SQL Data Cleaning Project and EDA – Step-by-Step Process

Step 1: Download Dataset

- Downloaded the Layoffs dataset (CSV) from Kaggle.
- The dataset contained real-world company layoff data, including columns like:
    - company, location, industry, total_laid_off, percentage_laid_off, date, stage, country, funds_raised_millions.

Step 2: Create Database and Import Dataset

- Created a new database in MySQL:
- CREATE DATABASE layoffs_project;
- USE layoffs_project;
- Imported layoffs.csv into MySQL using Table Data Import Wizard in MySQL Workbench.
- Verified data:
- SELECT * FROM layoffs LIMIT 10;

Step 3: Create Staging Table (Safe Copy)

- Created a staging table to avoid modifying raw data:
- CREATE TABLE layoffs_staging LIKE layoffs;
- INSERT INTO layoffs_staging SELECT * FROM layoffs; • Confirmed data count matched the original.

Step 4: Remove Duplicates

- Identified duplicates using CTE and ROW_NUMBER:

```
WITH duplicate_cte AS (
    SELECT *,
```

ROW_NUMBER() OVER (

PARTITION BY company, location, industry, total_laid_off, percentage_laid_off, date, stage, country, funds_raised_millions

) AS row_num

FROM layoffs_staging

)

SELECT * FROM duplicate_cte WHERE row_num > 1;

- Created a second staging table layoffs_staging2 with row_num column.
- Deleted rows where row_num > 1:
- DELETE FROM layoffs_staging2 WHERE row_num > 1;

Step 5: Standardize Data

- Trimmed spaces from text columns:
- UPDATE layoffs_staging2 SET company = TRIM(company);
- UPDATE layoffs_staging2 SET industry = TRIM(industry);
- Standardized industry names:
- UPDATE layoffs_staging2 SET industry = 'Crypto'
- WHERE industry LIKE 'Crypto%';
- Fixed country names:
- UPDATE layoffs_staging2
- SET country = TRIM(TRAILING '.' FROM country)
- WHERE country LIKE 'United States%';

Step 6: Fix Date Column

- Converted date column from text to DATE:
- UPDATE layoffs_staging2

- SET date = STR_TO_DATE(date, '%m/%d/%Y');
- ALTER TABLE layoffs_staging2
- MODIFY COLUMN date DATE;

Step 7: Handle Null and Blank Values

- Converted blanks to NULL:
- UPDATE layoffs_staging SET industry = NULL WHERE industry = '';
- Populated missing industry values using self-join:
- UPDATE layoffs_staging t1
- JOIN layoffs_staging t2
- ON t1.company = t2.company
- SET t1.industry = t2.industry
- WHERE (t1.industry IS NULL OR t1.industry = '')
- AND t2.industry IS NOT NULL;
- Deleted rows with null layoffs and percentage:
- DELETE FROM layoffs_staging2
- WHERE total_laid_off IS NULL AND percentage_laid_off IS NULL;

Step 8: Drop Temporary Columns

- Removed row_num column after cleaning:
- ALTER TABLE layoffs_staging2 DROP COLUMN row_num;

Step 9: Validate Cleaned Data

- Checked for duplicates:
- SELECT company, COUNT(*) FROM layoffs_staging2

- GROUP BY company HAVING COUNT(*) > 1;
- Verified unique industry and country values:
- SELECT DISTINCT industry FROM layoffs_staging2;
- SELECT DISTINCT country FROM layoffs_staging2;

Step 10: Final Clean Table

- Delivered a fully cleaned and optimized table layoffs_staging2:
  No duplicates
  Standardized industry/country names
  Converted date column
  Nulls handled and irrelevant rows removed

Exploratory Data Analysis (EDA)

Objective:

After completing data cleaning in MySQL, I performed Exploratory Data Analysis (EDA) to uncover insights, trends, and patterns in global layoffs using analytical SQL queries. The aim was to leverage the cleaned dataset to generate actionable findings and demonstrate real-world SQL querying proficiency.

Step-by-Step EDA Process

1️⃣ Identify Extremes and Key Metrics

Maximum Layoffs and Percentage Laid Off:

```
SELECT MAX(total_laid_off), MAX(percentage_laid_off)
FROM layoffs_staging2;
```

☑ Highlighted the largest recorded layoffs and identified cases where companies laid off 100% of their workforce.

Companies with Full Workforce Layoffs:

```
SELECT *
FROM layoffs_staging
```

```
WHERE percentage_laid_off = 1
ORDER BY total_laid_off DESC;
```

☑ Pinpointed companies that completely shut down operations.

