Université de Rennes 1

EDD

Data Warehouses

Open Food Facts Project

Authors
D. De Paola
L. Mandruzzato

Supervisor
Prof. Marc Bousse

Abstract. This paper describes the design process of a multidimensional model for analyzing data coming from the Open Food Facts platform. It aims to find an efficient solution to support Online Analytical Processes (OLAP). We can divide the followed approach into three main steps. Firstly, a design phase (sections 1 and 2) — i.e., UML modeling [1] and ER schema design [2]. Secondly, an implementation stage (sections 2 and 3) — i.e., SQL code to physically implement the ER schema [4] and Kettle transformations to populate it [5][6]. Finally, a querying process followed by some analysis (sections 4 and 5) — i.e., MDX queries design [7] and discussion of the results obtained.

Keywords: Business Intelligence · Kettle · MDX · Mondrian · Multidimensional Models · Pentaho · R-OLAP

October 2021

1 Design choices for the Mondrian Schema

1.1 Assignment and research directions

The Open Food Facts integration base structures the data about several products in three tables. One contains the users' pseudo, the second is reserved for the product categories — i.e., pnns1 and pnns2 — and the last with the different details for each single product update.

Leveraging data that has been given to us and the two MDX queries present in the assignment, we derived three main research directions.

1. The former research scope was to retrieve the distribution of products on the platform. In this way, we can make a fast analysis of the product coverage on several categories.
2. The second direction for information extraction was to find the update frequency for every single product, highlighting the possible presence of specific non-updated elements in the dataset and the most active typologies of contributors.
3. A third interesting research direction has been adopted to explore the nutritional grading on different categories of products. Through it, it is possible to discover healthier clusters of products and suggest to the industry a need for nutritional quality improvement in the related area.

1.2 Measures

To support the two queries given in the assignment, the following measures have been derived:

- **Number of products:** the number of different products, identified by their barcode.
- **Number of versions:** the number of different versions, identified by their barcode and last modified field — i.e., a value derived by the date modification field.

Three additional measures have been defined to support the research directions outlined in section 1.1, specifically for retrieving some useful information on the nutrition score and the number of updates:

- **Average Score:** Average grade of nutrition scores.
- **Number of creations:** The number of new products added to the platform. It is defined by counting the number of different barcodes.
- **Number of updates:** Count of different updates. Defined as $Number\ of\ versions - Number\ of\ creations$

1.3 Dimensions

By following the inspection objectives defined in section 1.1, the following dimensions were used to support our measures:

- **Dimension Date Creation and Modification** contains a single hierarchy with the version day, month, and year of creation or modification. The dimension scheme will contain all distinct and non-duplicated dates present in the database.
- **Dimension Category** contains a single hierarchy defined by four levels: version, product, PNNS2, PNNS1. The dimension scheme will contain all the information related to the product, its general category, and version.
- **Dimension Nutritional Info (Degraded)** contains a single hierarchy defined by four levels: version, score, class, presence. The levels respectively describe the version of the nutritional score, the nutrition score associated with a product and a version, the nutrition score class — i.e., A, B, C, D, E — and the presence of the nutrition score in the integration base.
- **Dimension Contributors (Degraded)** contains a single hierarchy defined by two levels: contributor and type. The table contains the information about contributors and their typology — i.e., corporate or individual.

The dimension usage mapping for the different measures resulted in the table presented in figure 1. From the scheme, it is possible to identify the distinction between two different cubes: the items cube (blue) and the versions cube (orange).

Comments Some important comments should be made on the contributors' dimension. Using the official database documentation [8] and by using some SQL test queries, it was possible to understand that, for each tuple in the versions table, only the original creator pseudo is provided. Since the contributor field does not change between different versions of a single product, it makes no sense to add this dimension in a cube that keeps into account the versions. Not only it is meaningless, but it can also bias the analysis. For instance, considering a brand new user A that adds a new product to the DB, and the latter is updated 20 times by other profiles, it would appear that user A is very active even though he has only created one single product. For this reason, as we will explain in section 1.4, we designed a cube that contains only the last updated versions for each barcode. In this way, it would appear that contributor A has interacted with the platform only once, which is correct.

