, indent=2)


df.printSchema()


# Count total rows in the DataFrame
row_count = df.count()
print(row_count)
```

```
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)
```

```
5000
```

2 Step 2 - Creating Relational Tables

We will split the main dataframe into four relational tables based on the dataset:

- Job Postings
- Industries (based on NAICS 2022 and SOC 5)
- Companies
- Location (including MSA - Metropolitan Statistical Area)

Each table should have a primary key and the necessary foreign keys to maintain relationships. Here's a table outlining the four relational tables, along with their relevant columns:

2.1 Example

Let's get 5 rows of id, title, and company_name from the job postings table.

```
spark.sql("SELECT ID AS job_id, title_raw FROM jobs LIMIT 5").show(truncate=False)
```

```
+----------------------------------------+--------------------------------------------------
+
|job_id                                  |title_raw
+----------------------------------------+--------------------------------------------------
+
|1f57d95acf4dc67ed2819eb12f049f6a5c11782c|Enterprise Analyst (II-III)
|0cb072af26757b6c4ea9464472a50a443af681ac|Oracle Consultant - Reports (3592)
|85318b12b3331fa490d32ad014379df01855c557|Data Analyst
|1b5c3941e54a1889ef4f8ae55b401a550708a310|Sr. Lead Data Mgmt. Analyst - SAS Product Owner
|cb5ca25f02bdf25c13edfede7931508bfd9e858f|Comisiones de $1000 - $3000 por semana... Comiensa
+----------------------------------------+--------------------------------------------------
+
```

2.2 Locations Table

Lets extract columns from the main dataframe to create a locations table. The columns are as follows:

I also sorted the locations table by MSA in ascending order and then added a location_id as primary key.

`LOCATION_ID (PK)`, `LOCATION`, `CITY_NAME`, `STATE_NAME`, `COUNTY_NAME`, `MSA`, `MSA_NAME`.

```python
from pyspark.sql.functions import col, monotonically_increasing_id

# using inbuilt pyspark select
# locations_df = df.select(
#     col("location"),
#     col("city_name"),
#     col("state_name"),
#     col("county_name"),
#     col("msa"),
#     col("msa_name")
# ).distinct().withColumn("location_id", monotonically_increasing_id())

#alternative using selectExpr
locations_df = df.selectExpr("monotonically_increasing_id() AS LOCATION_ID",
                             "location",
                             "city_name",
                             "state_name",
                             "county_name",
                             "msa",
                             "msa_name")

locations_df.createOrReplaceTempView("locations")
locations_df.show(truncate=False)
```

```
+-----------+-------------------------------------------------+------------------
+-------------+--------------------+-----+-------------------------------------
+
|LOCATION_ID|location                                         |city_name         |state_name
+-----------+-------------------------------------------------+------------------
+-------------+--------------------+-----+-------------------------------------
+
|0          |{\n  "lat": 33.20763,\n  "lon": -92.6662674\n}   |El Dorado, AR     |Arkansas
|1          |{\n  "lat": 44.3106241,\n  "lon": -69.7794897\n} |Augusta, ME       |Maine
Waterville, ME               |
|2          |{\n  "lat": 32.7766642,\n  "lon": -96.7969879\n} |Dallas, TX        |Texas
Fort Worth-Arlington, TX      |
|3          |{\n  "lat": 33.4483771,\n  "lon": -112.0740373\n}|Phoenix, AZ       |Arizona
Mesa-Chandler, AZ            |
|4          |{\n  "lat": 37.6392595,\n  "lon": -120.9970014\n}|Modesto, CA       |California
|5          |{\n  "lat": 0,\n  "lon": 0\n}                    |[Unknown City], AR|Arkansas
|6          |{\n  "lat": 33.4941704,\n  "lon": -111.9260519\n}|Scottsdale, AZ    |Arizona
Mesa-Chandler, AZ            |
|7          |{\n  "lat": 39.7589478,\n  "lon": -84.1916069\n} |Dayton, OH        |Ohio
Kettering, OH                 |
|8          |{\n  "lat": 41.1220409,\n  "lon": -74.5804378\n} |Franklin, NJ      |New Jersey
Newark-Jersey City, NY-NJ-PA|
|9          |{\n  "lat": 40.7501,\n  "lon": -73.997\n}        |New York, NY      |New York
Newark-Jersey City, NY-NJ-PA|
|10         |{\n  "lat": 35.6224561,\n  "lon": -117.6708966\n}|Ridgecrest, CA    |California
|11         |{\n  "lat": 21.3069444,\n  "lon": -157.8583333\n}|Honolulu, HI      |Hawaii
|12         |{\n  "lat": 0,\n  "lon": 0\n}                    |[Unknown City], GA|Georgia
|13         |{\n  "lat": 42.331427,\n  "lon": -83.0457538\n}  |Detroit, MI       |Michigan
Warren-Dearborn, MI          |
|14         |{\n  "lat": 32.2987573,\n  "lon": -90.1848103\n} |Jackson, MS       |Mississippi
|15         |{\n  "lat": 42.3600825,\n  "lon": -71.0588801\n} |Boston, MA        |Massachuset
Cambridge-Newton, MA-NH       |
|16         |{\n  "lat": 0,\n  "lon": 0\n}                    |[Unknown City], AZ|Arizona
|17         |{\n  "lat": 58.3019444,\n  "lon": -134.4197221\n}|Juneau, AK        |Alaska
|18         |{\n  "lat": 33.5185892,\n  "lon": -86.8103567\n} |Birmingham, AL    |Alabama
Hoover, AL                |
|19         |{\n  "lat": 37.7749295,\n  "lon": -122.4194155\n}|San Francisco, CA |California
Oakland-Berkeley, CA   |
+-----------+-------------------------------------------------+------------------
```