2️⃣ Aggregations for Company, Country, and Yearly Trends

Top Companies by Total Layoffs:

```
SELECT company, SUM(total_laid_off)
FROM layoffs_staging
GROUP BY company
ORDER BY 2 DESC;
```

☑ Ranked companies based on total layoffs.

Layoffs by Country:

```
SELECT country, SUM(total_laid_off)
FROM layoffs_staging
GROUP BY country
ORDER BY 2 DESC;
```

☑ Showed which regions were most impacted.

Layoffs by Year:

```
SELECT YEAR(date), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY YEAR(date)
ORDER BY 2 DESC;
```

☑ Visualized layoffs year-over-year for macro trend analysis.

3️⃣ Industry and Stage-Level Analysis

By Funding Stage:

```
SELECT stage, SUM(total_laid_off)
FROM layoffs_staging
GROUP BY stage
ORDER BY 2 DESC;
```

☑ Identified which stages (e.g., late-stage startups) had the largest layoffs.

Average Layoff Percentage by Company:

```
SELECT company, AVG(percentage_laid_off)
FROM layoffs_staging
GROUP BY company
ORDER BY 2 DESC;
```

☑ Measured companies' relative workforce impact.

4 Trend Analysis with Rolling Totals

Rolling Monthly Layoffs:

```
WITH rolling_total AS (
  SELECT SUBSTRING(date, 1, 7) AS month, SUM(total_laid_off) AS total_laid
  FROM layoffs_staging2
  WHERE SUBSTRING(date, 1, 7) IS NOT NULL
  GROUP BY month
)
SELECT month, total_laid,
      SUM(total_laid) OVER(ORDER BY month) AS rolling_by_month
FROM rolling_total;
```

☑ Tracked layoffs month-over-month and visualized cumulative effects.

5 Ranking Top Companies by Year

Top 5 Companies Each Year:

```
WITH company_year AS (
  SELECT company, YEAR(date) AS years, SUM(total_laid_off) AS total_laid_off
  FROM layoffs_staging2
  GROUP BY company, YEAR(date)
),
company_year_rank AS (
  SELECT *,
        DENSE_RANK() OVER(PARTITION BY years ORDER BY total_laid_off DESC) AS ranking
  FROM company_year
)
SELECT *
```

```
FROM company_year_rank
WHERE ranking <= 5;
```

☑ Listed the top 5 companies with the most layoffs per year using DENSE_RANK().

Key Insights from EDA

✓ Layoffs were highest in the tech industry and U.S.-based companies.

✓ Late-stage and publicly funded companies experienced larger layoffs compared to early-stage startups.

✓ Several companies experienced 100% workforce reductions, indicating closures.

✓ 2020 and 2023 saw major peaks, correlating with economic downturns and market corrections.

✓ Rolling monthly analysis revealed cumulative layoffs building over time, providing long-term context.

SQL Techniques Demonstrated

• Window Functions: ROW_NUMBER(), DENSE_RANK(), SUM OVER() for ranking and rolling totals.

• Aggregations & Grouping: Company, country, and stage-level metrics.

• Trend Analysis: Yearly, monthly, and cumulative layoffs.

• Analytical Queries: Extreme value identification, ranking, and percentage calculations.

Outcome

This EDA delivered clear, data-driven insights into global layoff patterns and workforce trends, demonstrating strong SQL skills in query optimization, business-focused analysis, and data storytelling. The structured outputs are ready for integration into BI dashboards or further visualization in tools like Power BI/Tableau.

CONCLUSION

This project demonstrates the end-to-end process of preparing and analyzing real-world layoff data using MySQL, covering both data cleaning and exploratory data analysis (EDA). The cleaning phase successfully transformed raw, inconsistent data into an analysis-ready dataset by removing duplicates, standardizing text fields, correcting date formats, handling null values, and optimizing the schema for accuracy and integrity.

Building on this cleaned dataset, the EDA phase uncovered key workforce trends across industries, companies, funding stages, and regions. Insights revealed significant layoffs concentrated in tech and U.S.-based companies, the dominance of late-stage and publicly funded firms in layoff volumes, and clear peaks tied to economic downturns. Using window functions, ranking queries, rolling totals, and aggregations, I highlighted high-impact companies, year-over-year changes, and cumulative layoff trends to provide a clear narrative of the data.

By integrating SQL's advanced querying capabilities with structured analysis, this project showcases the ability to convert raw business data into actionable insights, laying the groundwork for further visualization in BI tools like Power BI or Tableau. This approach mirrors real-world workflows essential for roles in data analysis, business intelligence, and data engineering, emphasizing both technical proficiency and business context.

Ultimately, this project highlights the value of SQL as a powerful tool for data cleaning, transformation, and analytics, turning messy, real-world datasets into clear, decision-ready intelligence for strategic workforce and business planning.