**Data Dictionary (Seller and Review Data)**

**Overview**

Sellers

Description: A table containing information on each seller’s location. For normalization reasons, we only keep the zip code prefixes. City and state data are kept in a separate table.

seller\_id: A 32-character string uniquely identifying each seller. (PK) (str)

seller\_zip\_code\_prefix: The first 5 digits of the zip code where the seller is located, in string format. (str)

Reviews

Description: A table containing information on reviews. As is typical of online marketplaces, reviews tend to be ascribed to the service and delivery rather than the product, which does shape the way we treat reviews. In essence, we should not treat reviews as a review of the product.

review\_id: A 32-character string identifying each review. (str)

order\_id: A 32-character string uniquely identifying each order. (PK) (str)

review\_score: Score assigned for a given review. (Int)

review\_comment\_title: Title of review. (str)

review\_comment\_message: Body of review. (str)

review\_creation\_date: Date when review was created. (Assumed) (datetime)

review\_answer\_timestamp: Timestamp when review was posted. Presumably there is an approval process that accounts for the delay between creation and posting. (Assumed, this always follows review\_creation\_date) (datetime)

**Data Wrangling**

Sellers

This table is derived from the olist\_sellers\_dataset.csv as provided by Generation. For this table, we fixed the seller\_zip\_code\_prefix column so that it contains strings of five digits. Originally, the column contained numeric zip code prefixes that omitted leading zeros. We drop the seller\_state and seller\_city columns as they fully depend on seller\_zip\_code\_prefix and are redundant. The relevant data is stored separately in another table. Sellers are uniquely identified by seller\_id, and this serves as our primary key.

Reviews

This table is derived from the olist\_order\_reviews\_dataset.csv as provided by Generation. The challenge here was to identify duplicates in review\_id and order\_id, and the reason they are duplicated.

For review\_id, each individual review can be written for a single or multiple order\_id. Each order\_id may contain a single item, or any combination of items from any combination of sellers. In the case where a single review\_id is assigned to multiple order\_id, the orders were submitted simultaneously. Speculatively, an order\_id is assigned to each instance of delivery. Products that are bundled together and delivered at the same time are assigned the same order\_id. Typically, identical products from one seller fall under this category, but any combination of products and sellers can also be this.

We decide not to drop any duplicate review\_id arising from this, as there are differences in the underlying data. Furthermore, as reviews are often for the quality of service and delivery rather than the product, there is no need for each individual review to be assigned to a single product.

For order\_id, we discover that some review\_id have duplicate order\_id, with differing timestamps and sometimes, content. In short, there appear to be two reviews written at different times for a given order. In such cases, we opt to keep the order\_id with the later timestamp, under the assumption that the later review is the updated review and the earlier one is redundant. This leaves us with order\_id as the primary key for the reviews table, as it uniquely identifies each row.