	N. Products	N. Versions	Avg Score	N. Creations	N. Updates
Dim. Date	X	X	X	X	X
Dim. Category	X	X	X	X	X
Dim. Nutritional Info	X	X	X	X	X
Dim. Contributors	X		X		

Fig. 1. *Mapping of Dimension usage on Measures*

1.4 Cubes

Items Cube The Items cube contains all the information required to support the analysis on the homogeneous updated distribution of products and the different average nutrition score on different items categories, following the first and third inspection objective defined in section 1.1.

Version Cube The Versions cube will instead support the analysis on the frequency of updates and the identity of the most important contributors for different categories — i.e., the second direction of section 1.1.

In the final XML, the nutritional value presence (present or NULL), the nutritional category (A, B, C, D, E), and the contributor typology (corporate or individual) have been managed through key expressions.

2 Relational Model

The relational schema adopted for our Data Mart is made of two dimensional tables and two facts tables: OFF_dim_date, OFF_dim_category, OFF_items_facts, and OFF_versions_facts. As clarified in section 1.3, the dimensions Nutritional Score and Contributors have been degraded into the fact tables. A nutr_score column was added to both the fact tables, while the pseudo field (contributors' identifier) was only included in the items fact table.

Different choices were made in the process of populating the database.

As concerns the two dimension tables, we can notice a slight difference between them. While for the category we used both a business key — i.e., version — and a technical key — i.e., TECH_KEY_CAT, for the date we only exploited the technical one — i.e., TECH_KEY_DATE. The introduction of the two technical keys, as in many examples seen during the lectures, simplified the populating processes and was extremely useful in the Kettle pipelines (section 3) used for loading the data.

As already anticipated in section 1.4, being the items cube used for the study of the actual distribution of products into the platform, the items fact table was populated with the foreign keys of the last updated versions — i.e., we want to have the last/current state of the product. The versions cube, instead, was populated by importing all the versions present into the integration base, referenced with their correspondent technical keys.

3 Kettle

We exploited an ETL tool delivered by Pentaho named Data Integration (Kettle) to populate the Data Mart. Through it, we managed to extract the necessary data from the integration base, transform it, and finally load it into the warehouse. We designed two different transformations, one for the dimension tables and the other for the fact tables. To be run, both of them need that the SQL script to create the structure of the DB — i.e., tables and fields — was already executed.

3.1 Dimension Tables Transformation

We know that there are two dimension tables to be populated (see section physical model). Both of them will have a correspondent pipeline within the transformation — i.e., a process that takes care to fill it. Inside each table, we can distinguish a certain number of fields and a technical primary key — i.e., a sequential number assigned from the database. It is necessary to have the latter before executing the second transformation because it represents the fact table foreign key. Thus, we can notice a specific execution order: complete the dimension tables before filling the fact ones. We can observe a structure shared by the pipelines. Firstly, a sub-part to retrieve data from the integration base. Since we prefer to query the BI as few times as possible, we used a single component to accomplish

this task for all four pipelines. After the retrieving process, for the date dimension, there is a component dedicated to filtering duplicates because we do not want different rows containing the same date. This process is fundamental for the next transformation to execute the DB lookup component to retrieve the sequential keys to add in the fact tables. In fact, after having removed duplicates, each row of the date dimension can be individuated using the primary key, but even using the set of all the other data — i.e., day, month, year. In the end, there are some steps focused on populating the already existing dimension tables with the data obtained from the stream.

3.2 Fact Tables Transformation

Even here, there is a structure shared between the two pipelines which populate the OFF_items_facts and OFF_versions_facts. It consists of an initial retrieving/creating sub-part that collects and adds all the fields necessary to obtain the technical keys. Thus, at the end of this process, we must have a stream containing all the data that distinguish all the sequential keys to fill each fact table. In a nutshell, if in a dimension table there is a row with "1" as technical sequential key and "A" and "B" as other data, that instance can be retrieved passing to the query "1" as a parameter, but even using "A" and "B". The latter querying typology is the one executed from the DB lookup component present after the first sub-part of the transformation. There is one of them for both the dimensions considered for each fact table. In the end, after having obtained the sequential keys — i.e., the foreign keys, we can proceed to put the results in the tables.

