**File Deliverables**

**Team -08**

Anil Kumar Bekkam

Rahul Boini

Mohan Jayaram Krishna Bojanapalli

Sankalp Bolisetti

Charan Reddy Marri

Mansa Maram

School Of Professional Studies, Saint Louis University

IS-5960-Masters Research Project

Prof. Maria Weber

May 06, 2025

**File Deliverables**

We have used different tools to clean the data and store values. That includes power BI and MySQL workbench.

* **Power BI**: We have used power BI power query to perform the data cleaning operations of the dataset which we have used to develop dashboard. Within power BI we have utilized the power query to clean the data using transform data utility. This section includes the power query code for different operations performed as part of cleaning the dataset.

The following operations are performed using the power query

**Filtered Rows11**

Table.SelectRows(#"Filtered Rows10", each [Revenue] <> null and [Revenue] <> "")

**Use:** Removes rows where the Revenue column is either blank or null.

**Split Column by Delimiter**

Table.SplitColumn(#"Filtered Rows11", "Location", Splitter.SplitTextByDelimiter(",", QuoteStyle.Csv), {"Location.1", "Location.2"})

**Use:** Splits the Location column into Location.1 and Location.2 at the comma.

**Changed Type 1**

Table.TransformColumnTypes(#"Split Column by Delimiter",{{"Location.1", type text}, {"Location.2", type text}})

**Use:** Sets Location.1 and Location.2 columns to text data type.

**Rename Columns**

Table.RenameColumns(#"Changed Type1",{{"Location.1", "City"}, {"Location.2", "state"}})

**Use:** Renames Location.1 → City and Location.2 → state.

**Filtered Rows 12**

Table.SelectRows(#"Renamed Columns", each [City] <> null and [City] <> "")

**Use:** Removes rows with blank or null City values.

**Filtered Rows 13**

Table.SelectRows(#"Filtered Rows12", each [state] <> null and [state] <> "")

**Use:** Removes rows with blank or null state values.

**Added Column 2**

Table.AddColumn(#"Filtered Rows13", "Standardized Job Title", each let

title = [Job Title],

standardTitle =

if Text.Contains(Text.Lower(title), "data scientist") then "Data Scientist"

else if Text.Contains(Text.Lower(title), "machine learning") then "Machine Learning Engineer"

else if Text.Contains(Text.Lower(title), "data engineer") then "Data Engineer"

else if Text.Contains(Text.Lower(title), "data analyst") then "Data Analyst"

else if Text.Contains(Text.Lower(title), "business intelligence") or Text.Contains(Text.Lower(title), " bi ") then "BI Developer"

else if Text.Contains(Text.Lower(title), "ai researcher") or Text.Contains(Text.Lower(title), "artificial intelligence researcher") then "AI Researcher"

else if Text.Contains(Text.Lower(title), "data architect") then "Data Architect"

else if Text.Contains(Text.Lower(title), "statistician") then "Statistician"

else if Text.Contains(Text.Lower(title), "analytics") then "Data Analyst"

else title

in

standardTitle)

**Use:** Adds a new column called Standardized Job Title by mapping variations in job titles to consistent, standard roles based on keywords in the Job Title column. If no match is found, it keeps the original title.

**Added Custom 3**

Table.AddColumn(#"Added Custom2", "common Skills", each if Text.Contains(Text.Lower([Job Description]), "python") then 1 else 0)

**Use:** Flags jobs that mention “python” in the description with a 1; others get 0.

**Added Custom 4**

Table.AddColumn(#"Added Custom3", "common skills.1", each if Text.Contains(Text.Lower([Job Description]), "sql") then 1 else 0)

**Use:** Flags jobs mentioning "SQL" in the description with a 1; others get 0.

**Renamed Columns 1**

Table.RenameColumns(#"Added Custom4",{{"common Skills", "python"}, {"common skills.1", "sql"}})

**Use:** Renames columns to represent the skills they flag python and SQL.

**Added Custom 5**

Table.AddColumn(#"Renamed Columns1", "Machine learning", each if Text.Contains(Text.Lower([Job Description]), "machine learning") or Text.Contains(Text.Lower([Job Description]), "ml") then 1 else 0)

**Use:** Flags jobs mentioning "machine learning" or "ML" in the description with a 1; others get 0.

**Added Custom 6**

Table.AddColumn(#"Added Custom5", "Data visualization", each if Text.Contains(Text.Lower([Job Description]), "tableau") or Text.Contains(Text.Lower([Job Description]), "power bi") or Text.Contains(Text.Lower([Job Description]), "data visualization") then 1 else 0)

**Use:** Flags jobs mentioning "Tableau," "Power BI," or "Data Visualization" in the description with a 1; others get 0.

**Added Custom 7**

Table.AddColumn(#"Added Custom6", "Deep learning", each if Text.Contains(Text.Lower([Job Description]), "deep learning") or Text.Contains(Text.Lower([Job Description]), "neural network") then 1 else 0)

