1. **Creating a table**

CREATE TABLE public\_us\_accident\_data (

id SERIAL PRIMARY KEY,

source VARCHAR(50),

tmc INTEGER,

severity INTEGER,

start\_time TIMESTAMP,

end\_time TIMESTAMP,

start\_lat DOUBLE PRECISION,

start\_lng DOUBLE PRECISION,

end\_lat DOUBLE PRECISION,

end\_lng DOUBLE PRECISION,

distance\_mi DOUBLE PRECISION,

description TEXT,

number INTEGER,

street VARCHAR(100),

side VARCHAR(10),

city VARCHAR(50),

county VARCHAR(50),

state VARCHAR(10),

zip\_code VARCHAR(10),

country VARCHAR(50),

timezone VARCHAR(50),

airport\_code VARCHAR(10),

weather\_timestamp TIMESTAMP,

temperature\_f DOUBLE PRECISION,

wind\_chill\_f DOUBLE PRECISION,

humidity\_percent DOUBLE PRECISION,

pressure\_in DOUBLE PRECISION,

visibility\_mi DOUBLE PRECISION,

wind\_direction VARCHAR(50),

wind\_speed\_mph DOUBLE PRECISION,

precipitation\_in DOUBLE PRECISION,

weather\_condition VARCHAR(100),

amenity BOOLEAN,

bump BOOLEAN,

crossing BOOLEAN,

give\_way BOOLEAN,

junction BOOLEAN,

no\_exit BOOLEAN,

railway BOOLEAN,

roundabout BOOLEAN,

station BOOLEAN,

stop BOOLEAN,

traffic\_calming BOOLEAN,

traffic\_signal BOOLEAN,

turning\_loop BOOLEAN,

sunrise\_sunset VARCHAR(10),

civil\_twilight VARCHAR(10),

nautical\_twilight VARCHAR(10),

astronomical\_twilight VARCHAR(10)

);

**1. Data Cleaning**

**A. Check for Missing Values**

* **Identify Missing Values:** Determine which columns have missing values and decide how to handle them.

As a group we decided to use removing the rows with missing values. For that we have implemented the following Query on SQL Shell(psql) with this we managed to delete 764142 rows are affected

DELETE FROM us\_accident\_data

WHERE

(end\_lat IS NULL) OR

(end\_lng IS NULL) OR

(precipitation\_in IS NULL) OR

(wind\_chill\_f IS NULL) OR

(wind\_speed\_mph IS NULL) OR

(visibility\_mi IS NULL) OR

(humidity\_percent IS NULL) OR

(pressure\_in IS NULL) OR

(temperature\_f IS NULL) OR

(weather\_timestamp IS NULL OR TRIM(weather\_timestamp::TEXT) = '') OR

(nautical\_twilight IS NULL OR nautical\_twilight = '') OR

(civil\_twilight IS NULL OR civil\_twilight = '') OR

(sunrise\_sunset IS NULL OR sunrise\_sunset = '') OR

(astronomical\_twilight IS NULL OR astronomical\_twilight = '') OR

(airport\_code IS NULL OR airport\_code = '') OR

(street IS NULL OR street = '') OR

(timezone IS NULL OR timezone = '') OR

(zipcode IS NULL OR zipcode = '') OR

(city IS NULL OR city = '') OR

(description IS NULL OR description = '') OR

(traffic\_signal IS NULL) OR

(roundabout IS NULL) OR

(station IS NULL) OR

(stop IS NULL) OR

(traffic\_calming IS NULL) OR

(country IS NULL OR country = '') OR

(turning\_loop IS NULL) OR

(no\_exit IS NULL) OR

(end\_time IS NULL OR TRIM(end\_time::TEXT) = '') OR

(start\_time IS NULL OR TRIM(start\_time::TEXT) = '') OR

(severity IS NULL) OR

(railway IS NULL) OR

(crossing IS NULL) OR

(junction IS NULL) OR

(give\_way IS NULL) OR

(bump IS NULL) OR

(amenity IS NULL) OR

(start\_lat IS NULL) OR

(start\_lng IS NULL) OR

(distance\_mi IS NULL) OR

(source IS NULL OR source = '') OR

(county IS NULL OR county = '') OR

(state IS NULL OR state = '') OR

(id IS NULL OR id = '');

