Data Analyst Project. 2025

## DATA CLEANING WITH MYSQL

Presented By: Muhammad Fakhri Azhar



Project.

Data Analyst

### INTRODUCTION

#### BACKGROUND

In today's data-driven era, data quality determines the accuracy of analysis and decision-making. Unfortunately, raw data often contains issues such as missing values, duplication, and inconsistent formatting, which can lead to misleading analysis results. Therefore, this project focuses on the data cleaning process using MySQL to ensure the data used is clean, valid, and ready for analysis. With proper data cleaning, we can minimize the risk of errors and improve the quality of the insights generated.

Data Analyst Project. 2025

## DATA OVERVIEW

This dataset contains information about company layoffs worldwide from the start of the COVID-19 pandemic, March 2020, to March 2023. It records various companies from different industries and countries that conducted layoffs during this period.

This dataset is useful for understanding global layoff trends during the crisis, including the most affected industries, critical layoff periods, and potential correlations with a company's financial status.

#### **Dataset Link:**

https://github.com/mfakhriazhar/data-cleaningsql/blob/main/layoffs\_data.csv

## KEY COLUMN IN DATASET INCLUDE:

- **Company** The name of the company that conducted layoffs
- Location The geographical location of the company
- **Industry** The industry sector the company belongs to
- Total Laid Off The number of employees laid off
- Date The date the layoff occurred
- **Stage** The company's funding stage at the time (e.g., post-IPO, pre-IPO, etc.)
- Percentage Laid Off The percentage of total employees laid off
- Funds Raised (USD) Total funds raised by the company



## DATA ISSUES

Before analysis, the dataset showed several issues: missing values in key columns, duplicate entries, inconsistent formats (like date and company names), and some extreme or unusual values. Cleaning these problems is essential to ensure accurate and trustworthy insights.

#### GOALS:

Remove Duplicates Data

Standardize the Data

Handle Null Values and blank values

1 Remove Any Columns

#### 1. REMOVE DUPLICATE

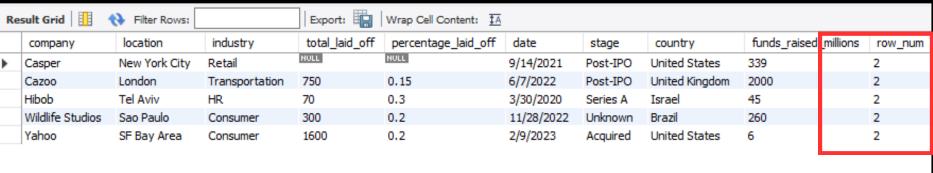
```
- | 🛵 | 🍼 🔍 🗻 🖃
                                     Don't Limit
50 • ⊖ CREATE TABLE `layoffs_staging2` (
51
         `company` text,
52
         `location` text,
53
         `industry` text,
         `total_laid_off` int DEFAULT NULL,
54
         `percentage laid off` text,
55
         `date` text,
57
         `stage` text,
58
         `country` text,
         `funds raised millions` int DEFAULT NULL,
60
         `row num` INT
         ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
61
62
```

```
Don't Limit
                                                  - | 🛵 | 🥩 🔍 🗻 🖃
      INSERT INTO layoffs_staging2
67
       Select *,

→ ROW_NUMBER() OVER(
69
       PARTITION BY company, location, industry, total_laid_off, percentage_laid_off, `date`,
       stage, country, funds raised millions)
71
72
       FROM layoffs_staging;
73
       SELECT *
75
       FROM layoffs_staging2
76
       WHERE row_num > 1;
```

#### RESULT





first, I create a new table by adding a row\_num column to find out which data is a duplicate. Data with row\_num > 1 is a duplicate and data with value 1 is a unique value.

