Data Analyst Project. 2025

DATA CLEANING

WITH MYSQL

Presented By: Muhammad Fakhri Azhar

01

Data Analyst Project. 2025

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.

02

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-cleaning-sql/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

03

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 :

01 Remove Duplicates Data

02 Standardize the Data

03 Handle Null Values and blank values

04 Remove Any Columns

DATA CLEANING

04

1. REMOVE DUPLICATE

```sql
CREATE TABLE `layoffs_staging2` (
  `company` text,
  `location` text,
  `industry` text,
  `total_laid_off` int DEFAULT NULL,
  `percentage_laid_off` text,
  `date` text,
  `stage` text,
  `country` text,
  `funds_raised_millions` int DEFAULT NULL,
  `row_num` INT
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

```sql
INSERT INTO layoffs_staging2
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 layoffs_staging2
WHERE row_num > 1;
```

RESULT

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country	funds_raised_millions	row_num
Casper	New York City	Retail	NULL	NULL	9/14/2021	Post-IPO	United States	339	2
Cazoo	London	Transportation	750	0.15	6/7/2022	Post-IPO	United Kingdom	2000	2
Hibob	Tel Aviv	HR	70	0.3	3/30/2020	Series A	Israel	45	2
Wildlife Studios	Sao Paulo	Consumer	300	0.2	11/28/2022	Unknown	Brazil	260	2
Yahoo	SF Bay Area	Consumer	1600	0.2	2/9/2023	Acquired	United States	6	2

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 CLEANING

04

1. REMOVE DUPLICATE

```
-- 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;
```

RESULT

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

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country	funds_raised_millions	row_num

DATA CLEANING

2. STANDARDIZING DATA

```sql
SELECT DISTINCT(company)
FROM layoffs_staging2;

-- in this company there are some values that contain white space so let's execute it using TRIM.
SELECT company, TRIM(company)
FROM layoffs_staging2;

-- If it's confirmed to be neat, we just need to update it.
UPDATE layoffs_staging2
SET company = TRIM(company);
```

RESULT

Result Grid | Filter Rows:

company	TRIM(company)
E Inc.	E Inc.
Included Health	Included Health
&Open	&Open
#Paid	#Paid
100 Thieves	100 Thieves
100 Thieves	100 Thieves
10X Genomics	10X Genomics
1stdibs	1stdibs
2TM	2TM
2TM	2TM
2U	2U

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.

DATA CLEANING

2. STANDARDIZING DATA

```sql
SELECT *
FROM layoffs_staging2
WHERE industry LIKE 'Crypto%';

-- let's just update everything to Crypto only
UPDATE layoffs_staging2
SET industry = 'Crypto'
WHERE industry LIKE 'Crypto%';
```

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

Result Grid

industry
NULL
Aerospace
Construction
Consumer
Crypto
Crypto Currency
CryptoCurrency
Data
Education
Energy

Result Grid

industry
NULL
Aerospace
Construction
Consumer
Crypto
Data
Education
Energy
Fin-Tech
Finance

DATA CLEANING

2. STANDARDIZING DATA

```sql
SELECT DISTINCT country
FROM layoffs_staging2
ORDER BY 1;
-- well we found out that there are United States and United States. let's execute it again

SELECT *
FROM layoffs_staging2
WHERE country LIKE 'United States%';
```

```sql
-- we remove the '.' in the value with trailing
SELECT DISTINCT country, TRIM(TRAILING '.' FROM country)
FROM layoffs_staging2
ORDER BY 1;

-- It's working, let's update the table again
UPDATE layoffs_staging2
SET country = TRIM(TRAILING '.' FROM country)
WHERE country LIKE 'United States%';
```

RESULT

Result Grid | Filter Rows: | Export:

country	TRIM(TRAILING '.' FROM country)
South Africa	South Africa
South Korea	South Korea
Spain	Spain
Sweden	Sweden
Switzerland	Switzerland
Thailand	Thailand
Turkey	Turkey
United Arab Emirates	United Arab Emirates
United Kingdom	United Kingdom
United States	United States
United States.	United States
Uruguay	Uruguay
Vietnam	Vietnam

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.

DATA CLEANING

2. STANDARDIZING DATA

```sql
-- we change the format to the standard date format in MySQL
SELECT `date`,
STR_TO_DATE(`date`, '%m/%d/%Y')
FROM layoffs_staging2;

-- let's update the table again
UPDATE layoffs_staging2
SET `date` = STR_TO_DATE(`date`,'%m/%d/%Y');
```

RESULT

Result Grid | Filter Rows:

date	STR_TO_DATE(`date`, '%m/%d/%Y')
12/16/2022	2022-12-16
7/25/2022	2022-07-25
11/17/2022	2022-11-17
1/27/2023	2023-01-27
7/13/2022	2022-07-13
1/10/2023	2023-01-10
8/4/2022	2022-08-04
4/2/2020	2020-04-02
6/1/2022	2022-06-01
9/1/2022	2022-09-01
7/28/2022	2022-07-28
8/29/2022	2022-08-29
6/3/2022	2022-06-03

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 CLEANING

2. STANDARDIZING DATA

```
-- After that, we change the data type from text to date
ALTER TABLE layoffs_staging2
MODIFY COLUMN `date` DATE;
```

RESULT

Information

Column: date

Collation:
utf8mb4_0900_ai_ci

Definition:
date text

Information

Column: date

Definition:
date date

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 CLEANING

3. HANDLE NULL VALUE AND BLANK VALUE

