Data warehousing

Davide Antonino Giorgio
Politecnico di Torino
Student id: s291477
s291477@studenti.polito.it
http://giorgiodavide.it

***Abstract*—The study conducted was articulated from the phases involved in the design of a data warehouse, starting from the specifications of the problem, to a phase of analysis of the information contained in it. This last phase has been faced through some specific queries to the DW and finally through the realization of some explanatory graphs.**

I. Problem overview

It is desired to perform some analysis on a food delivery company. Starting from the specifications of the tables in the OLTP database, it is required to carry out the design of the data warehouse and to satisfy some queries of particular interest. The data analysts want to analyze efficiently the information about the *delivery*, the *average revenue for delivery* and the *average delivery time*.

II. Data Warehouse Design

The design of the data warehouse was divided into two main steps:

1) definition and drafting of a *conceptual schema* [1] based on the requirements of the problem
2) realization and modeling of a *logical schema* [1] that also aims to evaluate aspects related to the performance of execution of the queries of greatest interest

A. Conceptual schema

Given the requirements of the food delivery company, it was decided to design the conceptual schema of the data warehouse as shown in the figure 1.
In detail, four different dimensions have been modelled:

- temporal
- geographical (for the restaurant of interest)
- payment method
- transport mode

Payment method and transport mode are two dimensions directly attached to the fact and later we will see how they will be treated for the realization of the logical schema.

The restaurant was modeled in the conceptual schema by including the information about the *category* and its *name*. Both pieces of information are linked directly to the restaurant and the name is defined as a *descriptive attribute*. The information about the location of the restaurant is also attached to it. In fact, the *city*, *province* and *region* of the restaurant is given. These attributes identify a geographic hierarchy.

A temporal hierarchy was also represented to model the time according to the problem specification. In this dimension we have: *date*, *month*, *semester* and *year*. *Holiday* is a boolean attribute that is directly attached to the *date*. The *day of the week* is another information associated to it.

Instead, the measures identified are: delivery time (*total_delivery_duration*), revenue (*total_revenue*), number of deliveries (*number_of_deliveries*).

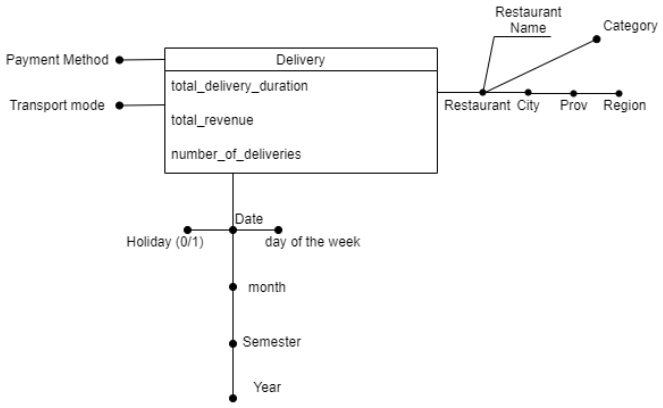

Fig. 1. Conceptual schema

B. Logical schema

The resulting schema aims to satisfy the requirements of the problem allowing however to resolve in efficient way the queries of main interest avoiding to carry out some onerous join operations. For a better comprehension, the logical schema is reported at the end of this section.

For example, it was chosen to *push down* the *transport mode* attribute within the *fact* table. Due to its size and the amount of different transport modes that exist (3), it was therefore avoided to choose solutions based on junk dimensions. This solution is given also by the fact that the queries of greater interest use the *transport mode* attribute during the analysis. It is convenient therefore to avoid the join with an eventual table and carry this information directly in the fact table. With this choice the size of the facts table will be bigger but we will be able to avoid join operations.

The *payment method*, instead, is not of particular interest for the queries that will be analyzed in the next section, also for this reason it has been avoided to push down it in the fact table. The *paymethod_id* is a surrogate key.

Regarding the temporal hierarchy, this was modeled in a traditional way leaving however the information on the *holiday (0/1)* and on the *day of the week* that are connected to the date.