Data Analyst Project. 2025

#### 1. REMOVE DUPLICATE

RESULT



```
Dont Limit

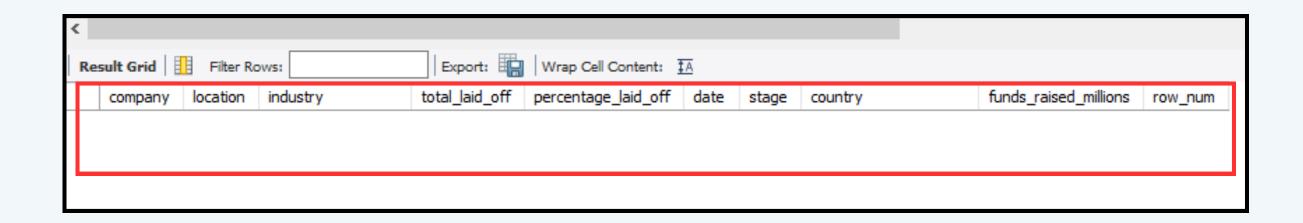
- After checking and rechecking and we are sure that the value is indeed duplicate data,
-- we execute it by deleting it from the table.

DELETE

FROM layoffs_staging2

WHERE row_num > 1;
```

After we make sure that all the values are duplicates, then we just delete them and voila, there is no duplicate data anymore.



Data Analyst

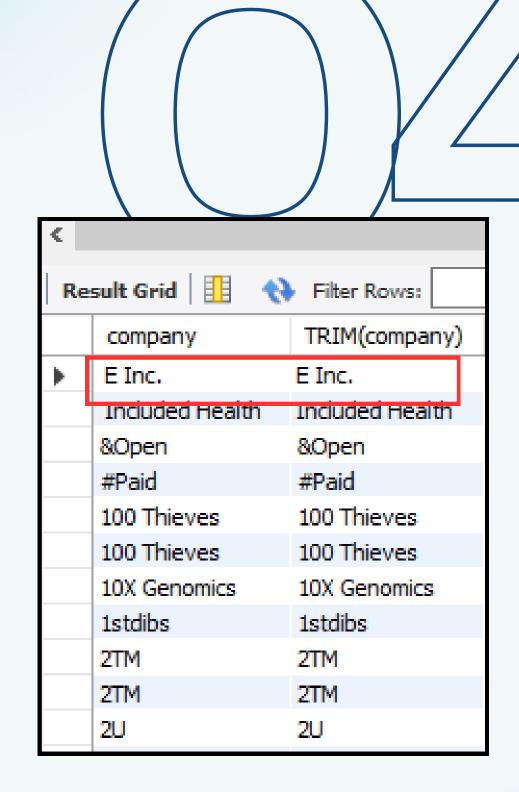
Project.

#### 2. STANDARDIZING DATA

#### RESULT

```
- 🛵 💜 Q 👖 🖃
      | 🐓 🖅 🙊 🔘 | 😘 | 🔘
                                        Don't Limit
       SELECT DISTINCT(company)
90 •
       FROM layoffs_staging2;
91
92
       -- in this company there are some values that contain white space so let's execute it using TRIM.
93
       SELECT company, TRIM(company)
94 •
       FROM layoffs_staging2;
95
96
       -- If it's confirmed to be neat, we just need to update it.
97
       UPDATE layoffs_staging2
98 •
       SET company = TRIM(company);
99
100
```

There are entries in the company column that contain extraneous white space. To address this issue, we use the TRIM function to clean up the data.