```
+-------------+--------------------+-----+-------------------------------------
+
only showing top 20 rows
```

2.3 Industries Table

industries INDUSTRY_ID (Primary Key), NAICS_2022_6, NAICS_2022_6_NAME, SOC_5, SOC_5_NAME, LOT_SPECIALIZED_OCCUPATION_NAME, LOT_OCCUPATION_GROUP

```
industries_df = df.selectExpr("monotonically_increasing_id() AS INDUSTRY_ID",
                              "naics_2022_4",
                              "naics_2022_4_name",
                              "naics_2022_5",
                              "naics_2022_5_name",
                              "naics_2022_6",
                              "naics_2022_6_name",
                              "soc_5 AS SOC_5",
                              "soc_5_name AS SOC_5_NAME",
                              "lot_specialized_occupation_name",
                              "lot_occupation_group")

industries_df.createOrReplaceTempView("industries")

industries_df.show(truncate=False)
```

```
+-----------+------------+------------------------------------------------------------
+------------+------------------------------------------------------------------------
+------------+------------------------------------------------------------------------
+-------+---------------+--------------------------------+--------------------
+
|INDUSTRY_ID|naics_2022_4|naics_2022_4_name                                           |naics_
+-----------+------------+------------------------------------------------------------
+------------+------------------------------------------------------------------------
+------------+------------------------------------------------------------------------
+-------+---------------+--------------------------------+--------------------
+
|0          |4413        |Automotive Parts, Accessories, and Tire Retailers           |44133
2051|Data Scientists|General ERP Analyst / Consultant|2310                |
|1          |5613        |Employment Services                                         |56132
2051|Data Scientists|Oracle Consultant / Analyst     |2310                |
|2          |5242        |Agencies, Brokerages, and Other Insurance Related Activities|52429
2051|Data Scientists|Data Analyst                    |2311                |
|3          |5221        |Depository Credit Intermediation                            |52211
2051|Data Scientists|Data Analyst                    |2311                |
|4          |9999        |Unclassified Industry                                       |99999
2051|Data Scientists|Oracle Consultant / Analyst     |2310                |
|5          |5178        |All Other Telecommunications                                |51781
2051|Data Scientists|Data Analyst                    |2311                |
|6          |3344        |Semiconductor and Other Electronic Component Manufacturing  |33441
2051|Data Scientists|Data Analyst                    |2311                |
|7          |5242        |Agencies, Brokerages, and Other Insurance Related Activities|52429
2051|Data Scientists|Data Analyst                    |2311                |
|8          |9999        |Unclassified Industry                                       |99999
2051|Data Scientists|General ERP Analyst / Consultant|2310                |
|9          |5415        |Computer Systems Design and Related Services                |54151
2051|Data Scientists|Data Analyst                    |2311                |
|10         |4238        |Machinery, Equipment, and Supplies Merchant Wholesalers     |42383
2051|Data Scientists|Data Analyst                    |2311                |
|11         |5613        |Employment Services                                         |56132
2051|Data Scientists|Data Analyst                    |2311                |
|12         |5223        |Activities Related to Credit Intermediation                 |52232
2051|Data Scientists|Data Analyst                    |2311                |
|13         |5416        |Management, Scientific, and Technical Consulting Services   |54161
2051|Data Scientists|General ERP Analyst / Consultant|2310                |
|14         |5239        |Other Financial Investment Activities                       |52394
2051|Data Scientists|Enterprise Architect            |2315                |
|15         |6113        |Colleges, Universities, and Professional Schools            |61131
2051|Data Scientists|Data Analyst                    |2311                |
|16         |5415        |Computer Systems Design and Related Services                |54151
2051|Data Scientists|General ERP Analyst / Consultant|2310                |
|17         |5613        |Employment Services                                         |56132
2051|Data Scientists|Oracle Consultant / Analyst     |2310                |
|18         |6214        |Outpatient Care Centers                                     |62149
2051|Data Scientists|Enterprise Architect            |2315                |
|19         |9999        |Unclassified Industry                                       |99999
2051|Data Scientists|Data Analyst                    |2311                |
+-----------+------------+------------------------------------------------------------
+------------+------------------------------------------------------------------------
+------------+------------------------------------------------------------------------
+-------+---------------+--------------------------------+--------------------
+
only showing top 20 rows
```

2.4 Companies Table

