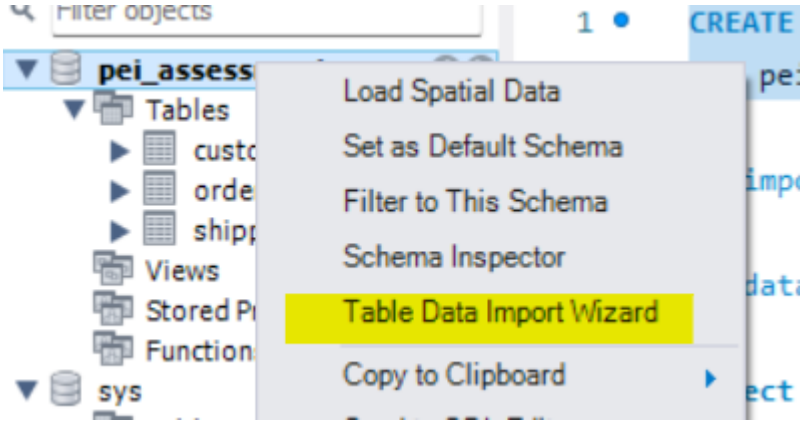
PEI Assessment

Data provided was in Json, CSV and Excel format for ease of operation. I converted all the other data formats to CSV using Python (the code for which is available here https://github.com/BharathNandan036/PEI/blob/main/Files_to_csv.ipynb) as I could easily load the csv data to a table in mysql server easily.

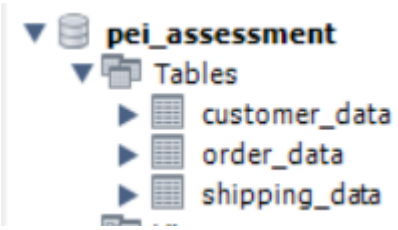
I created a database called PEI_Assessment using the below SQL:

```
CREATE DATABASE pei_assessment;
USE pei_assessment;
```

Then i imported the data (already converted to CSV) using the table import function in MySQL Workbench as shown below:



Imported all the 3 CSV files and the data base not looked like below:

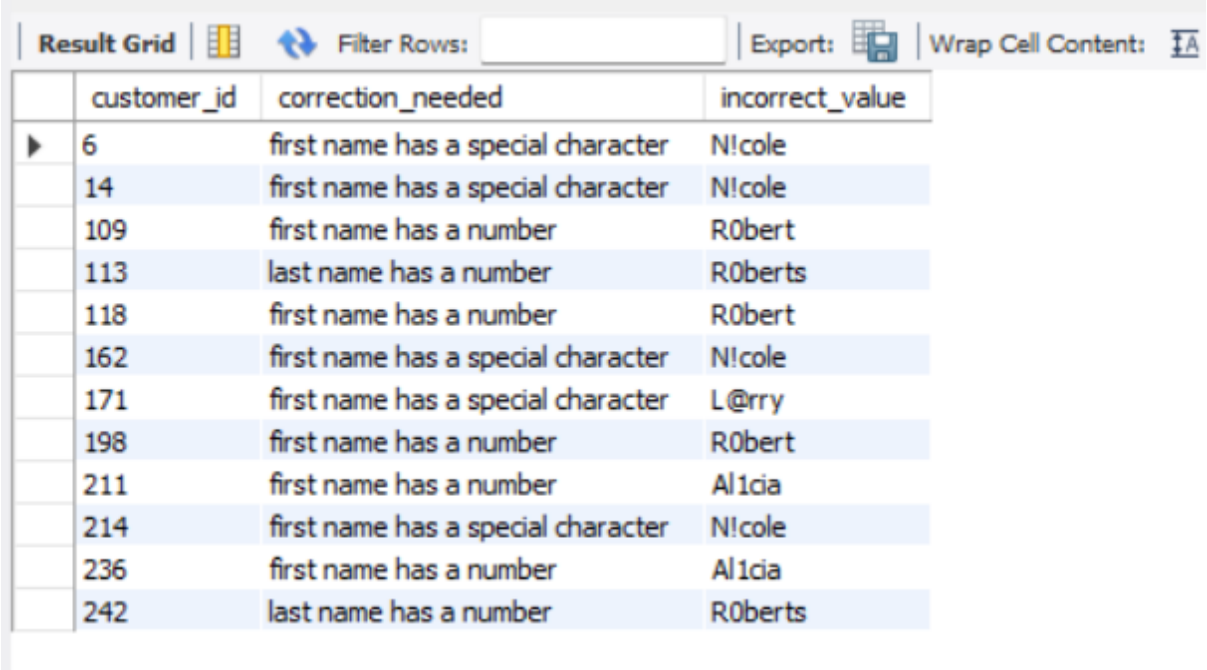


Customer Data Sanity check:

Started off with basic sanity checks and I understood that there were 250 rows of data and even the unique customer IDs were 250 this ensured that there were no duplicates.
Checked for null values in any column and also for special characters or numbers in any categoric variable columns.
I could understand that there were few rows in which either first name or last name had either a special character or a number within the data.

Code for the above is available here :
https://github.com/BharathNandan036/PEI/blob/main/Customer_data_sanity_checks.sql

Created a script to understand what is the incorrect field. This would help to debug and fix the issues.



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	correction_needed	incorrect_value
6	first name has a special character	N!cole
14	first name has a special character	N!cole
109	first name has a number	R0bert
113	last name has a number	R0berts
118	first name has a number	R0bert
162	first name has a special character	N!cole
171	first name has a special character	L@rry
198	first name has a number	R0bert
211	first name has a number	Al1cia
214	first name has a special character	N!cole
236	first name has a number	Al1cia
242	last name has a number	R0berts

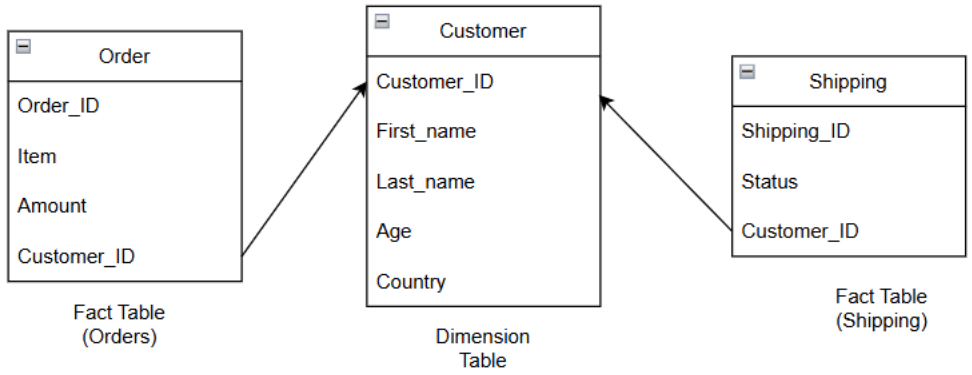
Orders data sanity checks
Checked for similar pattern of any incorrect values in any field or any record but the orders data was clean and required no changes. The scripts used to perform the checks is available here : https://github.com/BharathNandan036/PEI/blob/main/order_data_sanity_checks.sql

Shipping data sanity checks
Even Shipping data did not have any issues, the scripts used to perform the checks is available here :
https://github.com/BharathNandan036/PEI/blob/main/shipping_data_sanity_checks.sql

Context Checks
Checked for the context of customer data in every other fact table. I checked if the customers who placed the orders and customers to whom the shipments were booked were all a subset of customers from customer data. The queries used to check the context are a part of each of the files mentioned above.

I performed the entire task in python as we can incorporate these checks as a part of data ingestion and the above errors can be eliminated once for all. The python code is here :
https://github.com/BharathNandan036/PEI/blob/main/PEI_Data_Sanity_Checks.ipynb

Existing Data Model



I did not have tool in which I could build a detailed data model so i have used https://app.diagrams.net/ to draw a rough data model sketch for reference.

Anticipated Data

Customer Data (table name dim_customer)
Source Data Fields : Customer_ID , First, Last, Age, Country
Issues : First and last names has special characters and numbers in them.
Target Fields:

1. Customer_id string (numbers only) (primary key)
2. First_name string (alphabets only)
3. Last_name string (alphabets only)
4. Age integer
5. Country string (upper case only)

Order Data (table name fact_orders)
Source Data Fields : Order_ID , Item, Amount , Customer_ID
Target Fields :

1. Order_ID string (numbers only) (primary key)
2. Item_id string (numbers only) (primary key)
3. Amount float (decimal numbers only)
4. Customer_id string (numbers only) (foreign key)

Item Data (table name dim_item)
Source Data fields : Item (from orders table)
Target Fields :

