Data Wrangling using SQL

Data wrangling describes a series of processes designed to explore, transform, and validate raw datasets from their messy and complex forms into high-quality data with good integrity and consistency into produce valuable insights and guide business decisions in later analytics purposes.

For this Data Wrangling Project, I'm using electronic store sales dataset from Kaggle: https://www.kaggle.com/datasets/saumaydhaundiyal/electronic-store-sales-data

For the steps I'm using for this project are:

- 1. Data Discovery
- 2. Data Cleaning
- 3. Data Transformation
- 4. Data Enriching
- 5. Data Validating
- 6. Data Publishing

Data Discovery

The main purposes in this step will be:

- Import data from our local machine
- Gather useful insight & information for future step.

Importing Data

In this step, instead using Table Import Wizard in MySQL Workbench, we will be using MySQL console. The reason is because using Table Import Wizard is inefficient to importing large dataset (the progress is too slow)

With this method we need to create our empty table to store the data later. We set our preferred datatype and match the column from our raw dataset.

Then, since our data is in separated csv files. We need to merge it first (we can perform this using python or just CMD console)

```
C:\Users\renal>cd "C:/Users/renal/Documents/Renaldo's File/Data Analyst Portofolio -Renaldo
 Livando/Project2 Data Wrangling using SQL/Electronic Stores Sales Raw Data 2019"
C:\Users\renal\Documents\Renaldo's File\Data Analyst Portofolio -Renaldo Livando\Project2 D
ata Wrangling using SQL\Electronic Stores Sales Raw Data 2019>copy Sales*.csv merged_electr
onic_sales_data.csv
Sales_April_2019.csv
Sales_August_2019.csv
Sales_December_2019.csv
Sales_February_2019.csv
Sales_January_2019.csv
Sales_July_2019.csv
Sales_June_2019.csv
Sales_March_2019.csv
Sales_May_2019.csv
Sales_November_2019.csv
Sales_October_2019.csv
Sales_September_2019.csv
        1 file(s) copied.
```

Next, we need to make sure our local_infile variable is activate, then restart our console and login with local_infile used.

```
Microsoft Windows [Version 10.0.22631.2191]
(c) Microsoft Corporation. All rights reserved.

C:\Users\renal>mysql --local-infile=1 -u root -p
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with; or \g.
Your MySQL connection id is 11
Server version: 8.0.33 MySQL Community Server - GPL
```

And the last step in this step will load the data from raw dataset into our table that we created.

```
mysql> LOAD DATA LOCAL INFILE "C:/Users/renal/Documents/Renaldo's File/Data Analyst Portofolio -Renaldo Livando/Project2

Data Wrangling using SQL/merged_electronic_sales_data.csv"

-> INTO TABLE merged_sales_data
-> FIELDS TERMINATED BY ","

-> ENCLOSED BY '"'

-> LINES TERMINATED BY '\r\n'

-> IGNORE 1 ROWS

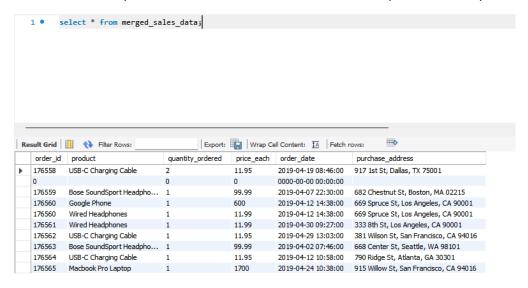
-> (order_id,product ,quantity_ordered,price_each,@order_date ,purchase_address )

-> set order_date = STR_TO_DATE(@order_date,'%m/%d/%y %H:%i');

Query OK, 186850 rows affected, 3055 warnings (3.58 sec)

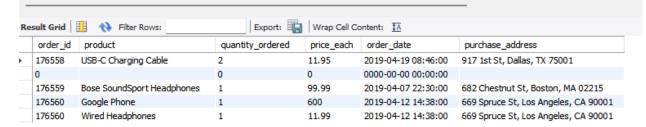
Records: 186850 Deleted: 0 Skipped: 0 Warnings: 3055
```

Give a check on MySQL Workbench to make sure the data is imported correctly.