4 MDX Queries

The MDX queries design process can be ideally split into two different moments. One during the first step of the project in order to frame the problem and get an idea of the measures and dimensions we would have needed. We can see these initial queries as guidelines to follow to have more clear the problem to solve. At a later time, after having a functioning version of our multidimensional model, we focused again on MDX queries to improve them. After having worked on the model, we were more aware of its potential and of what could and could not be done with it.

4.1 Query Description

Items Cube

- **Query n.1.** This query computes the number of product descriptions per creator and year of creation on rows (assigned query).
- **Query n.2.** This query verifies that the items cube works properly. If everything is correct, this query should return no results, because there are no barcodes that appear twice in the table (no versions, only products).
- **Query n.3.** This query spots which are the most and less healthy PNNS2 categories.
- **Query n.4.** This query highlights the categories that have a lack of information related to the nutritional score.
- **Query n.5.** This query gets information about how the different typologies of contributors impact the system adding products. Which are the "most active" categories of contributors?
- **Query n.6.** This query retrieve the number of products for the different pnns1 and pnns2 on columns per years on rows.

Versions Cube

- **Query n.1.** This query computes the number of versions per month on rows and PNNS1 values on columns of all product descriptions created in 2017 (assigned query). We can have updates even later than 2017, the only constraint is that the creation date of the products in question is dated in that year.
- **Query n.2.** This query allows understanding the number of updates per each product creation for each category.

5 Results

The MDX queries defined in the previous section support the three research directions set at the beginning: the coverage of products on different categories, the frequency of updates performed on the products, and the average nutritional score for several classes.

For the first analysis scope, we considered the results of queries number 1, 5, and 6 on the items cube (section 4.1). By the resulting aggregate table retrieved on Pentaho, it is possible to verify the general distribution of products on the platform. Regarding the contributors, we can spot that the proportion of individual contributions is higher than the corporate one in almost every product category. We can expect it for a crowdsourcing platform as Open Food Facts. Moreover, it is a perfect parameter to test the heterogeneous distribution of contributors. Examining the results of the first query, we can also monitor the number of corporate contributions on the platform and discover a

high activity in 2016 and a declining trend for the following years. A new strategy to engage Corporate Companies could be suggested to the Open Food Facts project.

Regarding the frequency of updates, we took into examination the results on queries number 1 and 2 on the version cube (section 4.1). From the first query, it is possible to understand the number of different versions for the products created in 2017. The results for the second query allow monitoring the rate between creations and updates for each PNNS1 category. On average, the rate of updates is very high, meaning that each of the PNNS1 categories is frequently updated.

Finally, to retrieve some insights on the nutritional score, we studied the result deriving from queries 3 and 4 on the Items Cube (section 4.1). From the first query, it was possible to verify which categories have more 'null' nutritional scores. From the second one, it is possible to find the classes with lower average nutrition score. For the category "Non-sugared beverages" could be advisable to reduce the number of 'null' nutritional scores. A final and extremely interesting result regards the average nutrition scores. The low average on some common categories such as "breakfast cereals" and "dressing and sauces", could be a piece of extremely valuable information for a company, suggesting the creation of new healthier products for that category. In a nutshell, this analysis may suggest to a company a possible blue ocean [9] in which enter with brand new products.

References

1. UML Cubes - OFF_UML_cubes.pdf
2. ER Schema - OFF_ER_schema.pdf
3. Mondiran XML schema - OFF_mondrian_schema.xml
4. SQL script to create physical DB - OFF_SQL_starModel.xml
5. Kettle transformation for dimension tables - OFF_kettle_dim_tables.ktr
6. Kettle transformation for fact tables - OFF_kettle_facts_tables.ktr
7. MDX Queries - OFF_MDX_queries.txt
8. OFF official documentation describing the fields present in their DB - data-fields.txt
9. Blue Ocean Strategy - https://www.blueoceanstrategy.com/what-is-blue-ocean-strategy/