```
#
# Step 1: Create Companies Table (Primary Key: company_id)
companies_df = df.select(
    col("company"),
    col("company_name"),
    col("company_raw"),
    col("company_is_staffing")
).distinct().withColumn("company_id", monotonically_increasing_id())
# companies_df.show(5)
companies = companies_df.toPandas()
companies.drop(columns=["company"], inplace=True)
companies.rename(columns={"company_is_staffing": "is_staffing"},
inplace=True)
companies.to_csv("./output/companies.csv", index=False)
companies.head()
```

	company_name	company_raw	is_staffing	company_id
0	Murphy USA	Murphy USA	False	0
1	Smx Corporation Limited	SMX	True	1
2	Sedgwick	Sedgwick	False	2
3	Wells Fargo	Wells Fargo	False	3
4	Unclassified	LH/GM	False	4

2.5 Job postings Table

```
# Step 4: Create Job Postings Table (Adding Foreign Keys)
job_postings_df = df.select(
    col("id").alias("job_id"),
    col("title_clean"),
    col("body"),
    col("company"),
    col("employment_type_name"),
    col("remote_type_name"),
    col("min_years_experience"),
    col("max_years_experience"),
    col("salary"),
```

```python
    col("salary_from"),
    col("salary_to"),
    col("location"),
    col("naics_2022_6"),
    col("posted"),
    col("expired"),
    col("duration")
)

job_postings_df.show(5)
```

```
+--------------------+--------------------+--------------------+--------
+--------------------+----------------+--------------------+--------------------
+------+-----------+---------+--------------------+------------+--------
+---------+--------+
|              job_id|         title_clean|                body| company|employment_type_nam
+--------------------+--------------------+--------------------+--------
+--------------------+----------------+--------------------+--------------------
+------+-----------+---------+--------------------+------------+--------
+---------+--------+
|1f57d95acf4dc67ed...|enterprise analys...|31-May-2024\n\nEn...|  894731|Full-
time (> 32 h...|          [None]|                   2|                   2|  NULL|       NUL
|0cb072af26757b6c4...|oracle consultant...|Oracle Consultant...|  133098|Full-
time (> 32 h...|          Remote|                   3|                   3|  NULL|       NUL
|85318b12b3331fa49...|        data analyst|Taking care of pe...|39063746|Full-
time (> 32 h...|          [None]|                   5|                NULL|  NULL|       NUL
|1b5c3941e54a1889e...|sr lead data mgmt...|About this role:\...|37615159|Full-
time (> 32 h...|          [None]|                   3|                NULL|  NULL|       NUL
|cb5ca25f02bdf25c1...|comisiones de por...|Comisiones de $10...|       0|Part-
time / full-...|          [None]|                NULL|                NULL| 92500|      3500
+--------------------+--------------------+--------------------+--------
+--------------------+----------------+--------------------+--------------------
+------+-----------+---------+--------------------+------------+--------
+---------+--------+
only showing top 5 rows
```

Adding Foreign keys to job postings table