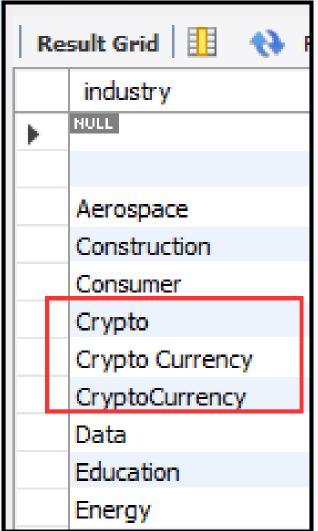


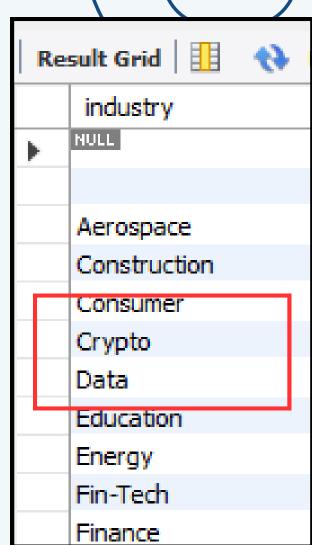
#### 2. STANDARDIZING DATA

```
Don't Limit
111 •
       SELECT *
       FROM layoffs staging2
112
       WHERE industry LIKE 'Crypto%';
113
114
       -- let's just update everything to Crypto only
115
       UPDATE layoffs_staging2
116 •
       SET industry = 'Crypto'
117
       WHERE industry LIKE 'Crypto%';
118
119
```

We've noticed that there are different terms being used, like "Crypto" and "Cryptocurrency," in the industry column. To keep everything clear and consistent, we thought it would be a great idea to update all references to just "Crypto."

#### RESULT





Data Analyst

Project.

#### 2. STANDARDIZING DATA

```
· 🛵 🕩 🔍 🕦 🖃
      SELECT DISTINCT country
       FROM layoffs staging2
132
133
      ORDER BY 1;
134
       -- well we found out that there are United States and United States. let's execute it again
135
      SELECT *
136 •
137
       FROM layoffs staging2
      WHERE country LIKE 'United States%';
138
139
```

```
Don't Limit
                                                      - k
        -- we remove the '.' in the value with trailing
140
        SELECT DISTINCT country, TRIM(TRAILING '.' FROM country)
141 •
        FROM layoffs staging2
142
        ORDER BY 1;
143
144
        -- It's working, let's update the table again
145
        UPDATE layoffs_staging2
146
        SET country = TRIM(TRAILING '.' FROM country)
147
        WHERE country LIKE 'United States%';
148
149
```





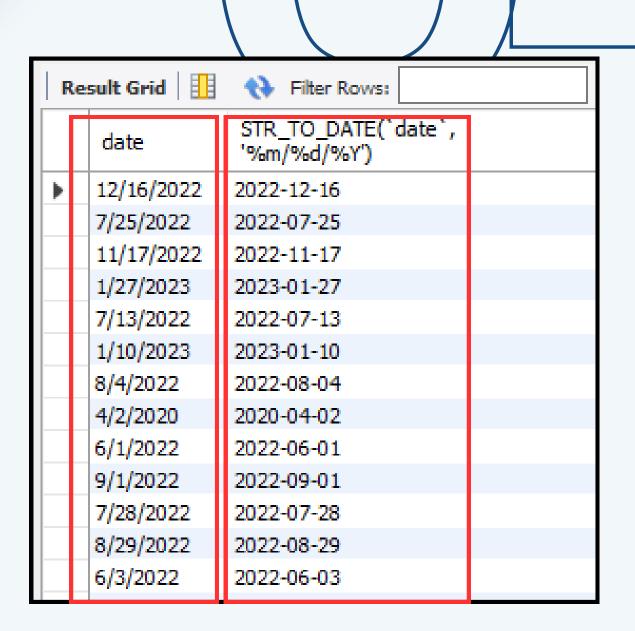
We found two instances of "United States" in the country column. To clean this up, we will use the TRAILING function to remove the period at the end. After that, we can proceed with updating the table accordingly.

#### 2. STANDARDIZING DATA

#### RESULT

```
Q 0
                         90
                                          Don't Limit
        -- we change the format to the standard date format in MySQL
157
        SELECT 'date',
158 •
        STR_TO_DATE('date', '%m/%d/%Y')
159
        FROM layoffs_staging2;
160
161
        -- let's update the table again
162
        UPDATE layoffs staging2
163
        SET 'date' = STR TO DATE('date', '%m/%d/%Y');
164
```

Next, we need to change the data format of the date column to the standard date format in MySQL. This adjustment will greatly facilitate our time series analysis by ensuring consistency in how dates are represented across the dataset. By adhering to the standard format, we can make our queries more effective and our analysis more accurate.



