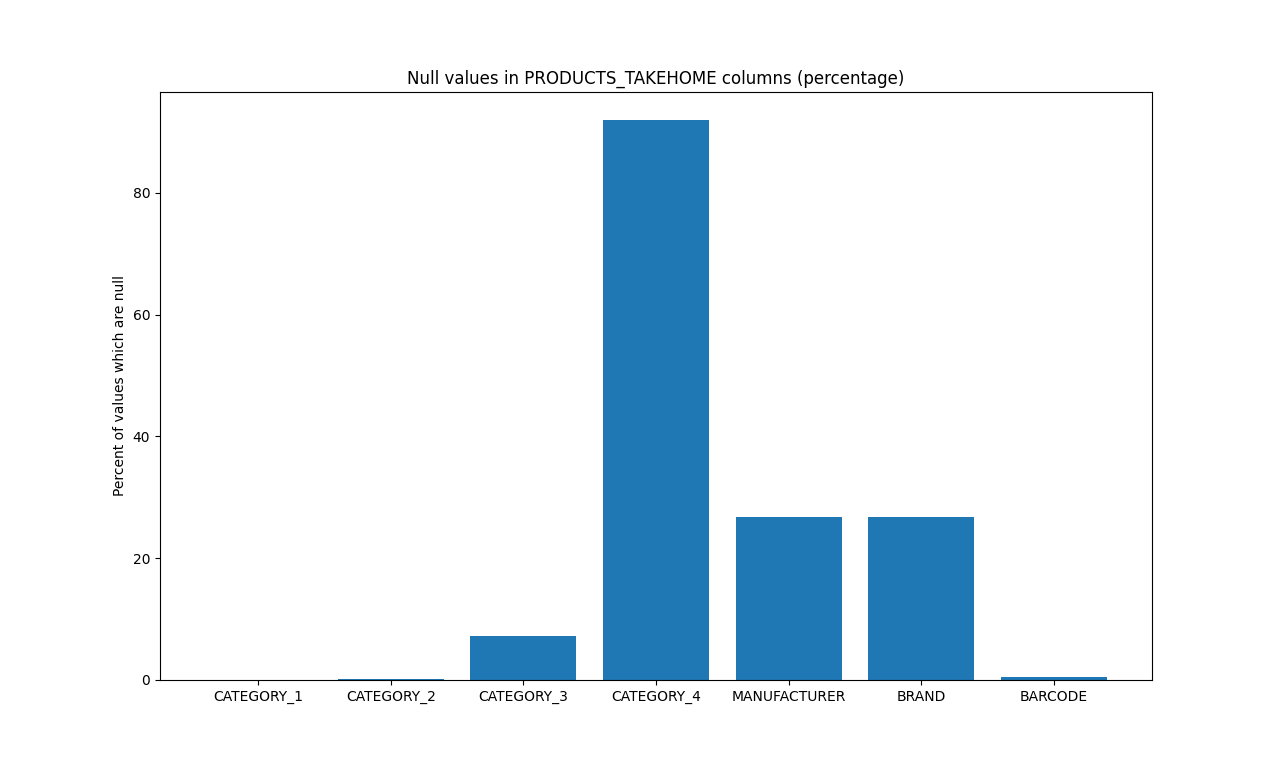
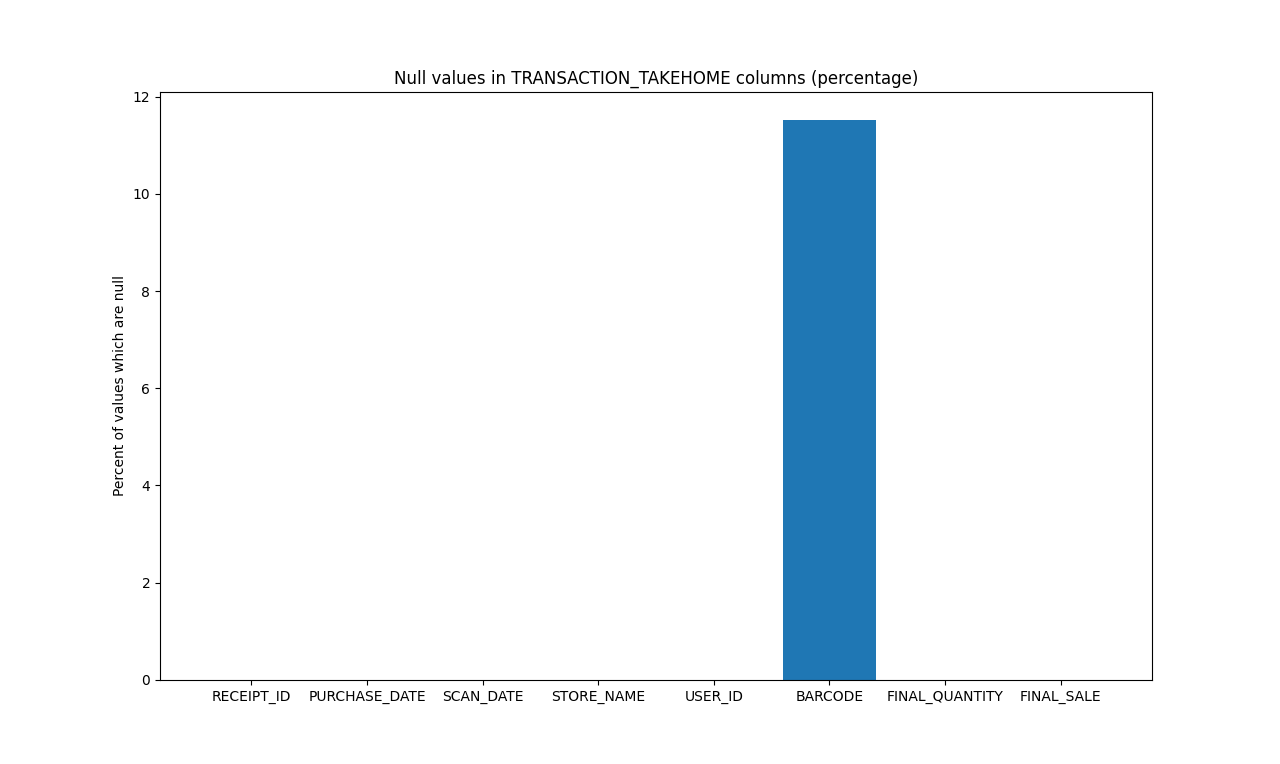
**Are there any data quality issues present?**