**B. Handle Outliers**

* **Identify Outliers:** Use statistical methods to find outliers.
* **Remove or Transform Outliers:**

To identify and display the outliers we used

WITH iqr\_stats AS (

SELECT

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY temperature\_f::DOUBLE PRECISION) AS q1\_temp,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY temperature\_f::DOUBLE PRECISION) AS q3\_temp,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY wind\_speed\_mph::DOUBLE PRECISION) AS q1\_wind,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY wind\_speed\_mph::DOUBLE PRECISION) AS q3\_wind,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY visibility\_mi::DOUBLE PRECISION) AS q1\_visibility,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY visibility\_mi::DOUBLE PRECISION) AS q3\_visibility,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY humidity\_percent::DOUBLE PRECISION) AS q1\_humidity,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY humidity\_percent::DOUBLE PRECISION) AS q3\_humidity,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY pressure\_in::DOUBLE PRECISION) AS q1\_pressure,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY pressure\_in::DOUBLE PRECISION) AS q3\_pressure,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY precipitation\_in::DOUBLE PRECISION) AS q1\_precipitation,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY precipitation\_in::DOUBLE PRECISION) AS q3\_precipitation,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY wind\_chill\_f::DOUBLE PRECISION) AS q1\_wind\_chill,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY wind\_chill\_f::DOUBLE PRECISION) AS q3\_wind\_chill,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY distance\_mi::DOUBLE PRECISION) AS q1\_distance,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY distance\_mi::DOUBLE PRECISION) AS q3\_distance,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY start\_lat::DOUBLE PRECISION) AS q1\_start\_lat,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY start\_lat::DOUBLE PRECISION) AS q3\_start\_lat,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY start\_lng::DOUBLE PRECISION) AS q1\_start\_lng,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY start\_lng::DOUBLE PRECISION) AS q3\_start\_lng,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY end\_lat::DOUBLE PRECISION) AS q1\_end\_lat,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY end\_lat::DOUBLE PRECISION) AS q3\_end\_lat,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY end\_lng::DOUBLE PRECISION) AS q1\_end\_lng,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY end\_lng::DOUBLE PRECISION) AS q3\_end\_lng

FROM us\_accident\_data

)

SELECT \*,

q1\_temp, q3\_temp,

q1\_wind, q3\_wind,

q1\_visibility, q3\_visibility,

q1\_humidity, q3\_humidity,

q1\_pressure, q3\_pressure,

q1\_precipitation, q3\_precipitation,

q1\_wind\_chill, q3\_wind\_chill,

q1\_distance, q3\_distance,

q1\_start\_lat, q3\_start\_lat,

q1\_start\_lng, q3\_start\_lng,

q1\_end\_lat, q3\_end\_lat,

q1\_end\_lng, q3\_end\_lng

FROM us\_accident\_data, iqr\_stats

WHERE

temperature\_f::DOUBLE PRECISION < (q1\_temp - 1.5 \* (q3\_temp - q1\_temp))

OR temperature\_f::DOUBLE PRECISION > (q3\_temp + 1.5 \* (q3\_temp - q1\_temp))

OR wind\_speed\_mph::DOUBLE PRECISION < (q1\_wind - 1.5 \* (q3\_wind - q1\_wind))

OR wind\_speed\_mph::DOUBLE PRECISION > (q3\_wind + 1.5 \* (q3\_wind - q1\_wind))

OR visibility\_mi::DOUBLE PRECISION < (q1\_visibility - 1.5 \* (q3\_visibility - q1\_visibility))

OR visibility\_mi::DOUBLE PRECISION > (q3\_visibility + 1.5 \* (q3\_visibility - q1\_visibility))

