Cleaning Layoffs Data for Advanced Analytics Using SQL

1. Overview

This SQL script is designed to clean the dataset `world_layoffs.layoffs`. The dataset contains information on company layoffs, and the main goal is to ensure the data is clean, standardized, and ready for analysis. The cleaning process includes:

- Removing duplicates
- Standardizing data
- Handling null values
- Removing unnecessary columns and rows

Each step in the cleaning process is outlined below.

---

2. Creating a Staging Table

To preserve data integrity, a staging table is created to perform the cleaning process. This ensures that the raw data remains untouched.

```sql
CREATE TABLE world_layoffs.layoffs_staging

LIKE world_layoffs.layoffs;

INSERT INTO world_layoffs.layoffs_staging
```

```
SELECT * FROM world_layoffs.layoffs;
```

---

3. Data Cleaning Process

3.1 Remove Duplicates

Duplicate records are removed using the `ROW_NUMBER()` function, which identifies rows with the same company, industry, total laid-off, and date. Rows where `row_num > 1` are considered duplicates and are removed.

```sql
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
    ) AS row_num
FROM world_layoffs.layoffs_staging;
```

Rows with `row_num >= 2` are deleted, and a new cleaned table `layoffs_staging2` is created.

```sql
```

```
CREATE TABLE world_layoffs.layoffs_staging2 (
    `company` text,
    `location` text,
    `industry` text,
    `total_laid_off` INT,
    `percentage_laid_off` text,
    `date` text,
    `stage` text,
    `country` text,
    `funds_raised_millions` int,
    `row_num` INT
);

INSERT INTO world_layoffs.layoffs_staging2
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
    ) AS row_num
FROM world_layoffs.layoffs_staging;

DELETE FROM world_layoffs.layoffs_staging2
WHERE row_num >= 2;
```

3.2 Standardize Data

3.2.1 Handle Null or Empty Industry Values

Null or empty values in the `industry` column are identified and updated. For companies with multiple records, null values are replaced by non-null values from other rows.

```sql
UPDATE world_layoffs.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;
```

3.2.2 Standardize Industry Naming

Industry names like "Crypto Currency" and "CryptoCurrency" are unified to "Crypto."

```sql
UPDATE layoffs_staging2
SET industry = 'Crypto'
WHERE industry IN ('Crypto Currency', 'CryptoCurrency');
```

3.2.3 Standardize Country Names

Country names are cleaned by removing unnecessary punctuation, such as trailing periods.

```sql
UPDATE layoffs_staging2
SET country = TRIM(TRAILING '.' FROM country);
```

3.2.4 Standardize Date Format

The `date` column is converted from string to `DATE` format.

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

ALTER TABLE layoffs_staging2
MODIFY COLUMN `date` DATE;
```

3.3 Handle Null Values

Null values in key columns such as `total_laid_off`, `percentage_laid_off`, and `funds_raised_millions` are left unchanged to facilitate accurate calculations during exploratory data analysis (EDA).

```sql
SELECT * FROM world_layoffs.layoffs_staging2 WHERE total_laid_off IS NULL;
```

3.4 Remove Unnecessary Rows

Rows with both `total_laid_off` and `percentage_laid_off` as `NULL` are deleted, as they provide no useful data.

```sql
DELETE FROM world_layoffs.layoffs_staging2
WHERE total_laid_off IS NULL
AND percentage_laid_off IS NULL;
```

3.5 Remove Unnecessary Columns

The `row_num` column, used for duplicate detection, is no longer needed and is dropped.

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

---

4. Final Dataset

The cleaned dataset `layoffs_staging2` is now ready for analysis. All duplicates, unnecessary rows, and columns have been removed, and the data has been standardized.

```sql
SELECT *
FROM world_layoffs.layoffs_staging2;
```

---

5. Summary

- Staging Table Creation: A working table `layoffs_staging` was created to keep the raw data intact.

- Duplicate Removal: Duplicates were identified and removed using `ROW_NUMBER()`.

- Data Standardization: Industry names, country names, and date formats were standardized.

- Handling Null Values: Null values were preserved for important columns, except in cases where both `total_laid_off` and `percentage_laid_off` were null.

- Clean Table: The cleaned table `layoffs_staging2` is the final version, ready for analysis.