\-- Data Cleaning

SELECT \\*

FROM world\\_layoffs.layoffs;

\-- 1\. Check for duplicates and Remove Duplicates

\-- 2\. Standardize the Data and fix errors

\-- 3\. Null Values or blank values

\-- 4\. Remove Any Columns and Rows that are not needed for the analysis

CREATE TABLE layoffs\\_stagings

LIKE world\\_layoffs.layoffs;

INSERT layoffs\\_stagings

SELECT \\*

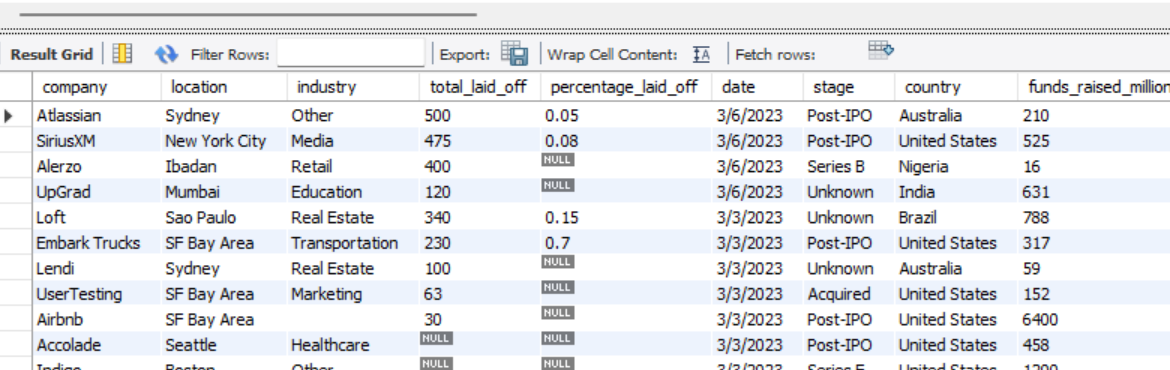
FROM world\\_layoffs.layoffs;

\-- 1\. First let's check for duplicates

SELECT \\*

FROM layoffs\\_stagings;

Output:



\-- Remove Duplicates

SELECT \\*,

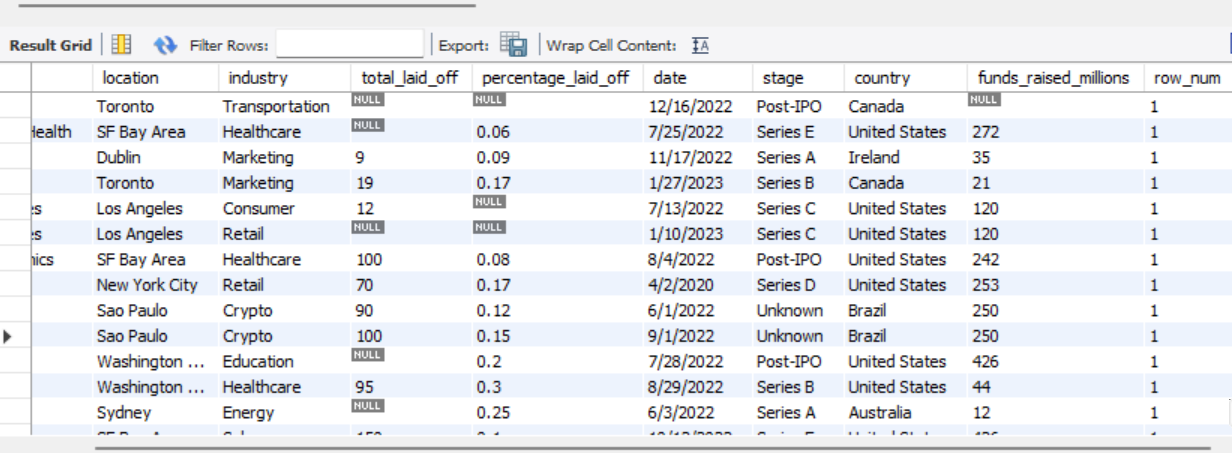
ROW\\_NUMBER() OVER(

PARTITION BY company, location, industry, total\\_laid\\_off, percentage\\_laid\\_off,

'date', stage, country, funds\\_raised\\_millions) AS row\\_num

FROM layoffs\\_stagings;

Output:



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\\_stagings

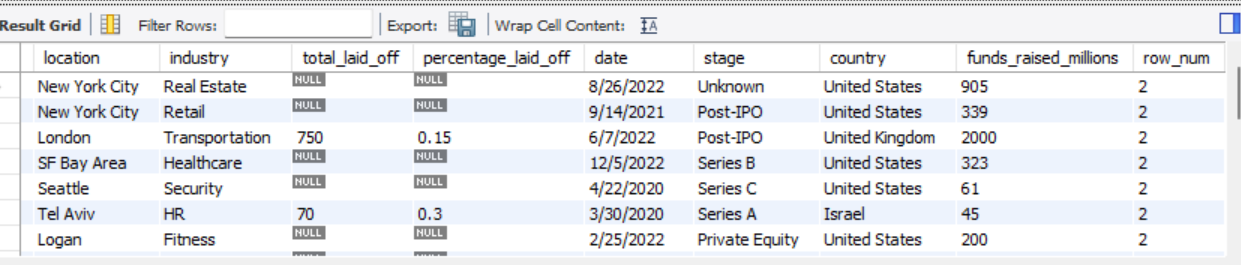
)