A particular choice that was made when designing the *time* table is related to the fact that it was **not** decided to use the surrogate key as the primary key of this table. In this case the date itself has been chosen as the primary key, aware that this will then be present in the facts table. The advantage of this choice is linked to the fact that the *date* is an information of particular interest for the queries which will be performed by our analysts and thanks of that we can avoid Join operations between the table of `facts` and that of `time` when we have to perform a query. The disadvantages are related to the fact that: the size of the table of facts will grow because the date is internally represented as a string, also the format in which the date will be encoded will be fixed and it will not be possible to change it easily. This choice should be discussed with the client and depending on the situation, one solution or the other should be chosen. Typically choosing a surrogate key is always a good solution but in case you are often interested in particular queries, you could also opt for such a choice. Moreover, the size of the database is also a feature that should be analyzed with the client. So, in the proposed solution, the attribute of the *time_id* of the time table is a string containing the date and not a surrogate key. Thanks by this we will be able to avoid a join operation in the query *a* that will be presented in the next section. The *restaurant_id* is a surrogate key.

Finally, there is the table of *restaurants* that contains all the information for the individual restaurant, including the geographical information that interests it.

Below is reported the logical schema proposed for the data warehouse:

- **TIME**(date_id, day_of_week, holiday, month, semester, year)
- **RESTAURANT**(restaurant_id, restaurant_name, category, city, province, region)
- **PAYMETHOD**(paymethod_id, payment_method)
- **FACT**(date_id, restaurant_id, paymethod_id, transport_mode, total_delivery_duration, total_revenue, number_of_deliveries)

III. Querying the data warehouse

In this section are reported some queries of particular interest.

`Query a`: *For each day, select the total revenue and the average revenue per delivery. Sort the result by date.*

```
SELECT
  date_id,
  SUM(total_revenue)
    AS daily_total_revenue,
  SUM(total_revenue) /
  SUM(number_of_deliveries)
    AS daily_avg_revenue_delivery
FROM fact f
GROUP BY f.date_id
ORDER BY f.date_id
```

In this query we can avoid the join operation between the `fact` table and the `time` table because the information of the date is directly stored in the fact table.

`Query b`: *Select the yearly revenue and the total number of deliveries for each restaurant. Sort the results by descending yearly revenue.*

```
SELECT
  restaurant_id,
  year,
  SUM(total_revenue)
    AS yearly_revenue,
  SUM(number_of_deliveries)
    AS total_number_of_deliveries
FROM fact f, time t
WHERE f.date_id = t.date_id
GROUP BY year, restaurant_id
ORDER BY SUM(total_revenue) DESC
```

`Query c`: *Separately for each transport mode and year, select the total number of deliveries and the average time for delivery.*

```
SELECT
  transport_mode,
  year,
  SUM(number_of_deliveries)
    AS total_num_deliveries,
  SUM(total_delivery_duration) /
  SUM(number_of_deliveries)
    AS avg_time_delivery
FROM fact f, time t
WHERE f.date_id = t.date_id
GROUP BY transport_mode, year
```

`Query d`: *Consider only the deliveries with "bike" as transport mode. Separately for each month and restaurant, select the total revenue and the average delivery time.*

```
SELECT
  restaurant_id,
  month,
  SUM(total_revenue)
    AS tot_revenue,
  SUM(total_delivery_duration) /
  SUM(number_of_deliveries)
    AS avg_delivery_time
FROM fact f, time t
WHERE
  f.transport_mode = 'bike' and
  f.date_id = t.date_id
GROUP BY month, restaurant_id
```

`Query e`: *Separately for date and transport mode, select the total revenue and the maximum delivery time.*

```
SELECT
  date_id,
  transport_mode,
  SUM(total_revenue)
    AS tot_revenue,
  MAX(total_delivery_duration)
    AS max_delivery_time
FROM fact
GROUP BY date_id, transport_mode
```

`Query f`: *Separately for each month, select the total revenue and the average daily revenue.* `Query f`

```
SELECT
  month,
```

```
  year,
  SUM(total_revenue)
    AS tot_revenue,
  SUM(total_revenue) /
  COUNT(DISTINCT t.date_id)
    AS avg_daily_revenue
FROM fact f, time t
WHERE f.date_id = t.date_id
GROUP BY month, year
```

The request in the text of query f is, however, a bit ambiguous. It is not in fact reported if the average of the total revenue must be calculated for all the days in a month or only for the days in which has been carried out at least one delivery (excluding therefore the days without no delivery belonging to the considered month). For this reason a second version of the query f is reported which carries out the total_revenue on all the days present in the month considered, thus also considering the days in a given month in which, however, no deliveries have been made. The syntax of this version of the query is that one of SQLite. [2]

```
SELECT
  month,
  year,
  SUM(total_revenue) AS tot_revenue,
  SUM(total_revenue) /
  strftime('%d',
           date(MIN(f.date_id),
              'start of month',
              '+1 month',
              '-1 day'
              )
           )
    AS avg_daily_revenue
FROM fact f, time t
WHERE f.date_id = t.date_id
GROUP BY month, year
```

The `'strftime'` function has been used in order to calculate the number of days present in the considered month.