```python
# Join with Companies Table to get company_id
job_postings_df = job_postings_df.join(companies_df.select("company", "company_id"), on="comp
# Join with Locations Table to get location_id
job_postings_df = job_postings_df.join(locations_df.select("location", "location_id"), job_po
how="left").drop("location")

# Join with Industries Table to get industry_id
job_postings_df = job_postings_df.join(industries_df.select("naics_2022_6", "industry_id"),
 job_postings_df.naics_2022_6 == industries_df.naics_2022_6,
 how="left").drop("naics_2022_6")

# Drop redundant columns
job_postings_df = job_postings_df.drop("company", "lat-long")
job_postings_df.createOrReplaceTempView("job_postings")

# Show final job_postings_df structure
job_postings_df.show(5)
```

```
25/09/21 05:16:26 WARN Column: Constructing trivially true equals predicate, 'location == loc
25/09/21 05:16:26 WARN Column: Constructing trivially true equals predicate, 'naics_2022_6 ==
[Stage 293:>  (0 + 1) / 1][Stage 294:>  (0 + 1) / 1][Stage 295:>  (0 + 0) / 1]

+--------------------+--------------------+--------------------+--------------------
+----------------+--------------------+--------------------+------+-----------
+---------+--------+--------+--------+----------+----------+-----------+
|              job_id|         title_clean|                body|employment_type_name|remote_t
+--------------------+--------------------+--------------------+--------------------
+----------------+--------------------+--------------------+------+-----------
+---------+--------+--------+--------+----------+----------+-----------+
|1f57d95acf4dc67ed...|enterprise analys...|31-May-2024\n\nEn...|Full-time (> 32 h...|
|1f57d95acf4dc67ed...|enterprise analys...|31-May-2024\n\nEn...|Full-time (> 32 h...|
|1f57d95acf4dc67ed...|enterprise analys...|31-May-2024\n\nEn...|Full-time (> 32 h...|
|1f57d95acf4dc67ed...|enterprise analys...|31-May-2024\n\nEn...|Full-time (> 32 h...|
|1f57d95acf4dc67ed...|enterprise analys...|31-May-2024\n\nEn...|Full-time (> 32 h...|
+--------------------+--------------------+--------------------+--------------------
+----------------+--------------------+--------------------+------+-----------
+---------+--------+--------+--------+----------+----------+-----------+
only showing top 5 rows
```

3 Query 1: Industry-Specific Salary Trends Grouped by Job Title

Identify **median salary trends** for job postings in the **Technology industry** (`NAICS_2022_5 = '51821'`), grouped by **specialized occupation**.

- **Filter the dataset**
  - Select job postings where `naics_2022_5 = '51821'` (Technology industry).
  - Ensure salary values are **not NULL and greater than 0**.
- **Join the relevant tables**
  - Use `job_postings` as the base table.
  - Join with `industries` using `industry_id`.
- **Aggregate data**
  - Group results by **industry name (`naics_2022_5_name`)** and **specialized occupation (`specialized_occupation`)**.
  - Compute the **median salary** using `PERCENTILE_APPROX()` for specialized occupation.
- **Order the results**
  - Sort by **median salary** in descending order.
- **Visualize results** using **Plotly**
  - Create a **grouped bar chart** where:
    - **X-axis** = `lot_specialized_occupation_name`
    - **Y-axis** = `median_salary`
    - **Color** = `industry_name`
- Ensure different specialized occupations are distinguishable across the industry.

Query compute median salary by specialized occupation in Tech in progress

```python
from pyspark.sql import functions as F

# Filter for Tech industry
tech = (
    df.where(
        (F.col("NAICS_2022_5") == "51821") &
        F.col("SALARY").isNotNull() &
        (F.col("SALARY") > 0)
    )
    .withColumn("salary_d", F.col("SALARY").cast("double"))
    .groupBy(
        F.col("NAICS_2022_5_NAME"),
        F.col("LOT_SPECIALIZED_OCCUPATION_NAME")   # ← aquí nombre real
    )
    .agg(F.expr("percentile_approx(salary_d, 0.5)").alias("median_salary"))
    .orderBy(F.desc("median_salary"))
)

tech.show(truncate=False)
```

```
+--------------------------------------------------------------------------------------
+---------------------------------+-------------+
|NAICS_2022_5_NAME                                                                     |LOT
+--------------------------------------------------------------------------------------
+---------------------------------+-------------+
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|Ent
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|SAP
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|Ora
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|Gen
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|Dat
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|Bus
|Computing Infrastructure Providers, Data Processing, Web Hosting, and Related Services|Dat
+--------------------------------------------------------------------------------------
+---------------------------------+-------------+
```

4 PLOT