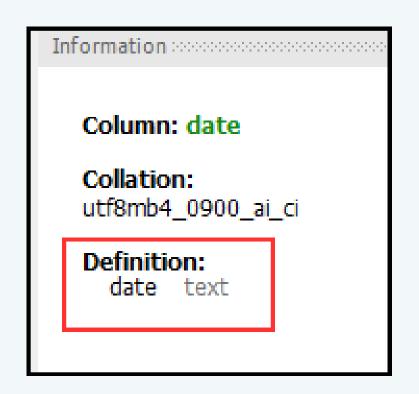
Data Analyst

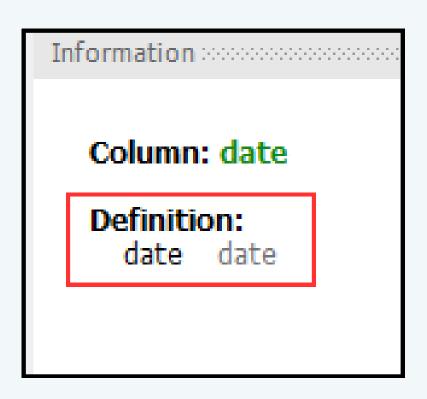
Project.

#### 2. STANDARDIZING DATA



#### **RESULT**





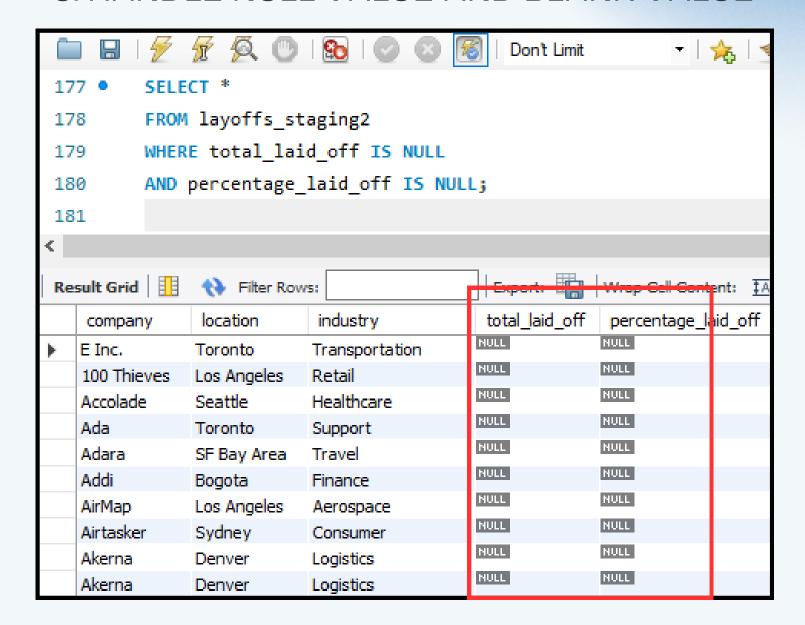


After updating the entries, we proceeded to change the data type of the date column from text to date. This was done using the ALTER TABLE command to ensure that the real raw data and the table remain intact. By doing this, we enhance the efficiency of queries and time series analysis, allowing for better data management and accuracy.

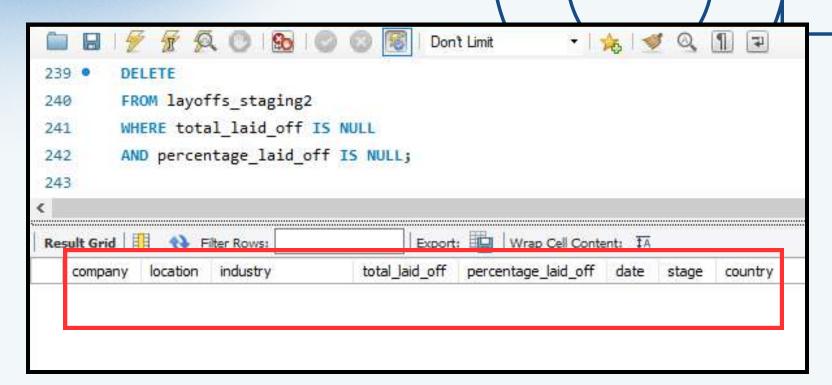
Data Analyst

Project.

