

```

-- Data Cleaning
-- 1. Removing Duplicates
-- 2. Standardizing Data
-- 3. Null Values or Blank Values
-- 4. Removing Any Columns or Rows

-- 1. Removing Duplicates

-- Creating Copy of the Table
SELECT *
FROM layoffs;

CREATE TABLE layoffs_staging
LIKE layoffs;

SELECT *
FROM layoffs_staging;

INSERT layoffs_staging
SELECT *
FROM layoffs;

-- Identifying 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_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;

-- Creating the Second Table to Remove Duplicates
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;

SELECT *
FROM layoffs_staging2;

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;

-- Deleting Duplicates
SELECT *
FROM layoffs_staging2
WHERE row_num > 1;

DELETE FROM layoffs_staging2
WHERE row_num > 1;

-- 2. Standardizing Data

-- Checking `company` Column
SELECT DISTINCT industry
FROM layoffs_staging2;

-- Removing White Spaces from `company` Column
SELECT company, TRIM(company)
FROM layoffs_staging2;

UPDATE layoffs_staging2
SET company = TRIM(company);

-- Checking `industry` Column
SELECT DISTINCT industry
FROM layoffs_staging2;

-- Correcting Mistakes in `industry` Column
UPDATE layoffs_staging2
SET industry = 'Crypto'
WHERE industry LIKE 'Crypto%';

-- Checking `location` Column
SELECT DISTINCT location
FROM layoffs_staging2
ORDER BY 1;

-- Checking `country` Column
SELECT DISTINCT country
FROM layoffs_staging2
ORDER BY 1;

-- Removing Dots at the End of Country Names
UPDATE layoffs_staging2
SET country = TRIM(TRAILING '.' FROM country)
WHERE country LIKE 'United States%';

-- Checking `date` Column
SELECT `date`
FROM layoffs_staging2;

-- Formatting `date` Column into Date Format
UPDATE layoffs_staging2
SET `date` = STR_TO_DATE(`date`, '%m/%d/%Y');

ALTER TABLE layoffs_staging2
MODIFY COLUMN `date` DATE;

```

-- 3. Null Values or Blank Values

```
SELECT *  
FROM layoffs_staging2;
```

-- Inspecting 'industry' Column

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

```
SELECT *  
FROM layoffs_staging2  
WHERE company = 'Airbnb';
```

```
SELECT t1.industry, t2.industry  
FROM layoffs_staging2 t1  
JOIN layoffs_staging2 t2  
  ON t1.company = t2.company  
WHERE (t1.industry IS NULL OR t1.industry = '')  
AND t2.industry IS NOT NULL;
```

```
UPDATE layoffs_staging2  
SET industry = NULL  
WHERE industry = '';
```

```
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;
```

-- 4. Removing Any Columns or Rows

-- Deleting Rows with NULL Values from `total_laid_off` and `percentage_laid_off` Columns with NULL Values as We Cannot Trust these Data

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

-- Other Data We Cannot Populate

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

-- Deleting `row_num` Column as It is No Longer Needed

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

-- Observing the Final Table

```
SELECT *  
FROM layoffs_staging2;
```