SELECT \\*

FROM duplicate\\_cte

WHERE row\\_num \> 1;

Output:

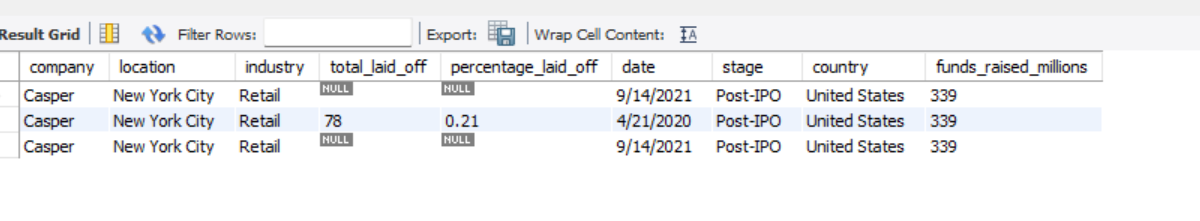


SELECT \\*

FROM layoffs\\_stagings

WHERE company \= 'Casper';

Output:



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\\_stagings

)

DELETE

FROM duplicate\\_cte

WHERE row\\_num \> 1;

CREATE TABLE \`layoffs\\_stagings2\` (

\`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;

SELECT \\*

FROM layoffs\\_stagings2;

INSERT INTO layoffs\\_stagings2

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\\_stagings;

SELECT \\*

FROM layoffs\\_stagings2

WHERE row\\_num \> 1;

DELETE

FROM layoffs\\_stagings2

WHERE row\\_num \> 1;

SELECT \\*

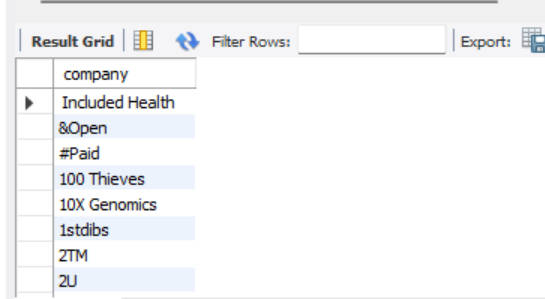
FROM layoffs\\_stagings2;

\-- 2\. Standardize data :- Standardizing data means finding issues with your data and fixing it

SELECT DISTINCT(company)

FROM layoffs\\_stagings2;

Output:



\-- TRIM() is used to remove whitespace

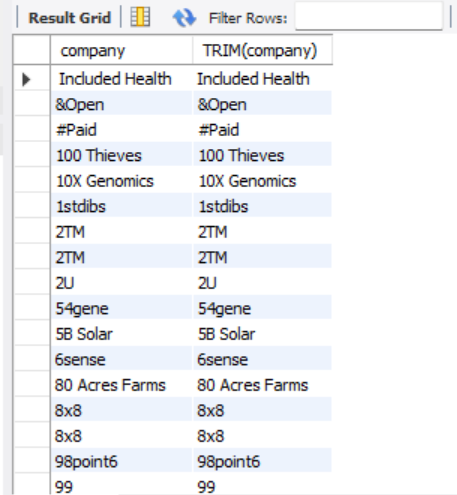
SELECT DISTINCT(TRIM(company))

FROM layoffs\\_stagings2;

SELECT company, TRIM(company)

FROM layoffs\\_stagings2;

Output:



UPDATE layoffs\\_staging2

SET company \= TRIM(company);

\-- If we look at industry it looks like we have null, I also noticed Crypto has multiple different variations. We need to standardize that. Let's set alll to Crypto

SELECT DISTINCT industry

FROM layoffs\\_stagings2

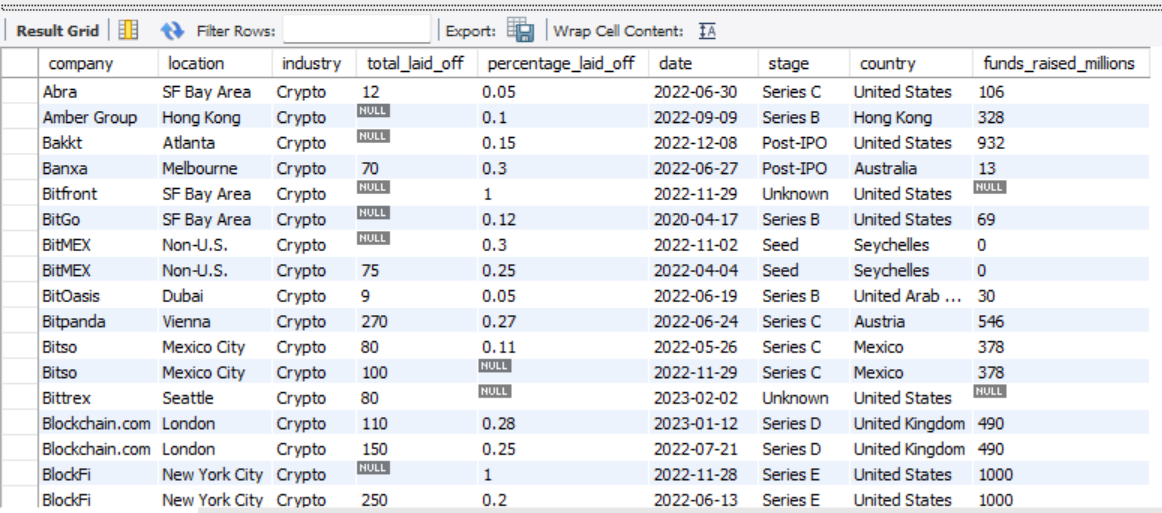
ORDER BY 1;