#### 3. HANDLE NULL VALUE AND BLANK VALUE



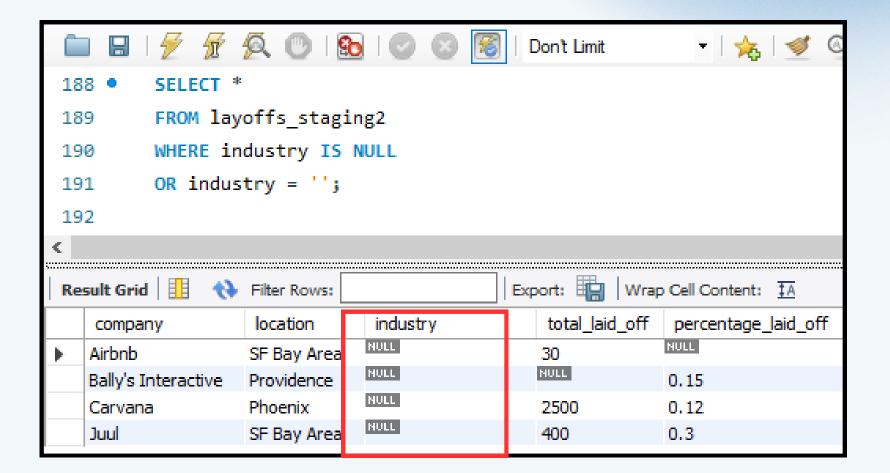
#### RESULT



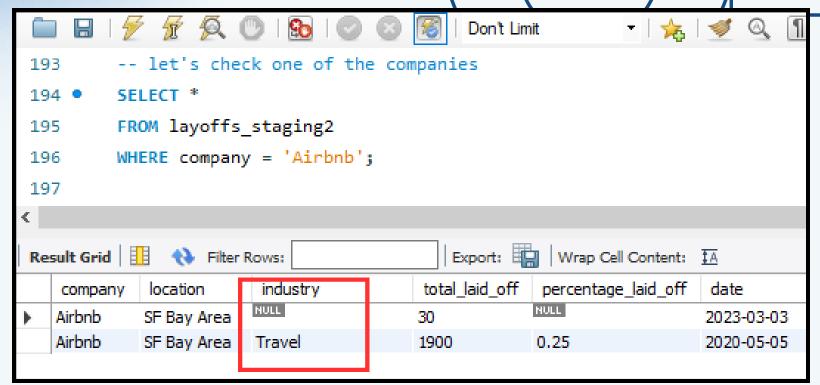
It turns out that there are a lot of NULLs in the columns for total\_laid\_off and percentage\_laid\_off. We need to decide whether to delete these rows or try to populate the data. However, we can't really populate the data effectively, as we don't have a reference column from which to derive this information, such as a total company count or the original data prior to layoffs. Given this situation, it seems more reasonable to delete the NULL entries rather than leave them in the dataset.

Data Analyst Project. 2025

#### 3. HANDLE NULL VALUE AND BLANK VALUE



#### RESULT



It appears there were some blank values in the industry column. After reviewing the data, we found that Airbnb is identified as a travel-related company. To address the blanks, we can fill those entries with the corresponding value of the company itself.