1. Item_ID string (numbers only) (primary key) new column to be created
2. Item_name string

Item Data is a new table added to optimise the data model and based on the reporting requirements, eg : we have 250 orders and item names repeated multiple times and the same names is saved multiple times if we keep item names in the order table then the data is not normalised. Hence we have to create a Item table name store all the attributes of the item like item name and different things related to it, additionally if any item gets renamed

example Headset as Head Set then we can easily handle this situation by changing it in the dimension table else the reporting team will face issues when trying to query the orders pertaining to that product and might have to write multiple where clauses based on how many ever times the names was changed else that part of the data dosenot get filtered. .
Instead if we have it at the item table the team can use the item_id in the where clause and get the accurate information easily and efficiently.
Similarly in the data set in the list of items ordered we can find "Harddisk" and this is a type and if some one changed this to "Hard Disk" then all the reports have to corrected accordingly.

Shipping Data (table name fact_shipping)
Source Data fields : Shipping_ID , status , customer_id
Target fields :

1. Shipping_id string (numbers only) (primary key)
2. Status string ENUM (Delivered, Pending)
3. Customer_id string (numbers only) (foreign key)

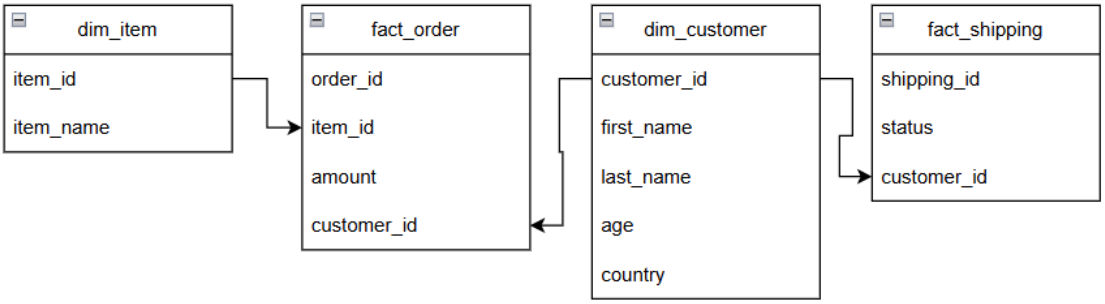
Here since in the data we only have 2 status pending and delivered we can retain but it is advised to use ENUM data type and have only values "Delivered" and "pending" so that any other typos can be avoided.

Other Recommendations:
It is better if we have order_id in the shipping table instead of customer_id because a shipping is always specific to a order and a given customer can have many orders. If we have customer_id in the shipping table then we cannot know which order is delivered and which is pending. Not suggested this in the anticipated data model because I am not sure if we have order_id level shipping information available in the source.

If we have any source for order date then we have to partition by order date so that querying becomes faster.

We have to include the record ingestion timestamps (record insert time stamp and record valid time stamp) for all the dimension tables so that we can understand the record validity and any of the updates.

Anticipated Data Model



Proposed Domain Model

Customer > Order : 1customer can order multiple times hence 1 to Many relationship.
Order > Item : There can be only 1 item per order but a item can be ordered many times hence many to 1 relationship.
Order > Shipping : 1 Order can have only 1 shipping record hence 1 to 1 relationship.

Reporting Requirements in the proposed model

1. Total Amount spent for all countries for pending deliveries
    1. Join Customer > Shipping > Order
    2. Retain only status = ‘pending’
    3. Group by country and status.
    4. Aggregation sum(amount)

2. Customer level transactions, quantity, amount and product details
    1. Join Customer > Order > Item
    2. Group by Customer
    3. Aggregation count(order_id) , count(item) , sum(amount) and optional to to add item names group_concat(item , ‘,’)

3. Top products by country
    1. Join Customer > Order > Item
    2. Group by country , item_name
    3. Aggregation count(item)
    4. Rank the values in CTE and then select the one with 1st rank

4. Top products purchased by age < 30 and age > 30
    1. Categorise age from customer
    2. Join Customer > Order > Item
    3. Group by age_category, item_name
    4. Aggregation count(item)
    5. Rank the values in CTE and then select the one with 1st rank

5. Country with least orders and sales
    1. Join Customer > Order
    2. Aggregation count(order_id) , sum(amount)
    3. Retain only the least value by sorting and limiting