SELECT \\*

FROM layoffs\\_stagings2

WHERE industry LIKE 'Crypto%';

Output:



UPDATE layoffs\\_stagings2

SET industry \= 'Crypto'

WHERE industry IN ('Crypto Currency', 'CryptoCurrency');

\-- now that's taken care of

SELECT DISTINCT industry

FROM layoffs\\_stagings2

ORDER BY industry;

SELECT DISTINCT country

FROM layoffs\\_stagings2

ORDER BY 1;

\-- We need to look at the country column, Everything looks good except apparently we have some "United States" and some "United States." with a period at the end. Let's standardize it.

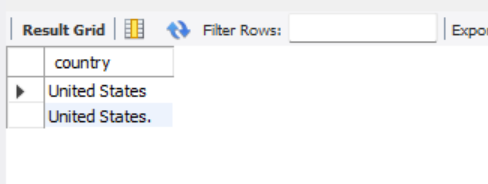
SELECT DISTINCT country

FROM layoffs\\_stagings2

WHERE country LIKE 'United States%'

ORDER BY 1;

Output:



SELECT DISTINCT country, TRIM(TRAILING '.' FROM country)

FROM layoffs\\_stagings2

ORDER BY 1;

UPDATE layoffs\\_satgings2

SET country \= TRIM(TRAILING '.' FROM country)

WHERE country LIKE 'United States%';

SELECT DISTINCT country

FROM layoffs\\_stagings2

ORDER BY country;

\-- Let's also fix the date columns

SELECT \`date\`,

STR\\_TO\\_DATE(\`date\`,'%m/%d/%Y')

FROM layoffs\\_stagings2;

\-- We can use str for date to update this field

UPDATE layoffs\\_stagings2

SET \`date\` \= STR\\_TO\\_DATE(\`date\`, '%m/%d/%Y');

\-- now we can convert the date type properly

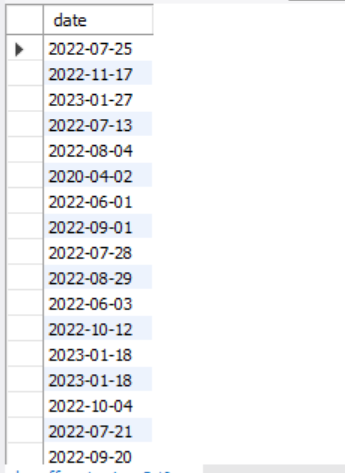
ALTER TABLE layoffs\\_stagings2

MODIFY COLUMN \`date\` DATE;

SELECT \`date\`

FROM layoffs\\_stagings2;

Output:



\-- 3\. Look at Null Values

\-- The null values in total\\_laid\\_off, percentage\\_laid\\_off and funds\\_raised\\_millions all look normal. I dont think I want to change alter that

\-- I like having them null because it makes it easier for calculations during the Exploratory Data Analysis(EDA) phase

\-- There's nothing i want to change with the null values

SELECT \\*

FROM layoffs\\_stagings2

WHERE total\\_laid\\_off IS NULL

AND percentage\\_laid\\_off IS NULL;

\-- We should set the blanks to nulls since those are typically easier to work with

UPDATE layoffs\\_stagings2

SET industry \= NULL

WHERE industry \= '';

\-- Now if we check those are all null

SELECT \\*

FROM layoffs\\_stagings2

WHERE industry IS NULL

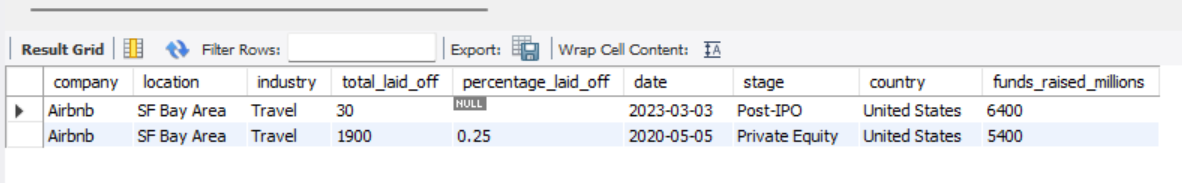
OR industry \= '';

