Written Response: Data Engineering Challenge – KI performance – Data Modelling Task

# Evaluate the quality of the data.

## Customer Profile Data Set

### Statistics:

Total Missing Cells: 540

% Missing Cells: 7.7%

Duplicate rows: 0

Notes: With only 7.7% of the cells missing and zero duplicate rows, the dataset is quite complete.

### Variables:

**Id**

100% distinct values

0 missing values

Values range from 1 – 1000

Notes: This is what we should expect from an ID column. There are no issues here.

**First\_name**

940 distinct values

0 missing values

Notes: There is a large variance in the names, with some duplicates. No issues here.

**Last\_name**

989 distinct values

0 missing values

Notes: No issues

**Email**

907 values

100% of values are distinct

93 missing values

Notes: We should have confidence in the 907 values that are present, given that they are all distinct. It would be worrying if there were duplicate emails across varying accounts.

**Gender**

8 distinct values

447 missing values

44.7% missing values

Notes: The number of missing values is potentially a considerable issue. We should be worried moving forward about making any sweeping generalizations since we do not have comprehensive coverage or understanding of the gender of the members in the data.

**Country**

125 distinct values

0 missing values

The most prominent countries are China, Indonesia, and Russia. The following word cloud gives us a good idea of what countries are most common:

A close up of words

Description automatically generated

**Address**

1000 distinct values and none are missing

Notes: There are no worries with regard to this column

A screenshot of a graph

Description automatically generated

This correlations heat map suggests that there is no correlation between gender and id.

## Product Profile Data Set

### Statistics:

Total Missing Cells: 1969 / 10000

% Missing Cells: 19.7%

Duplicate rows: 0

Notes: We see that nearly 1/5 of the cells are missing values, which is worrisome. We would be wise to dig into the variables to see where these missing values occur and what to look out for as we proceed with future processing.

### Variables:

**Id**

100% distinct values

0 missing values

Values range from 1 – 1000

Notes: This is what we should expect from an ID column. There are no issues here.

**name**

862 distinct values

88.3% distinct value percentage

24 missing values

Notes: The main takeaway here is that we should take note that there are products that have the same name. We need to us id to identify them, and id + name to differentiate between products with the same name.

**description**

676 distinct values

73.9% distinction

85 missing values

8.5% missing

Notes: It is hard to draw too much from this. It seems odd that products would have the same description, but we would have to identify individual cases to start to draw any major conclusions. Perhaps it’s something to be aware of.

**brand**

There are 10 distinct brands

17 records are missing a brand

Notes: This tells us that we are looking at products from 10 different companies. Almost all records state what brand the product belongs to.

**color**

397/1000 missing values == 39.7% missing

Notes: This is a large number of missing values; however, it doesn’t seem like a relatively less important column.

**barcode**

6/1000 == 0.6% are non-distinct

0 missing values

Notes: It seems strange that there are a few instances where the bar codes are identical. This might be something to specifically check manually.

**origin**

119 distinct values

13.1% distinction

90 missing == 9%

Notes: China is the main value listed here, which is not surprising because of their manufacturing prowess. There is a pretty high coverage rate at 91%.

**Material**

11 distinct

718/1000 == 71.8% missing

Notes: Mainly missing values here, so we should be cautious to work too heavily with this column when trying to make generalizations.

**Size**

605 / 1000 == 60.5% missing values

7 distinct sizes: 3XL, 2XL, XL, L, M, S, XS

A colorful rectangular boxes with text

Description automatically generated with medium confidence

**Price**

33 / 1000 == 3.3% missing values

Mean price == 17.58

Min 5

Max 29.97

Notes: Quite an even distribution of prices according to the histogram. Relatively small range and similarly low price point.