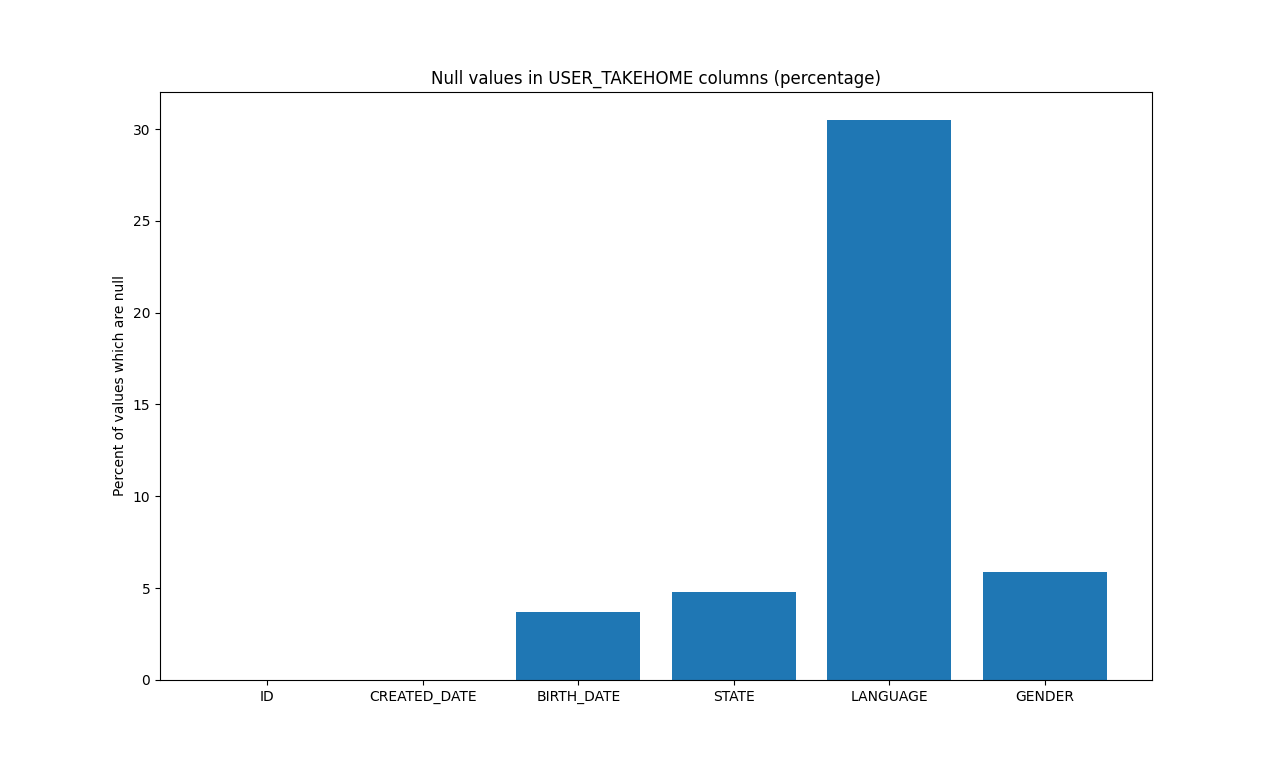
The products file contains a large number of missing values. Out of the 845,552 records in the products file, 226,472 (approximately 27%) are missing values for both the manufacturer and brand, and another two are missing values for the brand but not the manufacturer. Additionally, 4,025 records are missing barcode data; this is especially concerning because the barcode field is used to match records in the products file to records in the transaction file. While many products are missing some or all category fields, this is less concerning because these fields appear to be optional.



