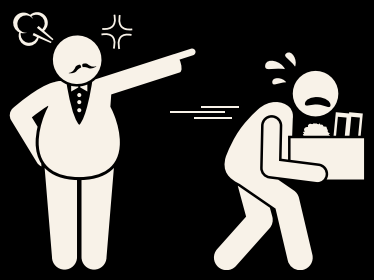
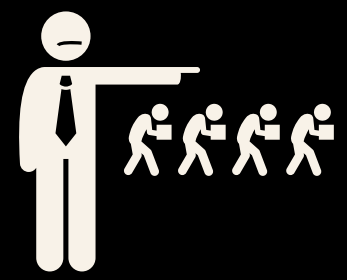
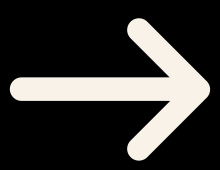


EXPLORING GLOBAL LAYOFFS



DATA CLEANING & ANALYSIS PROJECT



Github :

By : Sakthi K

UNDERSTANDING THE GLOBAL LAYOFFS SCENARIO

- The global economic slowdown has hit several sectors hard, resulting in mass layoffs across numerous companies.
- For instance, more than 191,000 workers at U.S.-based tech companies were laid off in mass job cuts in 2023, according to Crunchbase (2024), and the cuts have since continued growing in 2024.
- Similar downsizing trends have been observed in transportation, healthcare, marketing, finance, recruiting, consumer goods, crypto, food, and other industries.

→ By: sakthi

THE DATASET

- The dataset, sourced from platforms like Bloomberg, San Francisco Business Times, TechCrunch, and The New York Times, includes information on company layoffs.
- Key features include company names, locations, industries, total laid off, percentage laid off, dates, stages, countries, and funds raised.

Dataset is Available →

OR

WWW.KAGGLE.COM

→ By: sakthi

DATA CLEANING STEPS

1. Remove Duplicates
2. Standardizing the data
3. Dealing with Null Values or Blank values
4. Remove Unnecessary Columns

By: sakthi

SAMPLE STRUCTURE

- Here's a basic structure of the layoffs_raw table:
- Database Setup: A database with the layoffs_raw table containing your data.

```
CREATE TABLE `layoffs_raw` (
  `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
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

REMOVE DUPLICATES

- Identifying duplicate rows using window functions

sk*

Limit to 1000 rows

```
49
50    -- Identifying the Duplicate
51    WITH duplicate_cte AS
52    (
53    SELECT *,
54    ROW_NUMBER() OVER(
55    PARTITION BY company, location,
56     industry, total_laid_off, percentage_laid_off, 'date', stage
57     , country, funds_raised_millions) AS row_num
58    FROM layoffs_staging
59    )
60    DELETE
61    FROM duplicate_cte
62    WHERE row_num > 1;
```

result

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
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

→ By: sakthi

STANDARDIZING DATA

- Standardization ensures uniformity in data entries, which is essential for accurate analysis.

> Trimming Company Names

sk*

```
103
104
105 ●   SELECT company, TRIM(company)
106     FROM layoffs_staging2;
107
108 ●   UPDATE layoffs_staging2
109     SET company = TRIM(company);
110
```

MySQL

→ By: sakthi

> Standardizing Industry Names

```
sk*                                                    Limit to 1000 rows
115
116 ●   UPDATE layoffs_staging2
117     SET industry = 'Crypto'
118     WHERE industry LIKE 'Crypto%';
119
120 ●   SELECT DISTINCT location
121     FROM layoffs_staging2;
```

> Verifying the change

```
sk*
121 ●   SELECT DISTINCT location
122     FROM layoffs_staging2
123     ORDER BY 1;
```

MySQL

→ By: sakthi

Formatting Date Fields

```
sk*
126    SELECT DISTINCT date
127    FROM layoffs_staging2
128    ORDER BY 1;
```

```sql
-- Converting date format
UPDATE layoffs_staging2
SET `date` = STR_TO_DATE(`date`, '%m/%d/%Y');

-- Verifying the change
SELECT `date`
FROM layoffs_staging2;

-- Changing date column type to DATE
ALTER TABLE layoffs_staging2
MODIFY COLUMN `date` DATE;
```

Result Grid

	date
▶	12/16/2022
	7/25/2022
	11/17/2022
	1/27/2023
	7/13/2022
	1/10/2023
	8/4/2022
	4/2/2020
	6/1/2022
	9/1/2022
	7/28/2022
	8/29/2022
	6/3/2022
	10/12/2022

layoffs_staging2 6

OUTPUT

Output

Action Output

#	Time	Action	Message
12	18:07:28	SELECT DISTINCT date FROM layoffs_staging2 LIMIT 0, 1000	484 row(s) returned
13	18:10:45	SELECT DISTINCT date FROM layoffs_staging2 ORDER BY 1 LIMIT 0, 1000	484 row(s) returned

MySQL

→ By: sakthi

HANDLING MISSING VALUES

- Handling missing values is crucial to maintain dataset integrity.

> Identifying Missing Values