IV. Analysis of the results

Google Data Studio was used for this analysis. The dataset was automatically generated using a python library called Faker [3]. Note that: data generated by faker is not always logically related. For example, I noticed that when this provides an address in a given city-province-region, the city information might be made up. So, for example, a city could be associated with a wrong (Italian) province. Therefore, the data that will be analyzed is of low quality. However, I report the graphs made because I honestly admit that it was very nice for me to try to create these visualizations. Below are the visualizations deemed most interesting. However, a link to the Google Data Studio document has been added to the references [4]. Below I reported the most interesting graphs.

The figure 2 shows the total revenue per day for all the considered restaurants. The scale of the total revenue is logarithmic and shows that during the time the total revenue has not suffered flexion or increment. The chart shows the time up to the end of August 2017, the more extensive graph is shown in the Google Data Studio project. [4]

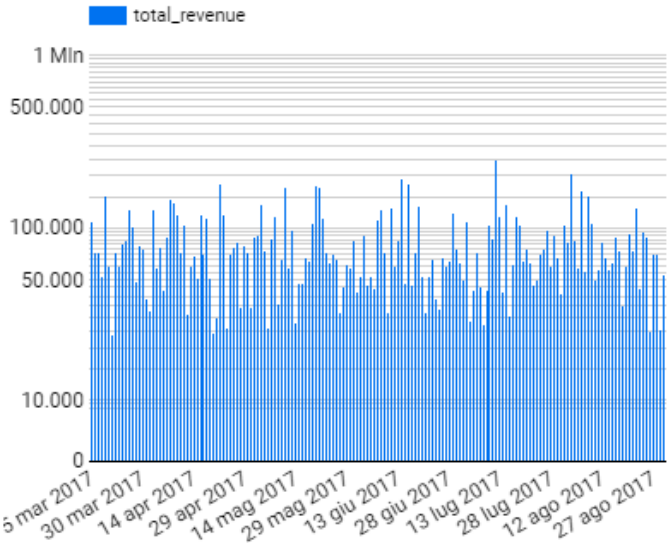

Fig. 2. total revenue per day

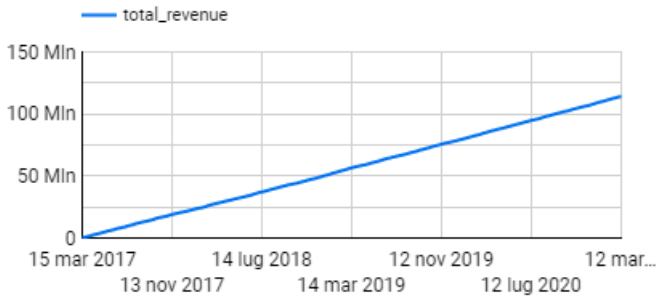

Fig. 3. cumulative total revenue

The figure 3 shows the cumulative gain that affected all interesting restaurants. As we can see from the graph, the curve has a linear trend and mirrors what was analyzed in the previous graph. 2

Instead, the chart 4 shows how revenue has varied over the years for different restaurants. The dataset does not cover the entire year 2017 so not all months for the year 2017 are present in the graph. The data is highly variable between restaurants. Tarantino Group had the absolute highest revenue and did so in 2020 but had no revenue in 2021.

Instead, this chart 5 shows the number of deliveries made in the various years, divided by delivery method. Over the years, the three delivery modes have a common profile. Deliveries made by car are slightly more abundant than those made by scooter or bicycle for all the years that has been considered for the study. However, the distribution among the various delivery modalities is similar.

Sometimes it is convenient to represent the results also through tables that allow to observe immediately a ranking. For example, in the case of the table shown 6, we can see how the total earnings and the maximum delivery time vary between the various modes of transport, differently between