Additionally, there is one other consideration which is not a data quality issue per se, but merits consideration from a data analysis standpoint. Many of the manufacturers and brands listed in the products file contain quotation marks or apostrophes, and others contain accented characters. When importing this data into a SQL database, care must be taken to ensure that a proper character set is used and that fields with quotation marks are handled correctly.



The transactions file contains a number of data typing issues. Out of the 50,000 records in the transaction table, 12,500 contain a string value of ‘zero’ in the quantity field instead of a numeric value, and 12,500 contain whitespace characters in the sale field instead of a numeric value. These two sets of 12,500 values were completely distinct. Due to the varying data types, SQL would not import the quantity and sale columns as numeric data. To work around this, I wrote and executed the Python script CleanTransactionsData.py to generate a version of the transactions file which was usable in my SQL database. After running some SQL queries which revealed that every unique receipt ID in the transaction file appeared an even number of times, I discovered that the records containing values of ‘zero’ or whitespace are all incomplete duplicate copies of complete records. Additionally, there are 5,762 records missing barcodes, making it impossible to connect these records to the products data. Finally, I discovered one unrealistically long store name which appears to be an error; I discovered it by attempting to import it to a SQL table with a 45-character limit on the store name column.



In the users file, the birth date field is formatted as a timestamp instead of a date. Several records are missing birth dates, states, languages, and genders, but this may be due to users declining to provide some or all of this information. The data for the user ID and the creation date (which exists as a timestamp) are present for all records. In particular, since all records have a user ID, all user records can be linked to transactions as appropriate. There are also some user birth dates, such as 1/1/1901, which are likely false. Given that these birthdays are likely user-submitted data, these users probably chose to submit a false birth date.

When comparing files with each other, I discovered that the transaction and user files seem to be incomplete. This becomes immediately apparent when joining the transaction and user data sets and indexing on the associated user ID. Out of the 50,000 records in the transaction file, only 262 contain user IDs that also exist in the user ID file; this becomes 131 after accounting for the extra, incomplete copies of transaction records mentioned previously. This overlap is large enough to test that the queries which answer the questions in section 2 are syntactically correct, but the results produced by the queries may change when more complete data sets are used.

**Are there any fields that are challenging to understand?**

In the Products file, the relationship between the fields Category\_1, Category\_2, Category\_3, and Category\_4 is not immediately obvious. After some examination of the code in Python (included in ProductsExploration.py), I believe that the values in Category\_2 represent subcategories of the values in Category\_1; a similar relationship holds for Category\_3 with Category\_2, and Category\_4 with Category\_3. In particular, if a value exists in a record for Category\_X, where X = 2, 3, or 4, there is also a value present for any field Category\_Y, where Y < X. The reverse does not necessarily hold, because not all categories in Category\_1, Category\_2, or Category\_3 have subcategories.

|  |  |
| --- | --- |

The plot on the left illustrates the percentage of values in each column of the products file which are null. The plot on the right illustrates the total number of null values in each column on a logarithmic scale. The four columns on the left represent the four category column; the steady increase in these columns is consistent with (approximately) exponential growth. Based on this, I suspect that the category columns are meant to represent a tree of categories and sub-categories, and there are four columns to accommodate the maximum depth of this tree.