OR humidity\_percent::DOUBLE PRECISION < (q1\_humidity - 1.5 \* (q3\_humidity - q1\_humidity))

OR humidity\_percent::DOUBLE PRECISION > (q3\_humidity + 1.5 \* (q3\_humidity - q1\_humidity))

OR pressure\_in::DOUBLE PRECISION < (q1\_pressure - 1.5 \* (q3\_pressure - q1\_pressure))

OR pressure\_in::DOUBLE PRECISION > (q3\_pressure + 1.5 \* (q3\_pressure - q1\_pressure))

OR precipitation\_in::DOUBLE PRECISION < (q1\_precipitation - 1.5 \* (q3\_precipitation - q1\_precipitation))

OR precipitation\_in::DOUBLE PRECISION > (q3\_precipitation + 1.5 \* (q3\_precipitation - q1\_precipitation))

OR wind\_chill\_f::DOUBLE PRECISION < (q1\_wind\_chill - 1.5 \* (q3\_wind\_chill - q1\_wind\_chill))

OR wind\_chill\_f::DOUBLE PRECISION > (q3\_wind\_chill + 1.5 \* (q3\_wind\_chill - q1\_wind\_chill))

OR distance\_mi::DOUBLE PRECISION < (q1\_distance - 1.5 \* (q3\_distance - q1\_distance))

OR distance\_mi::DOUBLE PRECISION > (q3\_distance + 1.5 \* (q3\_distance - q1\_distance))

OR start\_lat::DOUBLE PRECISION < (q1\_start\_lat - 1.5 \* (q3\_start\_lat - q1\_start\_lat))

OR start\_lat::DOUBLE PRECISION > (q3\_start\_lat + 1.5 \* (q3\_start\_lat - q1\_start\_lat))

OR start\_lng::DOUBLE PRECISION < (q1\_start\_lng - 1.5 \* (q3\_start\_lng - q1\_start\_lng))

OR start\_lng::DOUBLE PRECISION > (q3\_start\_lng + 1.5 \* (q3\_start\_lng - q1\_start\_lng))

OR end\_lat::DOUBLE PRECISION < (q1\_end\_lat - 1.5 \* (q3\_end\_lat - q1\_end\_lat))

OR end\_lat::DOUBLE PRECISION > (q3\_end\_lat + 1.5 \* (q3\_end\_lat - q1\_end\_lat))

OR end\_lng::DOUBLE PRECISION < (q1\_end\_lng - 1.5 \* (q3\_end\_lng - q1\_end\_lng))

OR end\_lng::DOUBLE PRECISION > (q3\_end\_lng + 1.5 \* (q3\_end\_lng - q1\_end\_lng));

**C. To delete the outliers**

-- Calculate IQR and bounds for outliers

WITH iqr\_stats AS (

SELECT

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY temperature\_f::DOUBLE PRECISION) AS q1\_temp,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY temperature\_f::DOUBLE PRECISION) AS q3\_temp,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY wind\_speed\_mph::DOUBLE PRECISION) AS q1\_wind,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY wind\_speed\_mph::DOUBLE PRECISION) AS q3\_wind,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY visibility\_mi::DOUBLE PRECISION) AS q1\_visibility,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY visibility\_mi::DOUBLE PRECISION) AS q3\_visibility,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY humidity\_percent::DOUBLE PRECISION) AS q1\_humidity,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY humidity\_percent::DOUBLE PRECISION) AS q3\_humidity,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY pressure\_in::DOUBLE PRECISION) AS q1\_pressure,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY pressure\_in::DOUBLE PRECISION) AS q3\_pressure,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY precipitation\_in::DOUBLE PRECISION) AS q1\_precipitation,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY precipitation\_in::DOUBLE PRECISION) AS q3\_precipitation,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY wind\_chill\_f::DOUBLE PRECISION) AS q1\_wind\_chill,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY wind\_chill\_f::DOUBLE PRECISION) AS q3\_wind\_chill,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY distance\_mi::DOUBLE PRECISION) AS q1\_distance,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY distance\_mi::DOUBLE PRECISION) AS q3\_distance,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY start\_lat::DOUBLE PRECISION) AS q1\_start\_lat,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY start\_lat::DOUBLE PRECISION) AS q3\_start\_lat,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY start\_lng::DOUBLE PRECISION) AS q1\_start\_lng,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY start\_lng::DOUBLE PRECISION) AS q3\_start\_lng,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY end\_lat::DOUBLE PRECISION) AS q1\_end\_lat,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY end\_lat::DOUBLE PRECISION) AS q3\_end\_lat,