Check and review our dataset

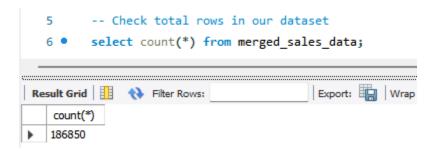
```
1   -- Select our dataset head
2   select * from merged_sales_data
3   LIMIT 5;
4
5
```



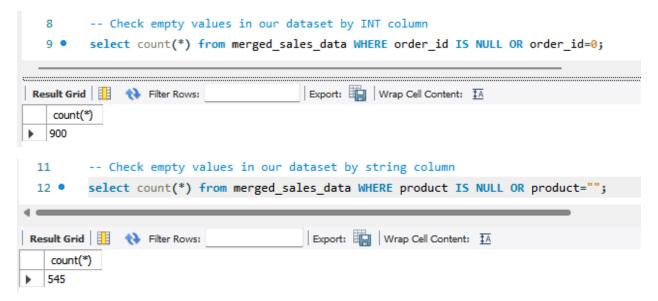
Insight from dataset review:

- All the column in the dataset is necessary for our later analysis, there's no need to commit column filtering
- That seems we need to extract the data from column "Order Date" and "Purchase Address" to make new column like "City", "Postal Code", "Month", etc.
- We can add new calculated column from column "Quantity Ordered" and "Price Each"

Check dataset total rows

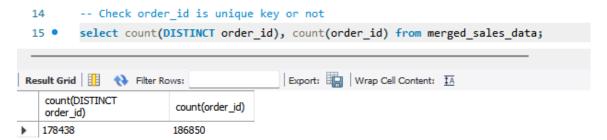


Check empty values in our dataset. We will use 2 columns with different data type to check this.



From result we got in here. We can conclude that the number of missing values quite high. And because the order_id column occur more missing values than the product column, we will be using order_id column to remove the missing values later instead of using string/text column.

Check if our dataset ID is unique key or not



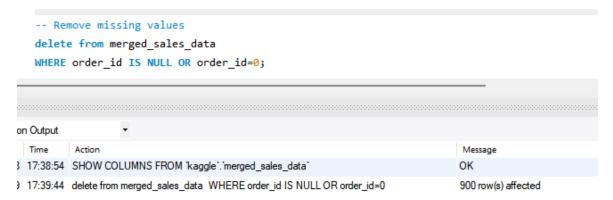
The number of rows is different. Means ID column in our dataset not unique.

Data Cleaning

The main purposes in this step will be remove errors that might distort or damage the accuracy of your analysis. This includes tasks like standardizing inputs, handling empty cells or missing values, handling duplicate values and fixing string data value by perform trimming.

Remove missing values

With insight from our Data Discovery, we already decide way to handle our data missing values is by removing it.



Handle duplicate values

We need to perform checking for decide action to handling the duplicate values.

Checking duplicate values

Check count of duplicate values occurred. We will using row number method.

```
-- Check duplicate values accross all columns
 6 • ⊖ select count(*) from (
           SELECT
 7
 8
               ROW_NUMBER() OVER (
 9
                   PARTITION BY order_id, product, quantity_ordered,
 10
 11
                                  price_each, order_date, purchase_address
                   ORDER BY order_id) AS row_num
12
 13
 14
               merged_sales_data
 15
 16
               where row_num>1;
Export: Wrap Cell Content: IA
  count(*)
 264
```

With this check, we found there's occur 264 duplicate values. Before we decide a way to handle these duplicate values, we need to understand and interpret the dataset.

- The data is recording any purchase in the store into a row.
- One Order ID should have One Order Date.
- One Order ID should have One Purchase Address.
- One Order ID could have several different products.

