```
-- Data Cleaning
 USE word_layoffs;


 SELECT * FROM layoffs;


 -- 1. Remove Duplicates
 -- 2. Standardize the Data
 -- 3. Null Values or Blank Values
 -- 4. Rename any column


 CREATE TABLE layoffs_staging
 LIKE layoffs;


 SELECT * FROM layoffs_staging;


 INSERT  layoffs_staging
 SELECT * FROM layoffs;



  SELECT * FROM layoffs_staging;


  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;


  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;


  SELECT * FROM layoffs_staging
  WHERE company='oda';


  DELETE FROM layoffs_staging
WHERE (company, location, industry, total_laid_off, percentage_laid_off, "date", stage, country,
funds_raised_millions) IN (
    SELECT company, location, industry, total_laid_off, percentage_laid_off, "date", stage, country,
funds_raised_millions
    FROM (
        SELECT company, location, industry, total_laid_off, percentage_laid_off, "date", stage, country,
funds_raised_millions,
               ROW_NUMBER() OVER (
                   PARTITION BY company, location, industry, total_laid_off, percentage_laid_off, "date",
stage, country, funds_raised_millions
                   ORDER BY company -- specify the order for determining duplicates, you can change this as
per your requirement
               ) AS row_num
        FROM layoffs_staging
    ) AS duplicates
    WHERE row_num > 1
);



  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;


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
                ORDER BY company -- specify the order for determining duplicates, you can change this as
per your requirement
            ) AS row_num
       FROM layoffs_staging;


SELECT * FROM layoffs_staging2
WHERE row_num >1 ;



SET SQL_SAFE_UPDATES = 0;


DELETE
FROM layoffs_staging2
WHERE row_num >1;


-- Standardizing data


SELECT DISTINCT(TRIM(company))
FROM layoffs_staging2;


SELECT company, TRIM(company)
From layoffs_staging2;
```

```
UPDATE layoffs_staging2
SET company = TRIM(company);

SELECT DISTINCT industry
FROM layoffs_staging2
ORDER BY 1;

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

UPDATE layoffs_staging2
SET industry = 'Crypto'
WHERE industry LIKE 'Crypto%';

SELECT DISTINCT country
FROM layoffs_staging2
ORDER BY 1;

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

SELECT DISTINCT country, TRIM(TRAILING '.' FROM country)
FROM layoffs_staging2
ORDER BY 1;

UPDATE layoffs_staging2
SET country =TRIM(TRAILING '.' FROM country)
WHERE country LIKE 'United States.';
```

```
UPDATE layoffs_staging2

SET `date` = STR_TO_DATE(`date`, '%m/%d/%Y');



ALTER TABLE layoffs_staging2

MODIFY COLUMN `date` DATE;



SELECT * FROM  layoffs_staging2;



SELECT *

FROM layoffs_staging2

WHERE total_laid_off IS NULL AND percentage_laid_off IS NULL;


SELECT DISTINCT industry

FROM layoffs_staging2

WHERE industry IS NULL

OR industry = '';


SELECT *

FROM layoffs_staging2

WHERE company= 'Airbnb';


UPDATE layoffs_staging2

SET industry = NULL

WHERE industry = '';


SELECT t1.industry, t2.industry FROM layoffs_staging t1

JOIN layoffs_staging t2 on t1.company = t2.company AND t1.location=t2.location
```

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

AND t2.industry IS NOT NULL;


UPDATE layoffs_staging2 t1

JOIN layoffs_staging2 t2

on t1.company = t2.company

SET t1.industry = t2.industry

WHERE t1.industry IS NULL AND t2.industry IS NOT NULL;



SELECT * FROM layoffs_staging2

WHERE company like 'Bally%';


SELECT * FROM layoffs_staging2

WHERE total_laid_off IS NULL AND percentage_laid_off IS NULL;


DELETE FROM layoffs_staging2

WHERE total_laid_off IS NULL AND percentage_laid_off IS NULL;


SELECT * FROM layoffs_staging2;


UPDATE layoffs_staging2

SET total_laid_off = 0

WHERE total_laid_off IS NULL;


UPDATE layoffs_staging2

SET percentage_laid_off = 0

WHERE percentage_laid_off IS NULL;


SELECT AVG(funds_raised_millions) FROM layoffs_staging2;
```

```
UPDATE layoffs_staging2

SET funds_raised_millions = 873.9954

WHERE funds_raised_millions IS NULL;


ALTER TABLE layoffs_staging2

DROP COLUMN row_num;



-- EXPLORATOTY DATA ANALYSIS


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


SELECT * FROM layoffs_staging2

WHERE percentage_laid_off =1

order by funds_raised_millions DESC;


SELECT company, SUM(total_laid_off)

FROM layoffs_staging2

GROUP BY company

ORDER BY 2 DESC;


SELECT MIN(date), max(date)

FROM layoffs_staging2;


SELECT industry, SUM(total_laid_off)

FROM Layoffs_staging2

GROUP BY industry

ORDER BY 2 DESC;


SELECT * FROM layoffs_staging2;
SELECT 'date', SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY 'date'
ORDER BY 1 DESC;

SELECT YEAR(date), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY Year(date)
ORDER BY 1 DESC;

SELECT stage, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY stage
ORDER BY 2 DESC;

SELECT company, AVG(percentage_laid_off)
FROM layoffs_staging2
GROUP BY company
ORDER BY 2 DESC;

SELECT SUBSTRING(date,1,7) as Month, SUM(total_laid_off)
from layoffs_staging2
GROUP BY Month;


SELECT SUBSTRING(date,1,7) as Month, SUM(total_laid_off)
from layoffs_staging2
WHERE SUBSTRING(date,1,7) IS NOT NULL
GROUP BY Month
ORDER BY 1 ASC;
```

```
WITH  Rolling_Total as
(
SELECT SUBSTRING(date,1,7) as Month, SUM(total_laid_off) AS total_off
from layoffs_staging2
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;

SELECT company, SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company
ORDER BY 2 DESC;



SELECT company, YEAR(date), SUM(total_laid_off)
FROM layoffs_staging2
GROUP BY company, YEAR(date)
ORDER BY 3 DESC;

WITH Company_Year (company, years, total_laid_off) 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
WHERE years IS NOT NULL
)
SELECT * FROM company_Year_Rank
WHERE Ranking <= 5;




WITH Company_Year (company, years, total_laid_off) AS
(
   SELECT company, YEAR(date) AS years, SUM(total_laid_off) AS total_laid_off
   FROM layoffs_staging2
   GROUP BY company, YEAR(date)
)
SELECT *, DENSE_RANK() OVER (PARTITION BY years ORDER BY total_laid_off DESC) AS rank
FROM Company_Year;

WITH Company_Year (company, years, total_laid_off) AS (
   SELECT company, YEAR(date) AS years, SUM(total_laid_off) AS total_laid_off
   FROM layoffs_staging2
   GROUP BY company, YEAR(date)
)
SELECT *, DENSE_RANK() OVER (PARTITION BY years ORDER BY total_laid_off DESC) AS `rank`
FROM Company_Year;
```