SELECT \\*

FROM layoffs\\_stagings2

WHERE company \= 'Airbnb';

Output:



SELECT t1.industry, t2.industry

FROM layoffs\\_stagings2 t1

JOIN layoffs\\_stagings2 t2

ON t1.company \= t2.company

WHERE (t1.industry IS NULL OR t1.industry \= '')

AND t2.industry IS NOT NULL;

\-- We need to populate those nulls if possible

UPDATE layoffs\\_stagings2 t1

JOIN layoffs\\_stagings2 t2

ON t1.company \= t2.company

SET t1.industry \= t2.industry

WHERE t1.industry IS NULL

AND t2.industry IS NOT NULL;

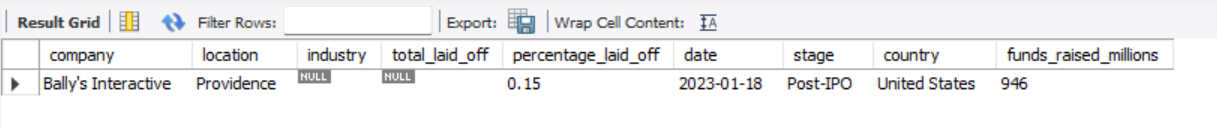
\-- If we check it looks like Bally's was the only one without a populated row to populate this null values

SELECT \\*

FROM layoffs\\_stagings2

WHERE company LIKE 'Bally%';

Output:



\-- 4\. Remove any columns and Rows we need to remove

SELECT \\*

FROM layoffs\\_stagings2

WHERE total\\_laid\\_off IS NULL

AND percentage\\_laid\\_off IS NULL;

\-- Delete Useless data we can't really use

DELETE

FROM layoffs\\_stagings2

WHERE total\\_laid\\_off IS NULL

AND percentage\\_laid\\_off IS NULL;

ALTER TABLE layoffs\\_stagings2

DROP COLUMN row\\_num;

SELECT \\*

FROM layoffs\\_stagings2;

\-- Exploratory Data Analysis

SELECT \\*

FROM layoffs\\_stagings2;

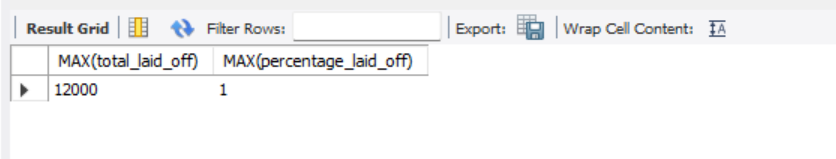
SELECT MAX(total\\_laid\\_off)

FROM layoffs\\_stagings2;

SELECT MAX(total\\_laid\\_off), MAX(percentage\\_laid\\_off)

FROM layoffs\\_stagings2;

Output:



SELECT MAX(percentage\\_laid\\_off), MIN(percentage\\_laid\\_off)

FROM layoffs\\_stagings2;

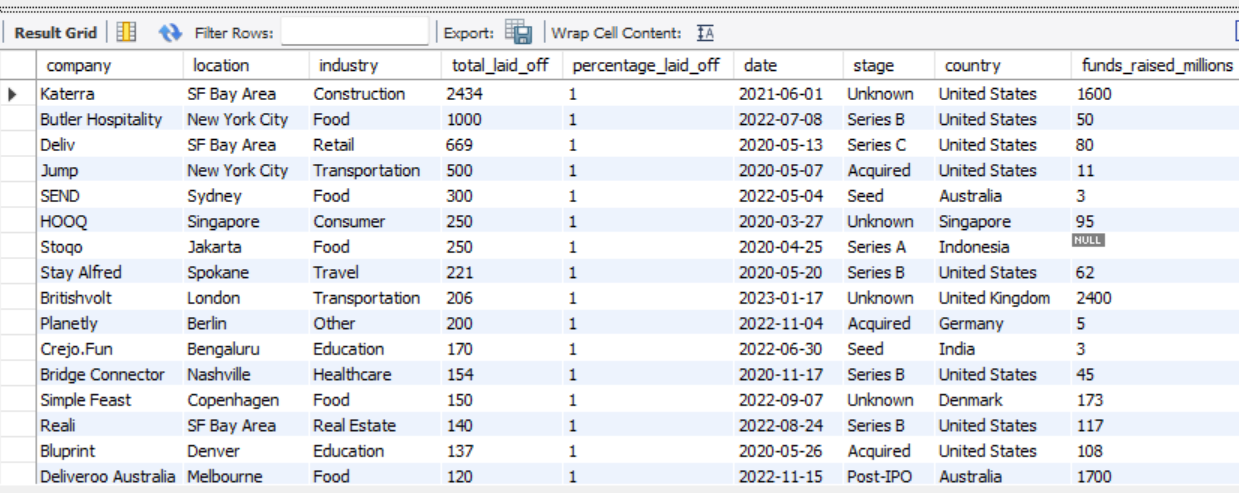
SELECT \\*

FROM layoffs\\_stagings2

WHERE percentage\\_laid\\_off \= 1

ORDER BY total\\_laid\\_off DESC;

Output:



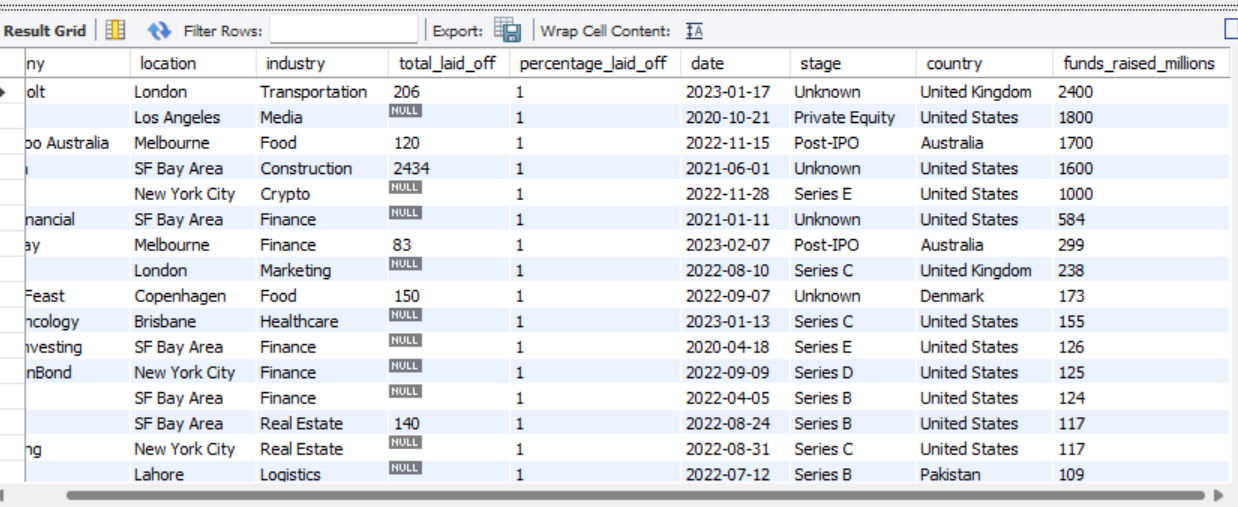
SELECT \\*

FROM layoffs\\_stagings2

WHERE percentage\\_laid\\_off \= 1

ORDER BY funds\\_raised\\_millions DESC;

Output:



\-- Companies with the highest layoff

SELECT company, SUM(total\\_laid\\_off)

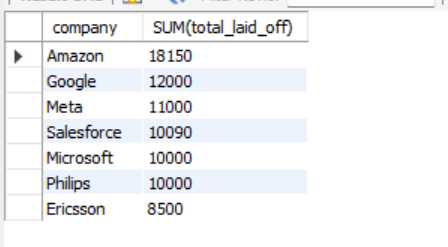
FROM layoffs\\_stagings2

GROUP BY company

ORDER BY 2 DESC

LIMIT 7;

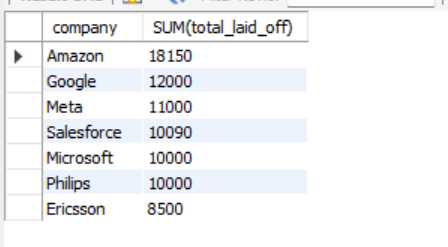
Output:



SELECT MIN(\`date\`), MAX(\`date\`)

FROM layoffs\\_stagings2;

Output:



\-- What is the total laid off of the industries?

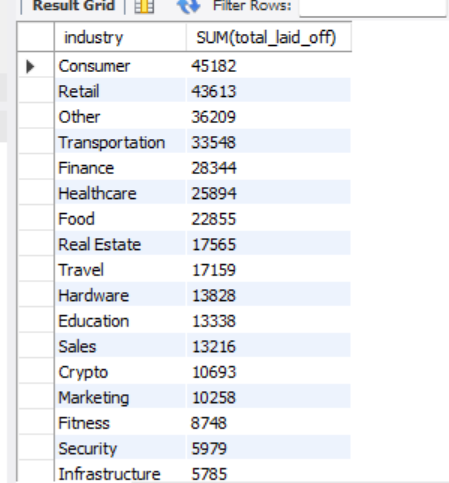
SELECT industry, SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

GROUP BY industry

ORDER BY 2 DESC;

Output:



SELECT industry, ROUND(AVG(percentage\\_laid\\_off),2) AS Average\\_percentage\\_laid\\_off

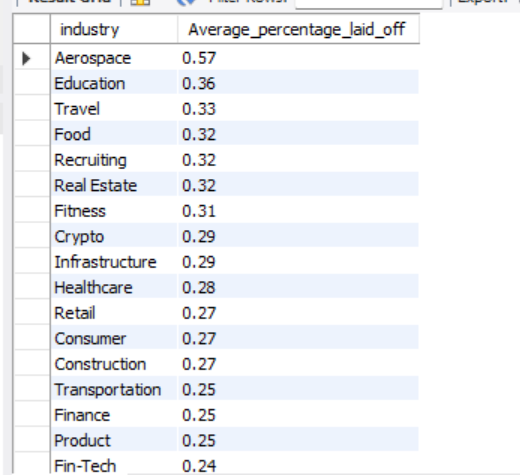
FROM layoffs\\_stagings2

WHERE percentage\\_laid\\_off IS NOT NULL

GROUP BY industry

ORDER BY Average\\_percentage\\_laid\\_off DESC;

Output:



\-- What is the total laid off of each countries?