With these 4 points, we can conclude that if Order ID have same Product recorded different rows doesn't make sense. So, we decide to remove duplicate rows that duplicated across all column.

Remove duplicate value

```
18
         -- Remove duplicate values
        DELIMITER $$
 20 • ① CREATE TABLE `merged sales data copy` (
 28
 29
         INSERT INTO merged_sales_data_copy
         select * from merged sales data
 30
 31
         group by order_id, product, quantity_ordered, price_each, order_date, purchase_address;
 32
 33
         DROP TABLE merged sales data;
 34
         ALTER TABLE merged_sales_data_copy RENAME TO merged_sales_data;
         SELECT COUNT(*) FROM merged sales data;
 35
 36
         $$
Result Grid | Filter Rows:
                                     Export: Wrap Cell Content: IA
   COUNT(*)
185686
```

Trim string column

We need perform trimming into our string column in case we got unnecessary space inside our value

```
-- Trim and remove double space from string column
39 •
       update merged_sales_data

    set product = trim(REPLACE(
41
                                 REPLACE(
                                     REPLACE(product,
42
43
                                     ' ','<>'),
                                 '×',''),
44
                             '<>',' '));
45
46
47
       update merged_sales_data

    ⇒ set product = trim(REPLACE(
48
49
                                 REPLACE(
50
                                     REPLACE(purchase_address,
                                     ' ','<>'),
51
                                 '><',''),
52
```

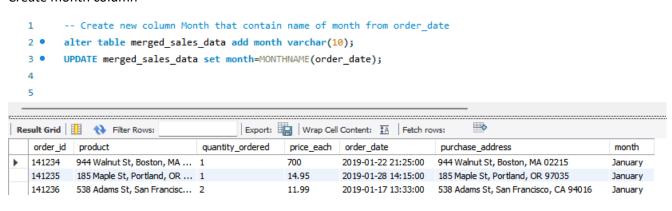
Data Transformation

The main purposes in this step will be:

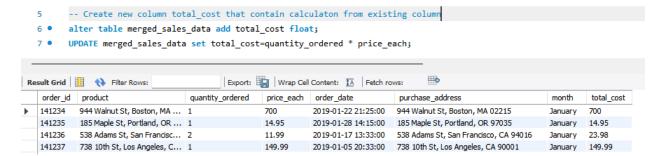
- Attribute Construction: in which new attributes are added or created from existing attributes
- Generalization: where low-level data attributes are converted into high-level data attributes (in this project is "Purchase Address" column).
- Re-Arrange Column