PERCENTILE\_CONT(0.25) WITHIN GROUP (ORDER BY end\_lng::DOUBLE PRECISION) AS q1\_end\_lng,

PERCENTILE\_CONT(0.75) WITHIN GROUP (ORDER BY end\_lng::DOUBLE PRECISION) AS q3\_end\_lng

FROM us\_accident\_data

)

-- Delete outliers based on the IQR method

DELETE FROM us\_accident\_data

USING iqr\_stats

WHERE

temperature\_f::DOUBLE PRECISION < (iqr\_stats.q1\_temp - 1.5 \* (iqr\_stats.q3\_temp - iqr\_stats.q1\_temp))

OR temperature\_f::DOUBLE PRECISION > (iqr\_stats.q3\_temp + 1.5 \* (iqr\_stats.q3\_temp - iqr\_stats.q1\_temp))

OR wind\_speed\_mph::DOUBLE PRECISION < (iqr\_stats.q1\_wind - 1.5 \* (iqr\_stats.q3\_wind - iqr\_stats.q1\_wind))

OR wind\_speed\_mph::DOUBLE PRECISION > (iqr\_stats.q3\_wind + 1.5 \* (iqr\_stats.q3\_wind - iqr\_stats.q1\_wind))

OR visibility\_mi::DOUBLE PRECISION < (iqr\_stats.q1\_visibility - 1.5 \* (iqr\_stats.q3\_visibility - iqr\_stats.q1\_visibility))

OR visibility\_mi::DOUBLE PRECISION > (iqr\_stats.q3\_visibility + 1.5 \* (iqr\_stats.q3\_visibility - iqr\_stats.q1\_visibility))

OR humidity\_percent::DOUBLE PRECISION < (iqr\_stats.q1\_humidity - 1.5 \* (iqr\_stats.q3\_humidity - iqr\_stats.q1\_humidity))

OR humidity\_percent::DOUBLE PRECISION > (iqr\_stats.q3\_humidity + 1.5 \* (iqr\_stats.q3\_humidity - iqr\_stats.q1\_humidity))

OR pressure\_in::DOUBLE PRECISION < (iqr\_stats.q1\_pressure - 1.5 \* (iqr\_stats.q3\_pressure - iqr\_stats.q1\_pressure))

OR pressure\_in::DOUBLE PRECISION > (iqr\_stats.q3\_pressure + 1.5 \* (iqr\_stats.q3\_pressure - iqr\_stats.q1\_pressure))

OR precipitation\_in::DOUBLE PRECISION < (iqr\_stats.q1\_precipitation - 1.5 \* (iqr\_stats.q3\_precipitation - iqr\_stats.q1\_precipitation))

OR precipitation\_in::DOUBLE PRECISION > (iqr\_stats.q3\_precipitation + 1.5 \* (iqr\_stats.q3\_precipitation - iqr\_stats.q1\_precipitation))

OR wind\_chill\_f::DOUBLE PRECISION < (iqr\_stats.q1\_wind\_chill - 1.5 \* (iqr\_stats.q3\_wind\_chill - iqr\_stats.q1\_wind\_chill))

OR wind\_chill\_f::DOUBLE PRECISION > (iqr\_stats.q3\_wind\_chill + 1.5 \* (iqr\_stats.q3\_wind\_chill - iqr\_stats.q1\_wind\_chill))

OR distance\_mi::DOUBLE PRECISION < (iqr\_stats.q1\_distance - 1.5 \* (iqr\_stats.q3\_distance - iqr\_stats.q1\_distance))