**Use:** Flags jobs mentioning "Deep Learning" or "Neural Network" in the description with a 1; others get 0.

**Added Custom 8**

Table.AddColumn(#"Added Custom7", "Cloud platform", each if Text.Contains(Text.Lower([Job Description]), "aws") or Text.Contains(Text.Lower([Job Description]), "azure") or Text.Contains(Text.Lower([Job Description]), "cloud") or Text.Contains(Text.Lower([Job Description]), "gcp") then 1 else 0)

**Use:** Flags jobs mentioning "AWS," "Azure," "Cloud," or "GCP" in the description with a 1; others get 0.

**Added Custom 9**

Table.AddColumn(#"Added Custom8", "Total Skills", each [Has Python] + [Has SQL] + [Has Machine Learning] + [Has Data Visualization] + [Has Deep Learning] + [Has Cloud Platforms])

**Use:** Calculates the total number of skills (from the specified categories) present for each job description.

**Filtered Rows 14**

Table.SelectRows(#"Added Custom9", each true)

**Use:** Keeps all rows in the table without applying any filtering.

**Removed Columns 1**

Table.RemoveColumns(#"Filtered Rows14",{"Total Skills"})

**Use:** Deletes the Total Skills column from the table.

**Added Column 10**

Table.AddColumn(#"Removed Columns1", "Experience Level", each let

title = Text.Lower([Job Title]),

description = Text.Lower([Job Description]),

// Check for senior positions

isSenior =

Text.Contains(title, "senior") or

Text.Contains(title, "sr.") or

Text.Contains(title, "sr ") or

Text.Contains(title, "lead") or

Text.Contains(title, "principal") or

Text.Contains(title, "staff") or

Text.Contains(title, "architect") or

Text.Contains(title, "manager") or

Text.Contains(title, "director") or

Text.Contains(title, "head of") or

Text.Contains(title, "iii") or

Text.Contains(title, "3") or

Text.Contains(title, "expert"),

// Check for mid-level positions

isMid =

Text.Contains(title, "ii") or

Text.Contains(title, "2") or

Text.Contains(title, "intermediate") or

Text.Contains(title, "mid") or

Text.Contains(title, "associate") or

(not isSenior and Text.Contains(description, "3+ years") or Text.Contains(description, "2+ years")),

// Determine final experience level

expLevel =

if isSenior then "Senior Level"

else if isMid then "Mid-Level"

else "Entry Level"

in

expLevel)

**Use:** Classifies job positions into "Senior Level," "Mid-Level," or "Entry Level" based on job title and description.

**Database operations**

**We have used MySQL Workbench to create the database and store data**

**Creating a Database**

create database DatasciencePathFinder**;**

**Create Proficiency\_level table**

CREATE TABLE proficiency\_level ( proficiency\_id INT AUTO\_INCREMENT PRIMARY KEY, proficiency\_name VARCHAR(255) );

**DML statement to view the table**

SELECT \* FROM DatasciencePathFinder.proficiency\_levels;

**Create Skills table**

CREATE TABLE skills ( skill\_id INT AUTO\_INCREMENT PRIMARY KEY, skill\_name VARCHAR(255), skill\_url VARCHAR(255), proficiency\_id INT, FOREIGN KEY (proficiency\_id) REFERENCES proficiency\_level(proficiency\_id) );

**DML statement to view the table**

SELECT \* FROM DatasciencePathFinder.skills;

**Create user\_jobs\_applied table**

CREATE TABLE user\_jobs\_applied ( user\_job\_applied\_id INT AUTO\_INCREMENT PRIMARY KEY, company VARCHAR(255), created\_at DATETIME(6), is\_applied BIT(1), job\_id\_applied BIGINT, job\_url VARCHAR(255), title VARCHAR(255), proficiency\_id INT, user\_id INT, FOREIGN KEY (proficiency\_id) REFERENCES proficiency\_level(proficiency\_id) -- Optionally, you can add FOREIGN KEY (user\_id) if there is a users table );

**DML statement to view the table**

SELECT \* FROM DatasciencePathFinder.user\_jobs\_applied;

**Create users table**

CREATE TABLE users ( user\_id INT AUTO\_INCREMENT PRIMARY KEY, google\_id VARCHAR(255), user\_email VARCHAR(255), user\_name VARCHAR(255) );

**DML statement to view the table**

SELECT \* FROM DatasciencePathFinder.users;

**Create users\_skills\_progress table**

CREATE TABLE user\_skill\_progress ( user\_skill\_progress\_id INT AUTO\_INCREMENT PRIMARY KEY, is\_completed BIT(1), proficiency\_id INT, skill\_id INT, user\_id INT, FOREIGN KEY (proficiency\_id) REFERENCES proficiency\_level(proficiency\_id), FOREIGN KEY (skill\_id) REFERENCES skills(skill\_id), FOREIGN KEY (user\_id) REFERENCES users(user\_id) );

**DML statement to view the table**

SELECT \* FROM DatasciencePathFinder.user\_skill\_progress;