```sql
SELECT *
FROM layoffs_staging2
WHERE total_laid_off IS NULL
AND percentage_laid_off IS NULL;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company	location	industry	total_laid_off	percentage_laid_off
E Inc.	Toronto	Transportation	NULL	NULL
100 Thieves	Los Angeles	Retail	NULL	NULL
Accolade	Seattle	Healthcare	NULL	NULL
Ada	Toronto	Support	NULL	NULL
Adara	SF Bay Area	Travel	NULL	NULL
Addi	Bogota	Finance	NULL	NULL
AirMap	Los Angeles	Aerospace	NULL	NULL
Airtasker	Sydney	Consumer	NULL	NULL
Akerna	Denver	Logistics	NULL	NULL
Akerna	Denver	Logistics	NULL	NULL

RESULT

```sql
DELETE
FROM layoffs_staging2
WHERE total_laid_off IS NULL
AND percentage_laid_off IS NULL;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country

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 CLEANING

3. HANDLE NULL VALUE AND BLANK VALUE

```
SELECT *
FROM layoffs_staging2
WHERE industry IS NULL
OR industry = '';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company	location	industry	total_laid_off	percentage_laid_off
Airbnb	SF Bay Area	NULL	30	NULL
Bally's Interactive	Providence	NULL	NULL	0.15
Carvana	Phoenix	NULL	2500	0.12
Juul	SF Bay Area	NULL	400	0.3

RESULT

```
-- let's check one of the companies
SELECT *
FROM layoffs_staging2
WHERE company = 'Airbnb';
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company	location	industry	total_laid_off	percentage_laid_off	date
Airbnb	SF Bay Area	NULL	30	NULL	2023-03-03
Airbnb	SF Bay Area	Travel	1900	0.25	2020-05-05

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 CLEANING

3. HANDLE NULL VALUE AND BLANK VALUE

```sql
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

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

company	location	industry	total_laid_off	percentage_laid_off
Airbnb	SF Bay Area	Travel	30	NULL
Airbnb	SF Bay Area	Travel	1900	0.25
Carvana	Phoenix	Transportation	2500	0.12
Carvana	Phoenix	Transportation	NULL	NULL
Carvana	Phoenix	Transportation	1500	0.08
Juul	SF Bay Area	Consumer	400	0.3
Juul	SF Bay Area	Consumer	900	0.3

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 CLEANING

04

4. REMOVE ANY COLUMN

RESULT

```
-- 4. Remove any columns
SELECT *
FROM layoffs_staging2;
```

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country	funds_raised_millions	row_num
Included Health	SF Bay Area	Healthcare	NULL	0.06	2022-07-25	Series E	United States	272	1
&Open	Dublin	Marketing	9	0.09	2022-11-17	Series A	Ireland	35	1
#Paid	Toronto	Marketing	19	0.17	2023-01-27	Series B	Canada	21	1
100 Thieves	Los Angeles	Consumer	12	NULL	2022-07-13	Series C	United States	120	1
10X Genomics	SF Bay Area	Healthcare	100	0.08	2022-08-04	Post-IPO	United States	242	1
1stdibs	New York City	Retail	70	0.17	2020-04-02	Series D	United States	253	1
2TM	Sao Paulo	Crypto	90	0.12	2022-06-01	Unknown	Brazil	250	1
2TM	Sao Paulo	Crypto	100	0.15	2022-09-01	Unknown	Brazil	250	1
2U	Washington D.C.	Education	NULL	0.2	2022-07-28	Post-IPO	United States	426	1
54gene	Washington D.C.	Healthcare	95	0.3	2022-08-29	Series B	United States	44	1

```
ALTER TABLE layoffs_staging2
DROP COLUMN row_num;
```

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country	funds_raised_millions
Included Health	SF Bay Area	Healthcare	NULL	0.06	2022-07-25	Series E	United States	272
&Open	Dublin	Marketing	9	0.09	2022-11-17	Series A	Ireland	35
#Paid	Toronto	Marketing	19	0.17	2023-01-27	Series B	Canada	21
100 Thieves	Los Angeles	Consumer	12	NULL	2022-07-13	Series C	United States	120
10X Genomics	SF Bay Area	Healthcare	100	0.08	2022-08-04	Post-IPO	United States	242
1stdibs	New York City	Retail	70	0.17	2020-04-02	Series D	United States	253
2TM	Sao Paulo	Crypto	90	0.12	2022-06-01	Unknown	Brazil	250
2TM	Sao Paulo	Crypto	100	0.15	2022-09-01	Unknown	Brazil	250
2U	Washington D.C.	Education	NULL	0.2	2022-07-28	Post-IPO	United States	426
54gene	Washington D.C.	Healthcare	95	0.3	2022-08-29	Series B	United States	44

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.

```
ALTER TABLE layoffs_staging2
DROP COLUMN row_num;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch rows:

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country	funds_raised_millions
Included Health	SF Bay Area	Healthcare	NULL	0.06	2022-07-25	Series E	United States	272
&Open	Dublin	Marketing	9	0.09	2022-11-17	Series A	Ireland	35
#Paid	Toronto	Marketing	19	0.17	2023-01-27	Series B	Canada	21
100 Thieves	Los Angeles	Consumer	12	NULL	2022-07-13	Series C	United States	120
10X Genomics	SF Bay Area	Healthcare	100	0.08	2022-08-04	Post-IPO	United States	242
1stdibs	New York City	Retail	70	0.17	2020-04-02	Series D	United States	253
2TM	Sao Paulo	Crypto	90	0.12	2022-06-01	Unknown	Brazil	250
2TM	Sao Paulo	Crypto	100	0.15	2022-09-01	Unknown	Brazil	250
2U	Washington D.C.	Education	NULL	0.2	2022-07-28	Post-IPO	United States	426
54gene	Washington D.C.	Healthcare	95	0.3	2022-08-29	Series B	United States	44

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.

06

07

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-cleaning-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