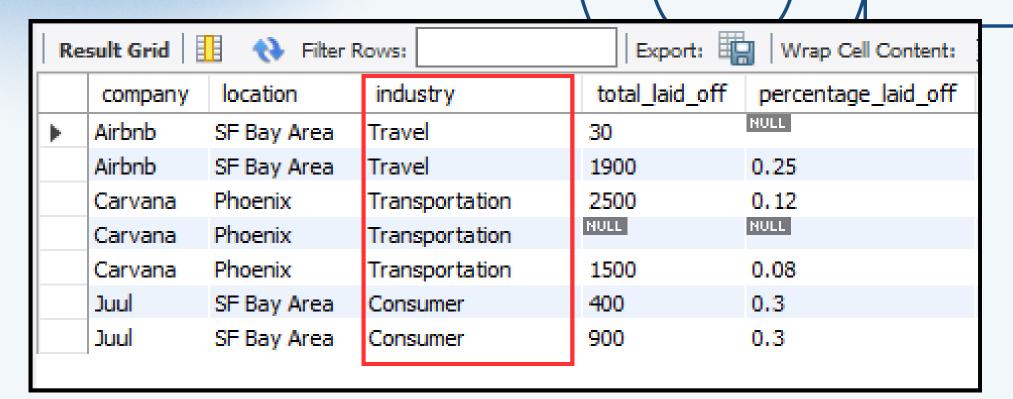
Data Analyst

Project.

#### 3. HANDLE NULL VALUE AND BLANK VALUE

```
🎢 👰 🕛 | 😘 | 🕝 🔞
                                 Don't Limit
UPDATE layoffs staging2 AS t1
JOIN layoffs staging2 AS t2
   ON t1.company = t2.company
SET t1.industry = t2.industry
WHERE t1.industry IS NULL
AND t2.industry IS NOT NULL;
-- done! now we check again
SELECT *
FROM layoffs staging2
WHERE company = 'Airbnb'
OR company = 'Carvana'
OR company = 'Juul';
```

#### RESULT

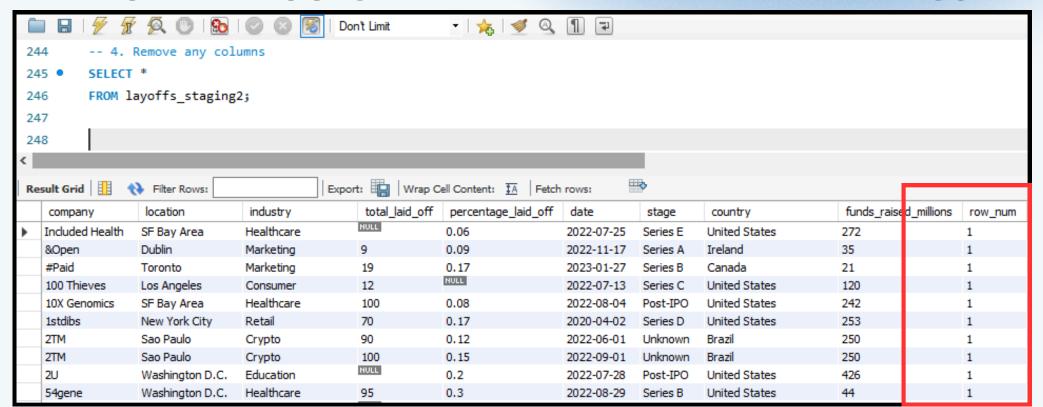


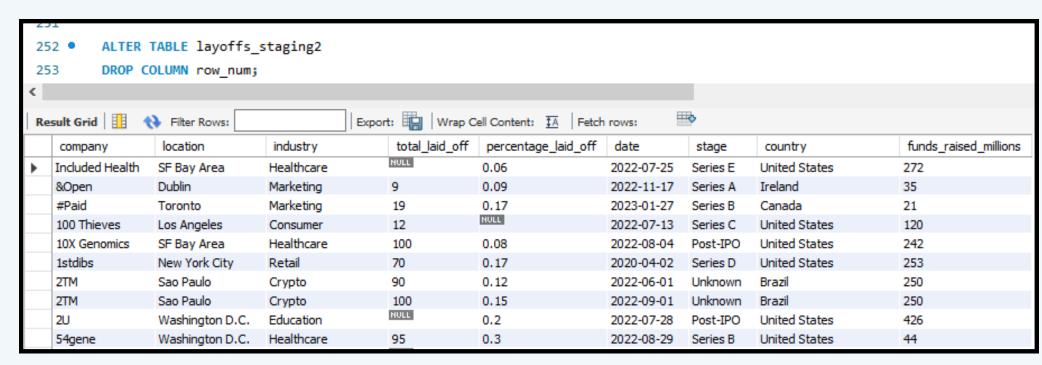
The approach to do this would involve performing a self-join on the table to update the blank values with non-blank ones. This way, we ensure that the industry column accurately reflects the nature of the companies listed. After we apply this solution, the blank values are filled with the industry of the nature of the companies listed. This solution is useful for maintaining consistency and completeness in our dataset.

