Products & Reviews documentation

The purpose of this document is to explain the contents of the product and review tables created, and the logic behind it.

A main consideration behind creating this data model was to reduce DWH load by separating data according to use cases: for example, users needing the product images are unlikely to need the product related items and vice-versa. For this reason, the data was split to a very granular level for fine-grain access, which in some ways also helps reduce memory usage by storing the minimal amount of data possible, instead of repeating information in rows.

Another consideration was the NULL handling: here exists a tradeoff between usability and memory usage that ought to be enforced at the organizational level. If NULLs are left as blank in the DWH, but members of the organization know how to work around it, that's the perfect situation. Otherwise, I prefer to write "Unknown" for strings and "-1" for numbers instead of NULL in most tables so that it becomes obvious quickly to any analysts and scientists that this data is missing or wrong or should not be used. The reason I prefer this over leaving NULLs is because in some query engines, the presence of NULLs can affect the way aggregate functions run, which can be confusing and problematic (I've recently had this experience at my current workplace where we changed query engines and that affected our query logic heavily due to differences in NULL handling).

Facts.Reviews

This table focuses on giving the reviews data as is, and there are minimal transformations done on it:

- The date is parsed from the strange string mm dd, yyyy format into a neat int
- The count of people who voted the comment as “helpful” is parsed from the list
- The count of people who voted the comment as “not helpful” is also parsed
- A unique review_id is created for ease of data management

The table ends up containing the following columns:

Column name	type	description
Review_id	Str	Unique identifier of the particular review (primary key)
Reviewer_id	Str	Unique identifier of the reviewer (foreign key)
Item_id	Str	Unique identifier of the item (foreign key)
Review_text	Str	Text of the review, as written by the reviewer
Review_summary	Str	Review summary, as written by the reviewer
Rating	Float	Rating given by the reviewer to this item
Count_review_helpful_yes	Int	Number of votes for “review helpful”
Count_review_helpful_no	Int	Number of votes for “review unhelpful”
Review_unix_time	Bigint	Timestamp of the review in UNIX, based on seconds
Review_date	Str	Date of the review, given as string

Dimensions.Reviewers

This table aims to store each reviewer id once, for easy joining to other columns. I initially thought every reviewer ID would have a one-to-one relationship with the reviewer username, but the usernames can change, which is why this table is just one column, to be used as a link, that manages to keep the primary key (reviewer id) unique

The table ends up containing the following columns:

Column name	type	description
Reviewer_id	Str	Unique identifier of the particular reviewer (primary key)

Dimensions.Reviewers_user_names

This table stores the (multiple) user names associated with each reviewer id. The columns are as follows:

Column name	type	description
Reviewer_id	Str	Unique identifier of the particular reviewer (primary key, multiple)
Reviewer_user_name	Str	User name of the reviewer

This table could also benefit from adding dates of user name changes

Dimensions.Date_dimension

This table stores the date information for ease of joining nice date elements. Normally we'd add the day name, week name, month name, relative weeks, etc., basically anything that anyone could need to represent the date in any dashboard or analysis. In this case the example is minimal and contains the following column:

Column name	type	description
date_as_int	int	Date as int, represented as YYYYMMDD (PK)

The reason for choosing "int" as the data type is that it's less memory intensive than "str".

Dimensions.Products

This is the main table where we store product information and base characteristics; this table also serves as the hub for joining additional information that is only required in edge cases. This table should contain all the products, whereas many of the "subordinates" will miss item ids for which there is no data. The columns in this table are:

Column name	type	description
Item_id	Str	Unique identifier of the item/object (primary key)
Title	Str	Title of the object
Brand	Str	Brand of the object
Description	Str	Description of the item
Price	Float	Price of the item
Currency	Str	Currency in which the item's price is quoted

Dimensions.Product_images

This table stores the URLs to the product images, and is a separate table because this data will only be required in a handful of very specific cases. The columns contained are:

Column name	type	description
Item_id	Str	Unique identifier of the item/object (primary key)
Image_url	Str	URL of the item image

Dimensions.Product_sales_ranking

This table stores information on the sales ranking of products. This is also data that will only be required in specific use cases, hence its own table. The columns contained are:

Column name	type	description
Item_id	Str	Unique identifier of the item/object (primary key)
Category_ranked	Str	The category in which the item is ranked
Ranking	Int	The ranking of the product in the given category

Dimensions.Product_categories

This table stores info on the categories to which items belong. Since each item belongs to multiple categories, it will show up multiple times, with each category on a new row. This table will make it easy to join each item to all its categories. The columns contained are:

Column name	type	description
Item_id	Str	Unique identifier of the item/object (primary key)
Category	Str	The category to which the item belongs

Dimensions.Product_bought_together

This table stores info on the items bought together with the focal one. Since each item was bought together with multiple others, it will show up multiple times, with each one on a new row. This table will make it easy to join each item to all its related items, which will come in very handy when trying to map networks, find related items, etc., which is worth the price to pay for storing this much data in one table (in my opinion). The columns contained are:

Column name	type	description
Item_id	Str	Unique identifier of the item/object (primary key)
Bought_together_with_item_id	Str	The id of the item together with which the focal item was bought

Dimensions.Product_also_viewed

This table stores info on the items also viewed in the same session as the focal one. Since each item was viewed together with multiple others, it will show up multiple times, with each one on a new row. This table will make it easy to join each item to all its related items, which will come in very handy when trying to map networks, find related items, etc., which is worth the price to pay for storing this much data in one table (in my opinion). The columns contained are:

Column name	type	description
Item_id	Str	Unique identifier of the item/object (primary key)
Also_viewed_item_id	Str	The id of the item together with which the focal item was bought