```
sk*                                                    Limit to 1000 rows
182 •  SELECT *
183    FROM layoffs_staging2
184    WHERE total_laid_off IS NULL
185    AND percentage_laid_off IS NULL;
186
187 •  UPDATE layoffs_staging2
188    SET industry = NULL
189    WHERE industry = "";
190
191 •  SELECT *
192    FROM layoffs_staging2
193    WHERE industry IS NULL OR industry = "";
194
195 •  SELECT *
196    FROM layoffs_staging2
197    WHERE company LIKE "Bally%";
```

result

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

company	location	industry	total_laid_off	percentage_laid_off	date	stage	country
Bally's Interactive	Providence	NULL	NULL	0.15	1/18/2023	Post-IPO	United States
Bally's Interactive	Providence	NULL	NULL	0.15	1/18/2023	Post-IPO	United States
Bally's Interactive	Providence	NULL	NULL	0.15	1/18/2023	Post-IPO	United States
Bally's Interactive	Providence	NULL	NULL	0.15	1/18/2023	Post-IPO	United States

→ By: sakthi

> Standardising null values

sk*

Limit to 1000 rows

```
200
201 ●   SELECT *
202     FROM layoffs_staging2 t1
203     JOIN layoffs_staging2 t2
204         ON t1.company = t2.company
205         AND t1.location = t2.location
206     WHERE (t1.industry IS NULL OR t1.industry = "")
207     AND t2.industry IS NOT NULL;
208
209 ●   UPDATE layoffs_staging2 t1
210     JOIN layoffs_staging2 t2
211         ON t1.company = t2.company
212     SET t1.industry = t2.industry
213     WHERE (t1.industry IS NULL)
214     AND t2.industry IS NOT NULL;
```

OUTPUT

Output

Action Output

#	Time	Action	Message
16	20:47:36	UPDATE layoffs_staging2 t1 JOIN layoffs_staging2 t2 ON t1.company = t2.company SET t1.industry = t2.industry WHERE (t1.industry IS NULL) AND t...	0 row(s) affected Rows matched: 0 Changed: 0 Warnings: 0
17	20:47:54	SELECT * FROM layoffs_staging2 t2 LIMIT 0, 1000	1000 row(s) returned
18	20:48:24	SELECT * FROM layoffs_staging2 LIMIT 0, 1000	1000 row(s) returned
19	23:23:43	SELECT * FROM layoffs_staging2 WHERE company LIKE "Bally%" LIMIT 0, 1000	4 row(s) returned
20	23:27:28	UPDATE layoffs_staging2 t1 JOIN layoffs_staging2 t2 ON t1.company = t2.company SET t1.industry = t2.industry WHERE (t1.industry IS NULL) AND t...	0 row(s) affected Rows matched: 0 Changed: 0 Warnings: 0

MySQL

→ By: sakthi

REMOVE UNNECESSARY COLUMNS

- Lastly, we drop columns that are no longer needed. Removing columns keeps the dataset clean and manageable.

```
sk*
222     -- Dropping the 'row_num' column
223 •   ALTER TABLE layoffs_staging2
224     DROP COLUMN row_num;
225
```

Final Result

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
10X Genomics	SF Bay Area	Healthcare	100	0.08	8/4/2022	Post-IPO	United States	242
1stdibs	New York City	Retail	70	0.17	4/2/2020	Series D	United States	253
1stdibs	New York City	Retail	70	0.17	4/2/2020	Series D	United States	253
2TM	Sao Paulo	Crypto	90	0.12	6/1/2022	Unknown	Brazil	250
2TM	Sao Paulo	Crypto	90	0.12	6/1/2022	Unknown	Brazil	250
2TM	Sao Paulo	Crypto	100	0.15	9/1/2022	Unknown	Brazil	250
2TM	Sao Paulo	Crypto	100	0.15	9/1/2022	Unknown	Brazil	250

→ By: sakthi

THANK YOU FOR YOUR TIME!

This project has provided valuable insights into global layoff data and the importance of clean data in making informed decisions. “Data is compelling and can drive smarter strategies and outcomes”. I'm passionate about continuing to explore new opportunities for data analysis and problem-solving.

If you have any questions or would like to discuss this project further, please feel free to reach out to me!"

Contact:

CHECK MY PROJECT:

→ By: Sakthi K