## Transactions Profile Data Set

### Statistics:

7 Variables: 5 numeric, 1 text, 1 DateTime

359 / 7000 == 5.1% missing cells

Notes:

* Customer\_id has 3.4% missing values (this is odd for transactions)
* Quantity has 1.3% missing values
* Comments has 31.2% missing values

### Variables:

**Id**

* 100% distinct values
* Ranges from 1 – 1000

**Customer\_id**

* 62.4% distinct values
* 34 missing
* Min == 2
* Max == 999
* Float, not int

Notes: There are some customers in the database with no transactions. Missing customer\_id’s are a bit odd. Customer\_id is a float, not an int value, so we should be careful to convert this either when reading, or as a pre-processing step.

**Datetime**

* 0 missing values 👍🏼

**Product\_id**

* 0 missing values
* 636 distinct values

**Quantity**

* Ranges from 1 – 10
* 13 missing values

**Total\_price**

* 0 missing values

Notes: We should be able to determine any missing quantities from the total price, given that we have 0 missing values.

**Comments**

* 351 distinct
* 312 missing

Notes: Not much to say here. There could be some interesting qualitative analysis.

**General thoughts**

1. **Customer Data Profile**:

* The email column has some missing values, indicating potential data quality issues.
* The gender column has many missing values, suggesting it could be difficult to draw meaningful conclusions from this field unless the missing values are handled.

1. **Product Data Profile**:

* The material and size fields have significant amounts of missing data, which could impact analyses relying on these attributes.
* The distribution of the price column and missing values can help in determining appropriate preprocessing or imputation strategies.

1. **Transactions Data Profile**:

* The customer\_id field has some missing values, indicating that not all transactions are linked to a known customer.
* The data includes a datetime field, suggesting it can be used for temporal analyses of transaction patterns over time.
* The presence of comments may offer some unstructured data insights if relevant analyses are pursued.

## Design the dimension model of the fact table and the dimension tables for storing the data in the data warehouse.

**Fact table: FactTransactions**

Primary Key:

* id - this serves as a unique identifier for each transaction

Foreign Keys:

* customer\_id – this serves to identify the customer making the purchase, and associates with the customer dimension table.
* product\_id – this serves to identify the products the customer is purchasing, and associates with the product dimension table.

Measures (these are generally self-explanatory and can be used for analysis):

* Quantity
* Total\_price
* Transaction\_date

**Dimension table: DimCustomer**

**Primary Key:** customer\_id

**Attributes:**

* first\_name
* last\_name
* email
* gender
* country
* address

**Dimension table: DimProduct**

**Primary Key:** product\_id

**Attributes:**

* name
* description
* brand
* color
* barcode
* origin
* material
* size
* price

## How are you going to partition your tables, and why?

**FactTransactions**

* Partition column: transaction\_date
* Rationale: Filtering and aggregation often happens over a date range, so partitioning on transaction\_date allows queries to trim partitions which will quickly reduce the total amount of data scanned.

**DimCustomer**

* Partition column: None
* Rationale: None of these columns make sense to partition on. We know from analysis that the gender attribute has many missing values, so it is a poor choice. Country is heavily skewed, so it would create imbalanced partitions. Customer\_id has high cardinality, so it would create too many partitions. We are left with first\_name, last\_name, email and address, none of which make sense to partition on for similar reasons.
* If I had to choose: If I had to choose a column to partition on, I would probably choose country since filtering is likely to happen over certain regions.

**DimProduct**

* Partition column: brand
* Rationale: The cardinality of brand (10) is a good number, the number of missing values is low, and the distribution is fairly uniform. There is a slight skew towards brand 3 with a 16.4% frequency, but all other brands range from 8.2 – 10.2% frequency.

## Write the SQL queries to answer the following questions:

1. How many products have been sold with profits?  
   **SQL Query:** SELECT COUNT(DISTINCT product\_id) AS num\_products\_sold\_with\_profits  
    FROM transactions  
    WHERE total\_price > 0;
2. Calculate the total amount of profits?  
   **SQL Query:**  
    SELECT SUM(total\_price) AS total\_profits  
    FROM transactions  
    WHERE total\_price > 0;
3. Which brand is performing the best?  
   **SQL Query:**  
    SELECT brand, SUM(total\_price) AS top\_brand  
    FROM transactions t  
    INNER JOIN product p  
    ON t.product\_id = p.id  
    GROUP BY brand  
    ORDER BY total\_profit DESC  
    LIMIT 1;
4. Sort customers by order of importance  
   **SQL Query:** SELECT t.customer\_id, SUM(t.total\_price) AS total\_value  
    FROM transactions t  
    INNER JOIN customer c  
    ON t.customer\_id = c.id  
    GROUP BY t.customer\_id  
    ORDER BY total\_value DESC;