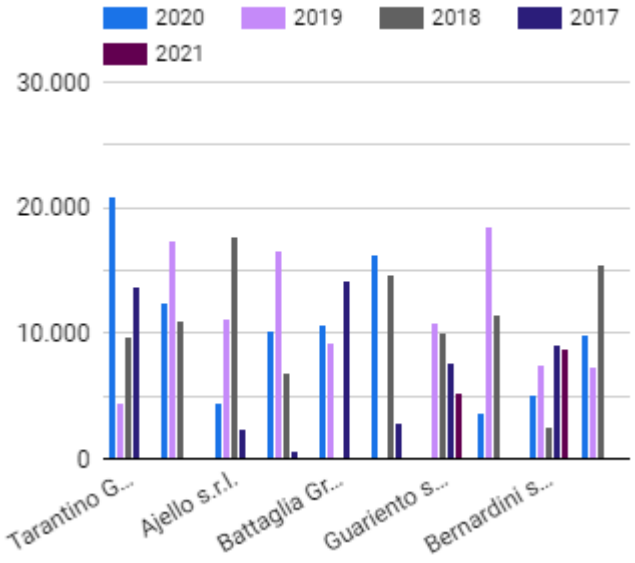

Fig. 4. yearly revenue for each restaurant (divided by name)

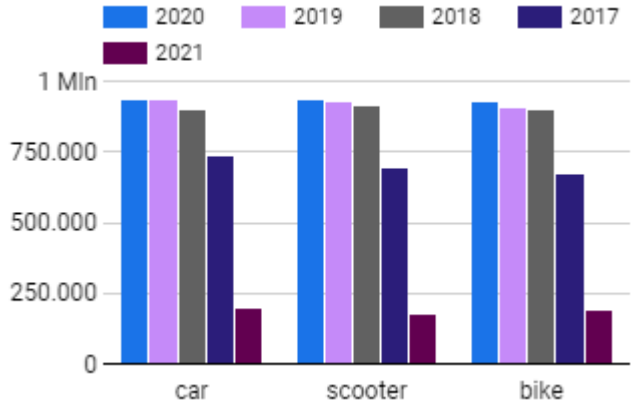

Fig. 5. total number of deliveries per year and transport mode

	transport_...	date...	total_reven...	total_d...
1...	scooter	14 feb...	118.142	119
2...	scooter	21 ge...	108.185	114
3...	car	15 feb...	105.536	120
4...	car	18 ma...	102.120	117
5...	car	12 lug...	101.283	117
6...	scooter	2 mar ...	100.688	110
7...	bike	12 lug...	97.617	118
8...	car	20 feb...	95.865	101

1 - 100 / 4379

Fig. 6. total revenue and the maximum delivery time separately for date and transport mode

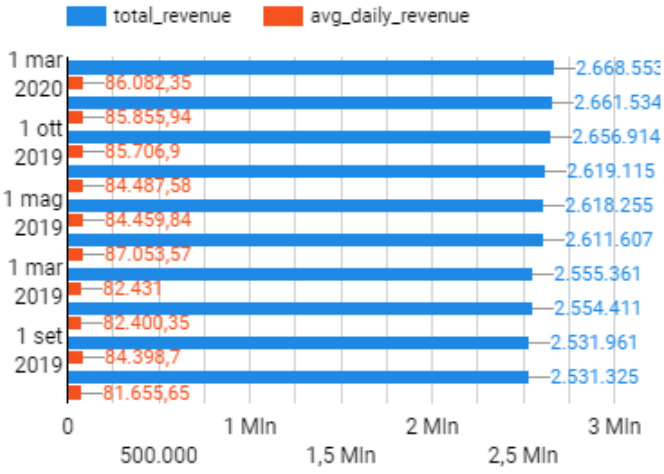

Fig. 7. for each month, select the total revenue and the average daily revenue

the dates. In the table 6, it can be seen that the deliveries that bring in the most total revenue are mostly made by scooter or car. On the other hand, deliveries by bicycle, also as highlighted in the previous chart 5, are less used and the total revenue is lower than the other delivery methods.

The figure 7 shows the revenue for each month and the average daily revenue. The chart represents therefore answer to the query f reported in the previous section. The total revenue for each month has a maximum peak in March 2020 where over all the restaurants the profit was over 2.6 million. Instead, the daily average revenue was of 86082,35 euro.

References

[1] M. Golfarelli and S. Rizzi, "Progettazione concettuale di data warehouse da schemi logici relazionali," in *Atti del Sesto Convegno Nazionale Sistemi Evolluti per Basi di Dati, SEBD 1998, Ancona, Italy, 23-25 Giugno 1998* (P. Atzeni, L. Cabibbo, and M. Panti, eds.), pp. 141–154, 1998.

[2] "Sqlite." https://www.sqlite.org/index.html.

[3] F. Zaninotto, "Faker." https://faker.readthedocs.io/en/master/.

[4] D. A. Giorgio, "Data warehousing - google data studio presentation." https://datastudio.google.com/reporting/4e98c1d9-d7aa-4737-9c90-147a19318832, 2021.