OR distance\_mi::DOUBLE PRECISION > (iqr\_stats.q3\_distance + 1.5 \* (iqr\_stats.q3\_distance - iqr\_stats.q1\_distance))

OR start\_lat::DOUBLE PRECISION < (iqr\_stats.q1\_start\_lat - 1.5 \* (iqr\_stats.q3\_start\_lat - iqr\_stats.q1\_start\_lat))

OR start\_lat::DOUBLE PRECISION > (iqr\_stats.q3\_start\_lat + 1.5 \* (iqr\_stats.q3\_start\_lat - iqr\_stats.q1\_start\_lat))

OR start\_lng::DOUBLE PRECISION < (iqr\_stats.q1\_start\_lng - 1.5 \* (iqr\_stats.q3\_start\_lng - iqr\_stats.q1\_start\_lng))

OR start\_lng::DOUBLE PRECISION > (iqr\_stats.q3\_start\_lng + 1.5 \* (iqr\_stats.q3\_start\_lng - iqr\_stats.q1\_start\_lng))

OR end\_lat::DOUBLE PRECISION < (iqr\_stats.q1\_end\_lat - 1.5 \* (iqr\_stats.q3\_end\_lat - iqr\_stats.q1\_end\_lat))

OR end\_lat::DOUBLE PRECISION > (iqr\_stats.q3\_end\_lat + 1.5 \* (iqr\_stats.q3\_end\_lat - iqr\_stats.q1\_end\_lat))

OR end\_lng::DOUBLE PRECISION < (iqr\_stats.q1\_end\_lng - 1.5 \* (iqr\_stats.q3\_end\_lng - iqr\_stats.q1\_end\_lng))

OR end\_lng::DOUBLE PRECISION > (iqr\_stats.q3\_end\_lng + 1.5 \* (iqr\_stats.q3\_end\_lng - iqr\_stats.q1\_end\_lng));

Combining the Boolean to one column

-- Add the new column 'road\_type' to the table

ALTER TABLE us\_accident\_data

ADD COLUMN road\_type TEXT;

-- Update the 'road\_type' column based on boolean columns

UPDATE us\_accident\_data

SET road\_type = CASE

WHEN roundabout THEN 'Roundabout'

WHEN station THEN 'Station'

WHEN stop THEN 'Stop'

WHEN traffic\_signal THEN 'Traffic Signal'

WHEN traffic\_calming THEN 'Traffic Calming'

WHEN turning\_loop THEN 'Turning Loop'

WHEN no\_exit THEN 'No Exit'

WHEN railway THEN 'Railway'

WHEN crossing THEN 'Crossing'

WHEN junction THEN 'Junction'

WHEN give\_way THEN 'Give Way'

WHEN bump THEN 'Bump'

WHEN amenity THEN 'Amenity'

ELSE 'Other' -- Use this to handle cases where none of the booleans are true

END;

Droping the columns with Boolean data type

ALTER TABLE us\_accident\_data

DROP COLUMN roundabout,

DROP COLUMN station,

DROP COLUMN stop,

DROP COLUMN traffic\_signal,

DROP COLUMN traffic\_calming,

DROP COLUMN turning\_loop,

DROP COLUMN no\_exit,

DROP COLUMN railway,

DROP COLUMN crossing,

DROP COLUMN junction,

DROP COLUMN give\_way,

DROP COLUMN bump,

DROP COLUMN amenity;

To see if the above action is working or not

SELECT column\_name, data\_type

FROM information\_schema.columns

WHERE table\_name = 'us\_accident\_data';

To drop other columns with less relevant for our analysis

ALTER TABLE us\_accident\_data

DROP COLUMN source,

DROP COLUMN end\_lat,

DROP COLUMN end\_lng,

DROP COLUMN description,

DROP COLUMN county,

DROP COLUMN country,

DROP COLUMN airport\_code,

DROP COLUMN wind\_direction;

DROP COLUMN nautical\_twilight,

DROP COLUMN astronomical\_twilight;