#### Which product do men buy the most?

**SQL Query:** WITH JoinedTransProd AS (  
 SELECT t.product\_id, t.customer\_id  
 FROM transactions t  
 INNER JOIN product p ON t.product\_id = p.id  
 ),  
 MaleCustomers AS (  
 SELECT id AS male\_customer\_id  
 FROM customer  
 WHERE gender = 'Male'  
 ),  
 Joined AS (  
 SELECT jtp.product\_id  
 FROM JoinedTransProd jtp  
 INNER JOIN MaleCustomers mc ON jtp.customer\_id = mc.male\_customer\_id  
 ),  
 ProductCount AS (  
 SELECT product\_id, COUNT(\*) AS purchase\_count  
 FROM Joined  
 GROUP BY product\_id  
 ),  
 MaxPurchaseCount AS (  
 SELECT MAX(purchase\_count) AS max\_count  
 FROM ProductCount  
 )  
 SELECT pc.product\_id, pc.purchase\_count  
 FROM ProductCount pc  
 INNER JOIN MaxPurchaseCount mpc ON pc.purchase\_count = mpc.max\_count;

#### Is there a product that both men and women love?

The following query combines the logic from 5 with the with females to find their intersection.  
  
WITH JoinedTransProd AS (  
 SELECT t.product\_id, t.customer\_id  
 FROM transactions t  
 INNER JOIN product p ON t.product\_id = p.id  
),  
MaleCustomers AS (  
 SELECT id AS male\_customer\_id  
 FROM customer  
 WHERE gender = 'Male'  
),  
FemaleCustomers AS (  
 SELECT id AS female\_customer\_id  
 FROM customer  
 WHERE gender = 'Female'  
),  
MaleJoined AS (  
 SELECT jtp.product\_id  
 FROM JoinedTransProd jtp  
 INNER JOIN MaleCustomers mc ON jtp.customer\_id = mc.male\_customer\_id  
),  
FemaleJoined AS (  
 SELECT jtp.product\_id  
 FROM JoinedTransProd jtp  
 INNER JOIN FemaleCustomers fc ON jtp.customer\_id = fc.female\_customer\_id  
),  
MaleProductCount AS (  
 SELECT product\_id, COUNT(\*) AS male\_purchase\_count  
 FROM MaleJoined  
 GROUP BY product\_id  
),  
FemaleProductCount AS (  
 SELECT product\_id, COUNT(\*) AS female\_purchase\_count  
 FROM FemaleJoined  
 GROUP BY product\_id  
),  
MaxMalePurchaseCount AS (  
 SELECT MAX(male\_purchase\_count) AS max\_male\_count  
 FROM MaleProductCount  
),  
MaxFemalePurchaseCount AS (  
 SELECT MAX(female\_purchase\_count) AS max\_female\_count  
 FROM FemaleProductCount  
),  
TopMaleProducts AS (  
 SELECT mpc.product\_id, mpc.male\_purchase\_count  
 FROM MaleProductCount mpc  
 INNER JOIN MaxMalePurchaseCount mmc ON mpc.male\_purchase\_count = mmc.max\_male\_count  
),  
TopFemaleProducts AS (  
 SELECT fpc.product\_id, fpc.female\_purchase\_count  
 FROM FemaleProductCount fpc  
 INNER JOIN MaxFemalePurchaseCount mfc ON fpc.female\_purchase\_count = mfc.max\_female\_count  
)  
SELECT tmp.product\_id, tmp.male\_purchase\_count, tfp.female\_purchase\_count  
FROM TopMaleProducts tmp  
INNER JOIN TopFemaleProducts tfp ON tmp.product\_id = tfp.product\_id;