```
import plotly.express as px
import plotly.io as pio

pio.renderers.default = "iframe"

pdf = tech.toPandas().sort_values("median_salary", ascending=True)

fig = px.bar(
    pdf,
    x="median_salary",
    y="LOT_SPECIALIZED_OCCUPATION_NAME",
    color="LOT_SPECIALIZED_OCCUPATION_NAME",  # colores por ocupación
    orientation="h",
    title="Median Salary Trends in the Technology Sector by Specialized Occupation",
    labels={
        "median_salary": "Median Salary ($)",
        "LOT_SPECIALIZED_OCCUPATION_NAME": "Specialized Occupation"
    },
    height=800,
    width=1000
)

fig.update_layout(showlegend=False)  # oculta la leyenda
fig.show()
```

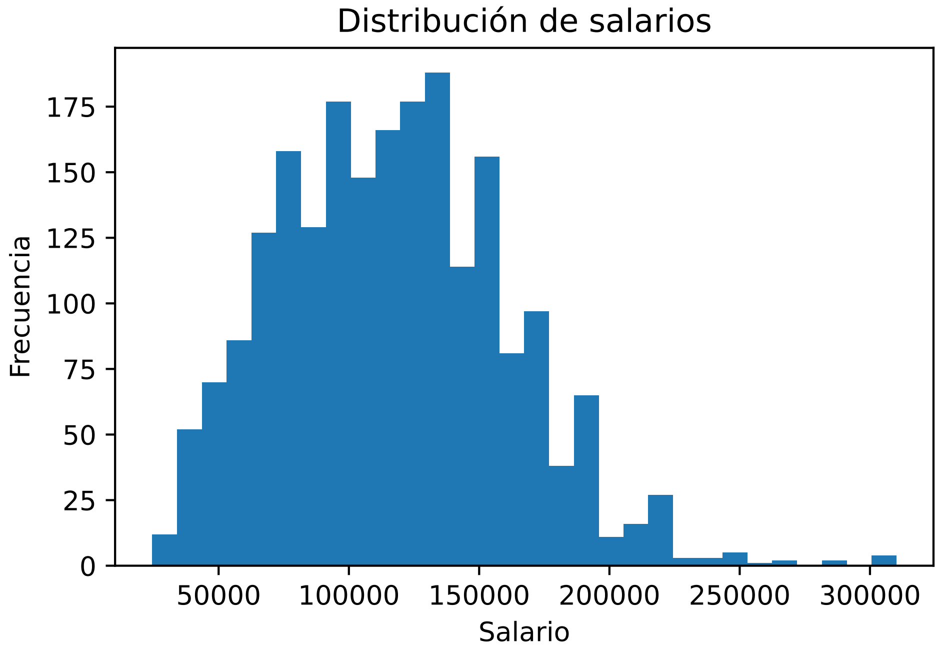
```
Unable to display output for mime type(s): text/html
```

```python
from pyspark.sql import functions as F
import plotly.express as px

# Filtrar y agrupar
tech2 = (df
    .where( (F.col("NAICS_2022_6") == F.lit("518210")) &
            F.col("SALARY").isNotNull() &
            (F.col("SALARY") > 0) )
    .groupBy(
        F.coalesce(F.col("NAICS_2022_6_NAME"), F.lit("NAICS 518210")).alias("industry_name'
        F.col("LOT_SPECIALIZED_OCCUPATION_NAME")
    )
    .agg(F.expr("percentile_approx(SALARY, 0.5)").alias("median_salary"))
    .orderBy(F.desc("median_salary"))
)
```

```python
import matplotlib.pyplot as plt
#from pyspark.sql.functions import col

df_pandas = df.select("SALARY").toPandas()
plt.hist(df_pandas['SALARY'].dropna(), bins=30)
plt.title("Distribución de salarios")
plt.xlabel("Salario")
plt.ylabel("Frecuencia")
plt.show()
```


Distribución de salarios
Frecuencia
Salario
0
25
50
75
100
125
150
175
50000
100000
150000
200000
250000
300000