## Layoffs Data Cleaning - SQL Pipeline Documentation

### Source Table

* `` This is the **original dataset** imported from a CSV file containing information about layoffs, including company names, industries, locations, dates, and number of employees laid off.

### Step 1: Create Backup Table (layoff\_staging)

CREATE TABLE IF NOT EXISTS world\_layoffs.layoff\_staging LIKE world\_layoffs.layoffs;  
INSERT INTO world\_layoffs.layoff\_staging SELECT \* FROM world\_layoffs.layoffs;

* A duplicate table is created to preserve the original data for reference.
* layoff\_staging acts as a working backup.

### Step 2: Create Working Table (layoffs\_staging2) for Cleaning

CREATE TABLE IF NOT EXISTS world\_layoffs.layoffs\_staging2 (..., checks INT);

* layoffs\_staging2 includes an extra column checks to hold ROW\_NUMBER() values, used for detecting duplicates.

### Step 3: Insert Data with Row Numbers for Duplicate Detection

INSERT INTO layoffs\_staging2  
SELECT \*, ROW\_NUMBER() OVER (...) AS checks FROM layoff\_staging;

* Duplicates are detected using ROW\_NUMBER() on rows with the same values across key columns.
* Rows with checks > 1 are considered duplicates.

### Step 4: Remove Duplicates

DELETE FROM layoffs\_staging2 WHERE checks > 1;

* Keeps only the first occurrence of each duplicate record.
* Removes redundant rows.

### Step 5: Standardize Text Data

UPDATE layoffs\_staging2 SET company = TRIM(company);  
UPDATE layoffs\_staging2 SET industry = 'Crypto' WHERE industry LIKE 'Crypto%';

* Removes leading/trailing spaces from company names.
* Standardizes variations like “Crypto”, “Cryptocurrency”, etc.

### Step 6: Clean Country/Location Fields

UPDATE layoffs\_staging2 SET country = 'United States' WHERE country LIKE 'United States%';

* Normalizes country names (e.g., “United States of America” to “United States”).

### Step 7: Convert Date Column to Date Type

UPDATE layoffs\_staging2 SET `date` = STR\_TO\_DATE(`date`, '%m/%d/%Y');  
ALTER TABLE layoffs\_staging2 MODIFY COLUMN `date` DATE;

* Converts date from text to DATE datatype for proper sorting and filtering.

### Step 8: Handle Null and Blank Fields

UPDATE layoffs\_staging2 SET industry = NULL WHERE industry = ' ';  
UPDATE s1 SET s1.industry = s2.industry  
FROM layoffs\_staging2 s1  
JOIN layoffs\_staging2 s2 ON s1.company = s2.company  
WHERE s1.industry IS NULL AND s2.industry IS NOT NULL;

* Replaces blank industries with NULL.
* Uses self-join to fill missing industry values based on company match.

### Step 9: Remove Rows with No Layoff Data

DELETE FROM layoffs\_staging2 WHERE total\_laid\_off IS NULL AND percentage\_laid\_off IS NULL;

* Filters out rows with no layoff data at all.

### Step 10: Final Cleanup

ALTER TABLE layoffs\_staging2 DROP COLUMN checks;

* Removes the temporary checks column used for duplicate detection.

### Final Result

* Cleaned data is now in layoffs\_staging2:
  + No duplicates
  + Consistent formats
  + Cleaned text and dates
  + Meaningful null handling