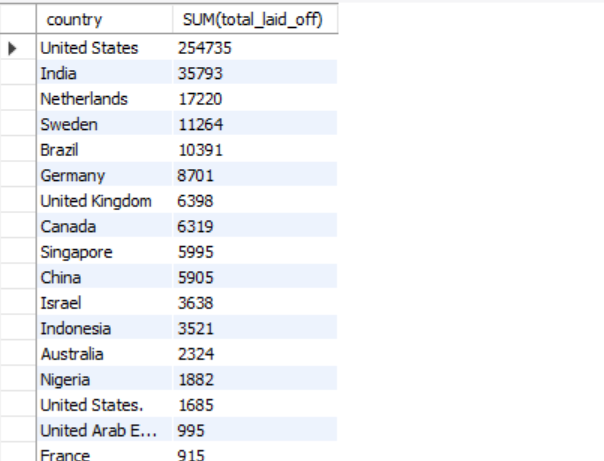
SELECT country, SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

GROUP BY country

ORDER BY 2 DESC;

Output:



SELECT country, ROUND(AVG(percentage\\_laid\\_off),2) AS Average\\_percentage\\_laid\\_off

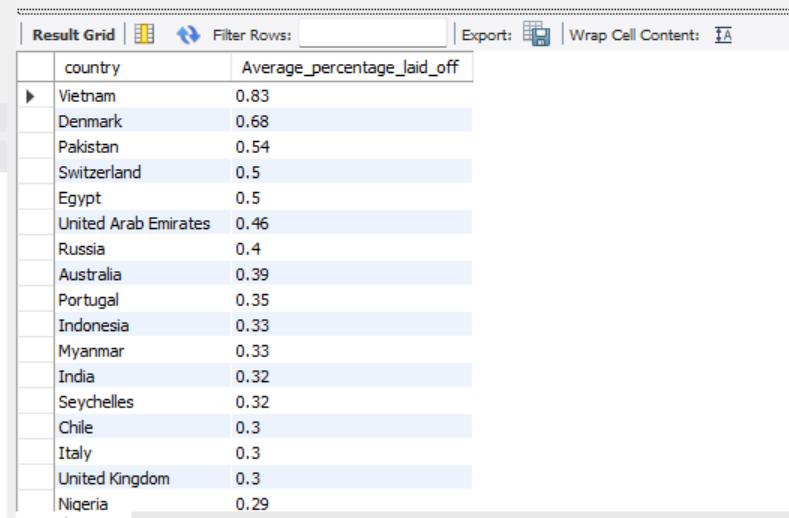
FROM layoffs\\_stagings2

WHERE percentage\\_laid\\_off IS NOT NULL

GROUP BY country

ORDER BY Average\\_percentage\\_laid\\_off DESC;

Output:

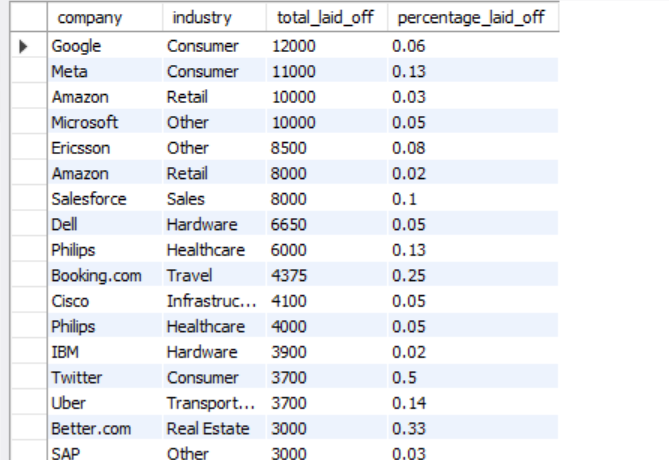


SELECT company, industry, total\\_laid\\_off, percentage\\_laid\\_off

FROM layoffs\\_stagings2

ORDER BY total\\_laid\\_off DESC;

Output:



\-- What is the total laid off for each year?

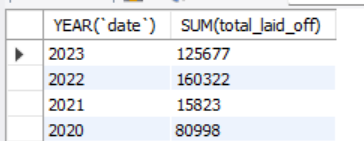
SELECT YEAR(\`date\`), SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

GROUP BY YEAR(\`date\`)

ORDER BY 1 DESC;

Output:



\-- What is the total laid off of each stage?