Attribute Construction

Create month column



Create calculation column total_cost



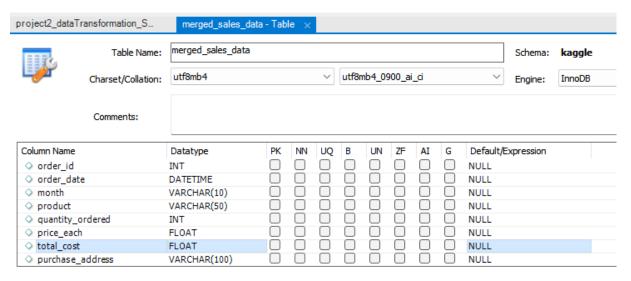
Generalization

We perform generalization on purchase_address column

```
9
        -- Perform generalization on purchase_address column
10
        DELIMITER $$
11 •
       alter table merged sales data add street VARCHAR(30);
        update merged_sales_data set street= SUBSTRING_INDEX(purchase_address, ',', 1);
12
        alter table merged_sales_data add city VARCHAR(30);
13
        update merged_sales_data set city= SUBSTRING_INDEX(SUBSTRING_INDEX(purchase_address, ',', 2),',',-1);
15
        alter table merged sales data add state VARCHAR(2);
        update merged_sales_data set state= LEFT(TRIM(SUBSTRING_INDEX(purchase_address, ',', -1)),2);
16
        alter table merged sales data add postal code VARCHAR(5);
17
        update merged_sales_data set postal_code=RIGHT(TRIM(SUBSTRING_INDEX(purchase_address, ',', -1)),5);
        select * from merged_sales_data;
        $$
20
Result Grid Filter Rows:
                                       Export: Wrap Cell Content: TA Fetch rows:
              quantity_ordered
                                                            purchase_address
                                                                                                                                             state postal_code
                               price_each order_date
                                                                                               month
                                                                                                       total_cost street
                                                                                                                                city
  Boston, MA ... 1
                               700
                                          2019-01-22 21:25:00 944 Walnut St, Boston, MA 02215
                                                                                              January
                                                                                                       700
                                                                                                                 944 Walnut St
                                                                                                                               Boston
                                                                                                                                            MΑ
                                                                                                                                                   02215
  ortland, OR ... 1
                               14.95
                                       2019-01-28 14:15:00 185 Maple St, Portland, OR 97035
                                                                                              January 14.95
                                                                                                                185 Maple St
                                                                                                                               Portland
                                                                                                                                            OR
                                                                                                                                                   97035
  San Francisc... 2
                               11.99
                                          2019-01-17 13:33:00 538 Adams St, San Francisco, CA 94016
                                                                                                       23.98
                                                                                                                 538 Adams St
                                                                                                                                                   94016
                                                                                              January
                                                                                                                                San Francisco
                                                                                                                                            CA
                               149.99 2019-01-05 20:33:00 738 10th St, Los Angeles, CA 90001
                                                                                                                738 10th St
  s Angeles, C... 1
                                                                                              January 149.99
                                                                                                                               Los Angeles
                                                                                                                                                   90001
```

Re-Arrange

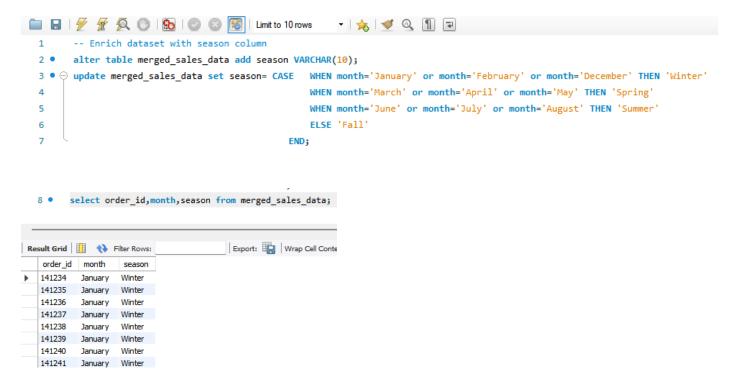
I'm using MySQL Workbench feature to easily re-arrange this by drag it into my preferred order.



This setting will execute syntax below:

Data Enriching

The main purpose in this step will be: enrich the dataset by adding values from other datasets.



Data Publishing

In MySQL, exported data by default won't export our table header. We need to add our header in first rows using UNION.

```
-- List our column name
 2 •
      SELECT group_concat(CONCAT("'",COLUMN_NAME,"'") ORDER BY ORDINAL_POSITION SEPARATOR ',')
       FROM INFORMATION_SCHEMA.COLUMNS
       WHERE TABLE_NAME = 'merged_sales_data'
       AND TABLE_SCHEMA = 'kaggle';
       -- Union and export our table to csv file
 8 • SELECT 'order_id', 'order_date', 'month', 'product', 'quantity_ordered', 'price_each', 'total_cost', 'purchase_address', 'street', 'city', 'state', 'postal_code
10
      SELECT * FROM merged_sales_data
      INTO OUTFILE "C:/Users/renal/Documents/Renaldo's File/Data Analyst Portofolio -Renaldo Livando/Project2 Data Wrangling using SQL/out/cleaned_electron
11
12
      FIELDS ENCLOSED BY '"
13
       TERMINATED BY ','
       LINES TERMINATED BY '\r\n';
```