# Assumptions

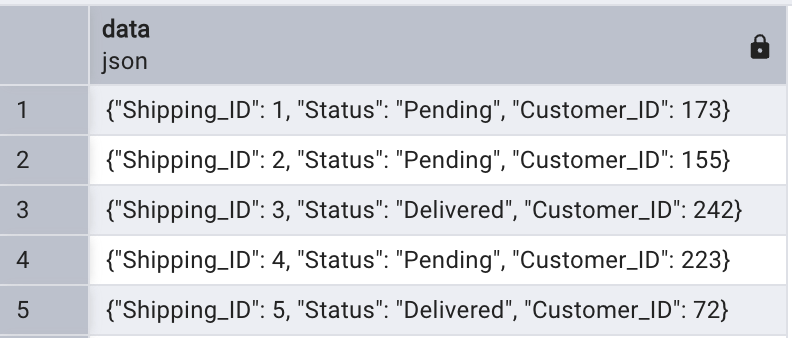
1. In customers table, Age should be > 0 and < 120
2. In customers table, first\_name & last\_name should not contain any numbers or special characters
3. In the customers table, customer\_id is unique for each customer and it can't have duplicates.
4. Assumption is that a person with the same first\_name, last\_name and age can exist in the same country provided they have a unique customer id.
5. In the orders table, amount should be always greater than 0
6. In the orders table, order\_id should be unique in the entire table
7. Assumption is that a customer can order same item with same amount more than once therefore uniqueness test is only applicable for order\_id column
8. There’s no direct link between shippings and orders therefore shipping status would be considered as a dimension as it does not have link to order and any shipping date associated.
9. In shipping table a customer can only have either Delivered or Pending status if for a customer we have more than one value or Delivered/Pending then that would be considered as duplicate
10. Not all customers have shipping data
11. Not all costumes have orders data

# Step 1: Loading Data into PostgreSQL

The source data is provided in three files **Customer.xls**, **Order.csv** & **Shipping.json.** In order to explore the data for data quality checks (accuracy, completeness & reliability) I have loaded it into **PostgreSQL** as tables using **Python**. Python Code used to load the data can be seen here.

# Step 2: Creating shippings table from JSON

* The shipping.json file contains an array/list of JSON objects. I have loaded it into the database as one column table with one row per JSON object.



* I have used the following code to convert it into a flat table where each key will become a column and values would become the data in each row:

| CREATE TABLE sales.shippings AS  SELECT   CAST (data->>'Shipping\_ID' AS INT) AS shipping\_id,  data->>'Status' AS status,  CAST(data->>'Customer\_ID' AS INT) AS customer\_id FROM sales.shipping\_json ; |
| --- |

# Step 3: Data Quality Checks (Accuracy, Completeness & Reliability):

In this step I have explored all three data tables for below listed data quality checks. SQL code for data quality checks can be found here.

## Accuracy

* **Valid Value Ranges:** In this test I have looked at the **customers.age** column and **orders.amount** column to see if they contain valid data i.e. no negatives, 0 and out of range values.
* **Correct Formats:** In this test I have checked all the TEXT columns to see if they have any Special Characters or numeric values. Columns for which this test is performed are listed below:
  + customers.first\_name
  + customers.last\_name
  + orders.item
* **Accepted Values:** This test checks if we have any unwanted or strange data in the columns or If a column has any un-wanted values that it’s not intended to have. This test is performed on below listed columns:
  + customers.country
  + orders.items
  + shippings.status

## Completeness

* **Test for NULLS OR BLANKS:** This test helps in identifying the columns that are not supposed to have NULL/BLANKS but still have them. This test is applied on almost all the columns of three tables except **customers.last\_name** column.
* **Referential Integrity:** This test is performed on **orders** & **shippings** tables. It checks for each record in these tables and we have an equivalent record in the **customers** table. These tables contain **FOREIGN KEY** to the **customers** table and in order to maintain the referential integrity each **customer\_id** in both the tables should have a customer record in the **customers** table.

## Reliability

* **Duplicates:** This test looks for duplicate rows in each table. For this test for each table I have made below listed assumptions to check for the duplicates**:**
  + **Customers**
    - **customer\_id** should be unique for each customer
    - People with same **first\_name**, **last\_name** and **age** can exist in the same country provided they have unique **customer\_id**
  + **Orders**
    - **order\_id** column should not contain duplicate **order\_id**
    - For a **customer\_id** we can have multiple orders with the same **item** & **amount**.
  + **Shippings**
    - **shipping\_id** column should not contain duplicate **shipping\_id.**
    - For a **customer\_id** we can’t have duplicate shipping statuses aka combinations of **customer\_id** & **status** should be unique
* **Consistent Data Types:** For this test we have to make sure that while creating the table we should have the right data types assigned so that JOIN would be easy & fast. Data Types mapping for each table are listed below & columns that are used for joining the tables are highlighted in red.:  
  + **Customers**:
    - customer\_id : INTEGER
    - first\_name: TEXT
    - last\_name: TEXT
    - age: INTEGER
    - country: TEXT
  + **Orders**:
    - order\_id: INTEGER
    - item : TEXT
    - amount: NUMERIC(10,2)
    - customer\_id : INTEGER
  + **Shippings**:
    - shipping\_id: INTEGER
    - status: TEXT
    - customer\_id: INTEGER

# Step 4: Write a user story with technical details for Data Engineers to build the data Mart.

[Link to the Story with Technical Specifications](https://docs.google.com/document/d/1nJzi3Ml7ei8riEAh2EC3Cjx6it2DJsgR21XBuQ4vsN8/edit?tab=t.0)

# Step 5: Create Data Flow Mapping

[Link to the Data Flow Mapping](https://drive.google.com/file/d/138nhSC686340IP8-6ijt8EP4SPd1l_1y/view?usp=drive_link)