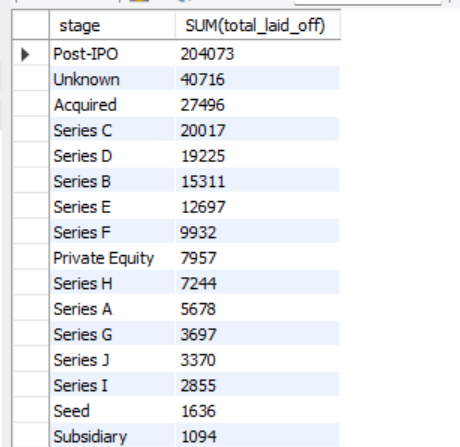
SELECT stage, SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

GROUP BY stage

ORDER BY 2 DESC;

Output:



SELECT stage, SUM(total\\_laid\\_off) AS total\\_laid\\_off

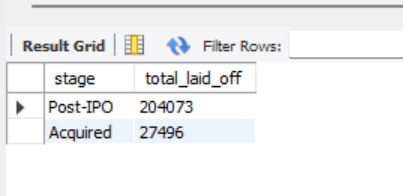
FROM layoffs\\_stagings2

WHERE stage IN ('Acquired', 'Post-IPO')

GROUP BY stage

ORDER BY 2 DESC;

Output:



SELECT SUBSTRING(\`date\`, 1, 7\) AS \`MONTH\`, SUM(total\\_laid\\_off)

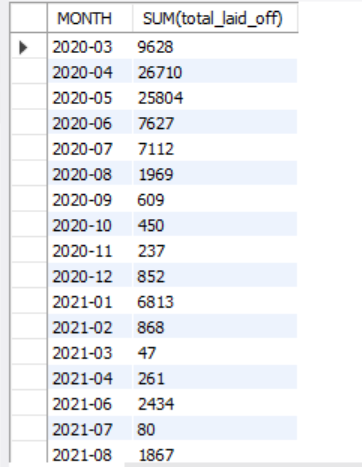
FROM layoffs\\_stagings2

WHERE SUBSTRING(\`date\`, 1, 7\) IS NOT NULL

GROUP BY \`MONTH\`

ORDER BY 1 ASC;

Output:



\-- Rolling total of Layoffs per Month

WITH Rolling\\_Total AS

(

SELECT SUBSTRING(\`date\`, 1, 7\) AS \`MONTH\`, SUM(total\\_laid\\_off) AS Total\\_off

FROM layoffs\\_stagings2

WHERE SUBSTRING(\`date\`, 1, 7\) IS NOT NULL

GROUP BY \`MONTH\`

ORDER BY 1 ASC

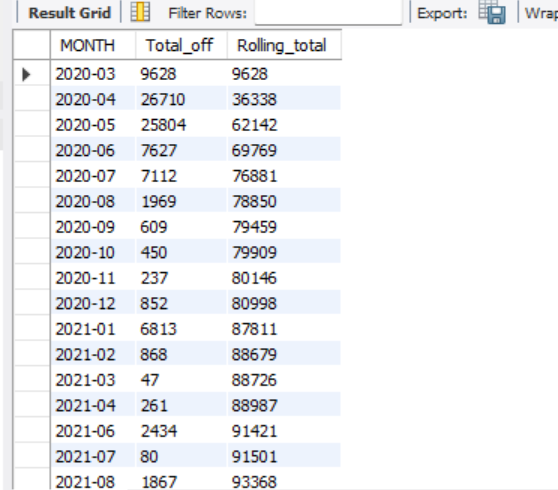
)

SELECT \`MONTH\`, Total\\_off,

SUM(Total\\_off) OVER(ORDER BY \`MONTH\`) AS Rolling\\_total

FROM Rolling\\_Total;

Output:



SELECT company, SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

GROUP BY company

ORDER BY 2 DESC;

Output:



\-- Identify the top 5 companies with the highest percentage of layoffs.

SELECT company, ROUND(SUM(percentage\\_laid\\_off),2)

FROM layoffs\\_stagings2

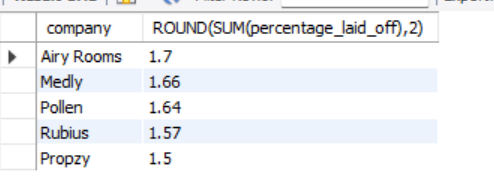
WHERE percentage\\_laid\\_off IS NOT NULL

GROUP BY company

ORDER BY 2 DESC

LIMIT 5;

Output:



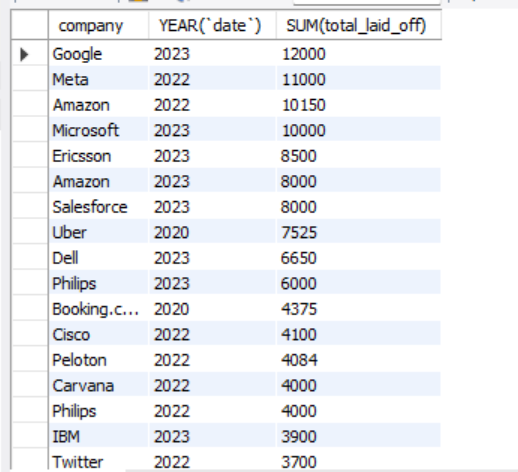
SELECT company, YEAR(\`date\`), SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

GROUP BY company, YEAR(\`date\`)

ORDER BY 3 DESC;