Data Analyst Project. 2025

#### 4. REMOVE ANY COLUMN

#### RESULT



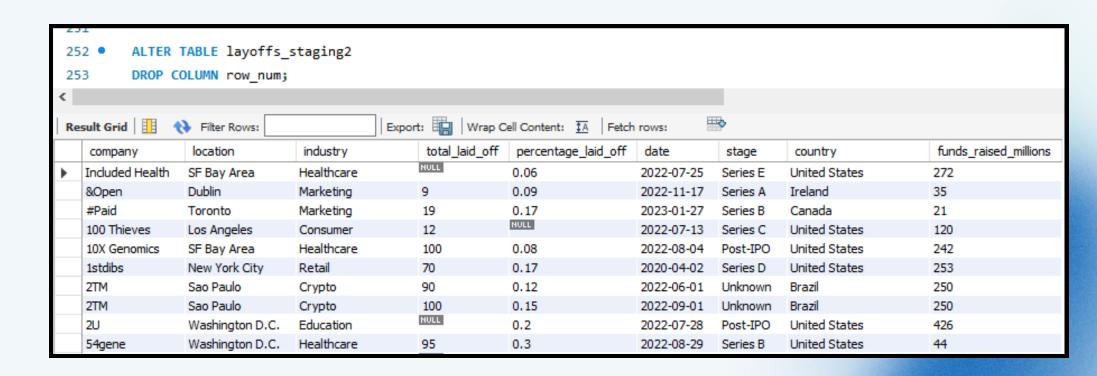




Since the raw\_num column is no longer necessary, we can confidently drop it using the alter table command. Additionally, we have the flexibility to remove other columns as needed to better suit our requirements.

## FINAL RESULTS

Following the data cleaning process in MySQL, the dataset has been successfully refined to eliminate duplicate records, address missing values in critical fields, and correct formatting inconsistencies. We have standardized and cleaned essential columns, including total\_laid\_off, percentage\_laid\_off, and date. This enhanced dataset now serves as a solid and reliable foundation for deeper analysis, enabling us to effectively identify layoff trends across various industries, time periods, and regions.



Data Analyst

Project.



#### CONCLUSION

This project really highlights just how vital data cleaning is for achieving high-quality analysis. By using MySQL, we tackled challenges like missing data, duplicates, and inconsistent formats, which helped ensure our data is reliable and accurate.

With clean data in hand, we can uncover meaningful insights and make thoughtful, informed decisions about global layoff trends. As we move ahead, this dataset is not just ready—it's set for advanced analysis, including visualization and predictive modeling, which will help us better understand and respond strategically.



Data Analyst

Project.

2025

# CLOSING & CONTACT INFO

Thank you for your attention!

This project shows that cleaning the data is not just a step—it's the foundation. With a clean dataset, we're now ready to move forward with confidence in our analysis.

Github Code:

https://github.com/mfakhriazhar/data-

<u>cleaning</u>-

sql/blob/main/Project%20Data%20Cleaning%

20with%20MySQL.sql

Email: mfkriazh57@gmail.com

Phone: 0857-2454-9367

LinkedIn: Muhammad Fakhri Azhar

Portfolio: Click here

GitHub: mfakhriazhar

## THANK YOU

READY TO TELL THE STORY BEHIND THE DATA?

mfkriazh57@gmail.com