Output:



WITH Company\\_Year (company, years, total\\_laid\\_off) AS

(

SELECT company, YEAR(\`date\`), SUM(total\\_laid\\_off)

FROM layoffs\\_stagings2

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

WHERE years IS NOT NULL

)

SELECT \\*

FROM Company\\_Year\\_Rank

WHERE Ranking \<= 5;

Output:



SELECT company, funds\\_raised\\_millions, SUM(total\\_laid\\_off)

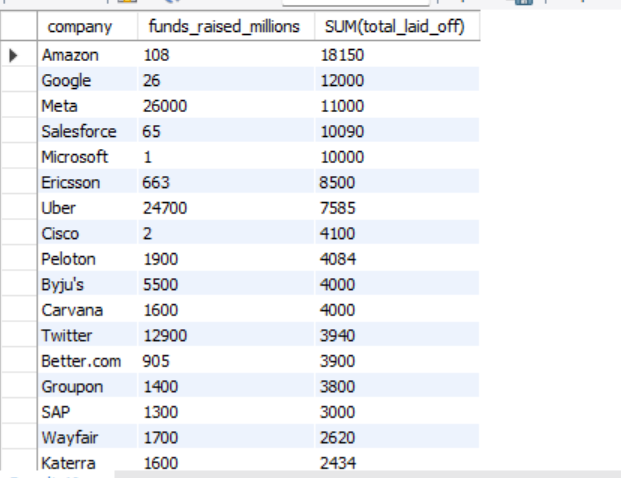
FROM layoffs\\_stagings2

WHERE funds\\_raised\\_millions IS NOT NULL

GROUP BY company, funds\\_raised\\_millions

ORDER BY SUM(total\\_laid\\_off) DESC;

Output:

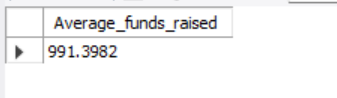


SELECT AVG(funds\\_raised\\_millions) AS Average\\_funds\\_raised

FROM layoffs\\_stagings2

WHERE total\\_laid\\_off IS NOT NULL AND funds\\_raised\\_millions IS NOT NULL;

Output:



WITH significant\\_layoffs AS (

SELECT \\*

FROM layoffs\\_stagings2

WHERE total\\_laid\\_off \> 100 \-- Filter companies with more than 100 layoffs

)

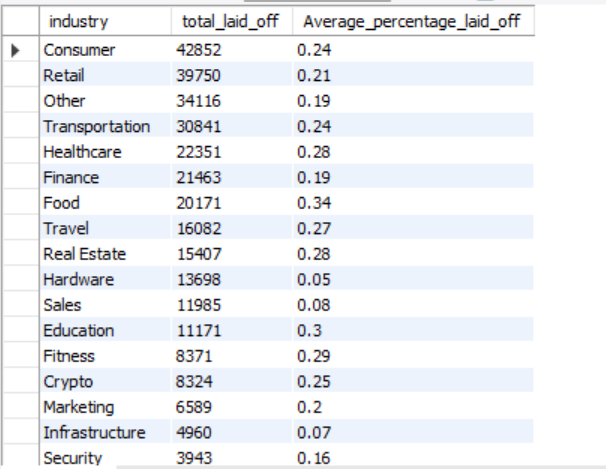
SELECT industry, SUM(total\\_laid\\_off) AS total\\_laid\\_off, ROUND(AVG(percentage\\_laid\\_off),2) AS Average\\_percentage\\_laid\\_off

FROM significant\\_layoffs

GROUP BY industry

ORDER BY total\\_laid\\_off DESC;

Output:



\-- Relationship between the amounts of funds raised and percentage laid off

SELECT funds\\_raised\\_millions, AVG(percentage\\_laid\\_off) AS avg\\_percentage\\_laid\\_off

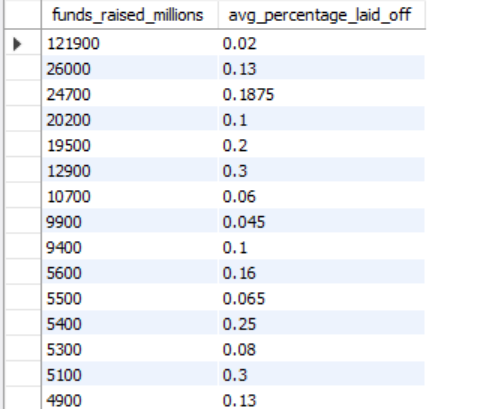
FROM layoffs\\_stagings2

WHERE percentage\\_laid\\_off IS NOT NULL AND funds\\_raised\\_millions IS NOT NULL

GROUP BY funds\\_raised\\_millions

ORDER BY funds\\_raised\\_millions DESC;

Output:



\-- Which sectors are most impacted by layoff

SELECT industry, COUNT(company) AS companies\\_affected, SUM(total\\_laid\\_off) AS total\\_laid\\_off

FROM layoffs\\_stagings2

WHERE total\\_laid\\_off IS NOT NULL

GROUP BY industry

ORDER BY total\\_laid\\_off DESC;

Output:

