Project 1 - Data Warehousing

Marco Rossini
Politecnico di Torino
Student ID: s291482
m.rossini@studenti.polito.it

***Abstract*—This report illustrates a design proposal of a Data Warehouse aimed at efficiently analyzing data of a food delivery company. A conceptual schema and a logical schema for the Data Warehouse is presented and discussed. In addition, the solved SQL queries needed to meet the problem's specifications are reported, together with the visualizations of the corresponding results.**

I. Problem overview

This section provides an overview of the problem at hand and specifies the associated project requirements.

A food delivery company is interested in efficiently analyzing data related to its own performed deliveries. The company is already equipped with an OLTP database for transaction management, which stores the delivery logs and serves as the company's data source. Specifically, the OLTP database tables are structured as in Figure 1:

```
RIDER (RiderSSN, Surname, Name, DateBirth)
TRANSPORT (TransportID, TransportName)
PAYMENT_MODE (PaymentModeID, PaymentMode,
AvailableDiscount)
RESTAURANTS (RestaurantID, RestaurantName,
RestaurantCategory, RestaurantAddress)
DELIVERY (RiderSSN, DeliveryDate, DeliveryHour,
DeliveryMinute, RestaurantID, TransportID,
PaymentModeID, Amount, Distance, DeliveryDuration)
```

Fig. 1. Company OLTP database structure.

The available OLTP database, however, is designed for fast transactions management, and is not intended for efficiently analyzing complex queries involving aggregations of multiple tables. For this reason, the company's Data Analysts have been interested in developing a Data Warehouse specifically aimed at efficiently handling OLAP queries.

More specifically, the Data Analysts are interested in analyzing the information about the delivery, the *average revenue* for delivery and the *average delivery time* (in minutes).

In particular, the Data Warehouse must be designed to efficiently analyze the following information:

- The restaurant name and city, province and region of the restaurant. The restaurant is located in a specific city.
- The category of the restaurant. A restaurant belongs to only one category. There are five possible categories (*Indian*, *Italian*, *Pizzeria*, *Chinese/Japanese*, *Other*).
- The date, day of the week, if the day is a holiday or not, month, semester and year of the delivery.
- The payment method. There are four payment methods (*Bancomat*, *Credit card*, *Cash* and *Satispay*).
- The transport mode (*Bike*, *Scooter*, *Car*).

Moreover, the Data Analysts are interested in analyzing the following situations:

- Daily, monthly, and yearly revenue, the average delivery time and the number of deliveries for each restaurant.
- Daily, monthly, and yearly revenue, the average delivery time and the number of deliveries for each transport mode.
- The total revenue, the number of deliveries and the average delivery time for payment method.
- The total revenue and number of deliveries for each day of the week and restaurant.

In this report, a Data Warehouse design aimed at meeting these requirements is proposed and motivated. In particular, a conceptual schema and a logical schema for the Data Warehouse is presented and discussed. In addition, the SQL queries needed to meet the problem's specifications are reported, together with the visualizations of the corresponding results.

II. Data Warehouse Design

This section discusses the proposed design for the Data Warehouse, consisting of a conceptual schema and a logical schema.

A. Conceptual schema

The proposed conceptual schema for the Data Warehouse is structured as in Fig. 2:

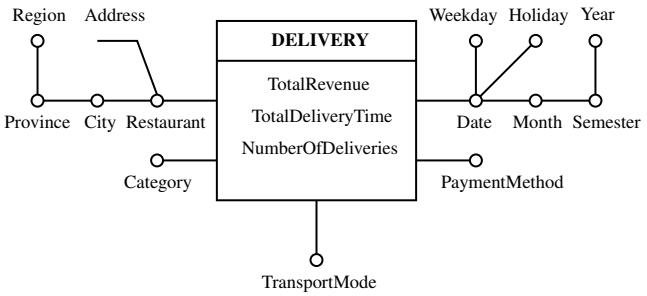

Fig. 2. Proposed conceptual schema for the Data Warehouse.

As the model shows, the attributes of interest for the analysis (as specified by the requirements) were organized into five dimensions:

- DIM_RESTAURANT: stores the information associated to a restaurant. The restaurant's city, province and region are organized as a hierarchy, while the address represents a descriptive attribute.
- DIM_TIME: stores the information associated to the dates on which deliveries were performed. The date's month, semester and year are organized as a hierarchy, while the weekday and wheter the date is a holiday or not (as a boolean value) represent two additional dimension attributes.
- DIM_CATEGORY: stores the information associated to the specific kind of cuisine each restaurant belongs to (*Indian*, *Italian*, *Pizzeria*, *Chinese/Japanese*, *Other*).
- DIM_PAYMENT_METHOD: stores the information associated to the specific payment method used by a customer to pay for a delivery (*Bancomat*, *Credit card*, *Cash* and *Satispay*).
- DIM_TRANSPORT_MODE: stores the information associated to the specific transport mode used by a rider to fulfill a delivery (*Bike*, *Scooter*, *Car*).

It can be noted that not all attributes stored in the OLTP database were modeled: information concerning the riders, the discounts and the delivery distance was discarded, as considered not relevant for the analysis at hand.

Also, the attribute Category, despite being directly related to a restaurant, was organized into a separate degenerate dimension rather than being included in DIM_RESTAURANT. This choice is due to the fact that the project specifications require the attribute Category to be analyzed efficiently. Including Category in DIM_RESTAURANT would have required a join with the associated table for each analysis on the attribute even when other restaurant information was not needed, slowing down the computation. Since the table associated to DIM_RESTAURANT could also be very large, a separate dimension for Category was created in order to provide better performance.

Finally, three measures were created for the fact table: TotalRevenue, TotalDeliveryTime and NumberOfDeliveries. By dividing the former two measures by the latter, the *average revenue* and the *average delivery time* for delivery can be computed, as required by the project specifications.

B. Logical schema

The proposed logical schema for the Data Warehouse is structured as in Fig. 3:

DIM_RESTAURANT (RestaurantID, Restaurant, Address, City, Province, Region)
DIM_TIME (TimeID, Date, Weekday, Holiday, Month, Semester, Year)
DIM_CATEGORY (CategoryID, Category)
FACT_DELIVERY (RestaurantID, CategoryID, TimeID, PaymentMethod, TransportMode, TotalRevenue, TotalDeliveryTime, NumberOfDeliveries)

Fig. 3. Proposed logical schema for the Data Warehouse.

As the model shows, the previously created fact table and dimensions were organized into four tables, leveraging on a *star schema* representation:

- DIM_RESTAURANT: stores the information associated to a restaurant.
- DIM_TIME: stores the information associated to the dates on which deliveries were performed (the attribute Month was formatted as YYYY-MM in order to allow monthly aggregated queries to be computed using a single GROUP BY operation).
- DIM_CATEGORY: stores the information associated to the specific kind of cuisine each restaurant belongs to.
- FACT_DELIVERY: stores the information associated to a delivery, along with the adopted payment method (by a customer) and transport mode (by a rider).

It can be noted that not all the dimensions (previously modeled in the conceptual schema) were associated to a corresponding table. Dedicated tables DIM_RESTAURANT, DIM_TIME and DIM_CATEGORY were created to store the restaurant, time and category information respectively.

Attributes concerning the payment method and the transport mode were instead comprised in the FACT_DELIVERY fact table via push-down. This choice is due to the fact that the project specifications require the attributes PaymentMethod and TransportMode to be analyzed efficiently. Furthermore, such attributes are included in the situations of interest for the Data Analysts (as specified by the project specifications) and are involved in multiple queries. As described in [1] (Para. 9.1.8), including the corresponding degenerate dimensions directly in the fact table will cause its size to increase, but no joins will be necessary to retrieve any information on them, hence providing better performance. The trade-off between memory allocation and faster computation was therefore exploited, favoring the latter in order to ensure a faster retrieval of such frequently queried attributes.

It can be noted that no push-down was performed on the attribute Category. This choice is due to the fact that, although required to be analyzed efficiently by the project specifications, such attribute was never involved in any of the requested queries, and was hence considered to have a slightly lower priority. Furthermore, introducing a supplementary push-down in addition to those already performed on PaymentMethod and TransportMode would have caused an undesirable massive increase of the cardinality of the fact table (from a factor of 12 to a factor of 60) and was therefore avoided.

III. Querying the Data Warehouse

This section reports the SQL queries needed to meet the problem's specifications. The data used for performing the queries was generated and tested using Google Colaboratory. The notebook used for generating the data is available at this link. The notebook used for testing the queries is instead available at this link.

A. For each day, select the total revenue and the average revenue per delivery. Sort the result by date.

```
SELECT Date, SUM(TotalRevenue) as TotalRevenue,
    SUM(TotalRevenue) / SUM(NumberOfDeliveries) as
    AverageRevenuePerDelivery
FROM fact_delivery F, dim_time T
WHERE F.TimeID = T.TimeID
GROUP BY Date
ORDER BY Date;
```

B. Select the yearly revenue and the total number of deliveries for each restaurant. Sort the results by descending yearly revenue.

```
SELECT Restaurant, Year, SUM(TotalRevenue) as
    YearlyRevenue, SUM(NumberOfDeliveries) as
    TotalNumberOfDeliveries
FROM fact_delivery F, dim_time T, dim_restaurant R
WHERE F.RestaurantID = R.RestaurantID AND F.TimeID
    = T.TimeID
GROUP BY Restaurant, Year
ORDER BY YearlyRevenue DESC;
```

C. Separately for each transport mode and year, select the total number of deliveries and the average time for delivery.

```
SELECT TransportMode, Year, SUM(NumberOfDeliveries)
    as TotalNumberOfDeliveries,
    SUM(TotalDeliveryTime) /
    SUM(NumberOfDeliveries) as AverageDeliveryTime
FROM fact_delivery F, dim_time T
WHERE F.TimeID = T.TimeID
GROUP BY TransportMode, Year;
```

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 Month, Restaurant, SUM(TotalRevenue) as
    TotalRevenue, SUM(TotalDeliveryTime) /
    SUM(NumberOfDeliveries) as AverageDeliveryTime
FROM fact_delivery F, dim_time T, dim_restaurant R
WHERE F.TimeID = T.TimeID AND F.RestaurantID =
    R.RestaurantID AND F.TransportMode = 'Bike'
GROUP BY Month, Restaurant;
```

E. Separately for date and transport mode, select the total revenue and the maximum delivery time.

```
SELECT Date, TransportMode, SUM(TotalRevenue) as
    TotalRevenue, MAX(TotalDeliveryTime) as
    MaxDeliveryTime
FROM fact_delivery F, dim_time T
WHERE F.TimeID = T.TimeID
GROUP BY Date, TransportMode;
```

F. Separately for each month, select the total revenue and the average daily revenue.

```
SELECT Month, SUM(TotalRevenue) as TotalRevenue,
    SUM(TotalRevenue) / COUNT(DISTINCT Date) as
    AverageDailyRevenue
FROM fact_delivery F, dim_time T
WHERE F.TimeID = T.TimeID
GROUP BY Month;
```

IV. Analysis of the results

This section discusses the visualizations associated to the previously performed queries. The following charts were generated using Google Data Studio. The related report containing all the generated visualizations is available at this link.

A. Query visualization

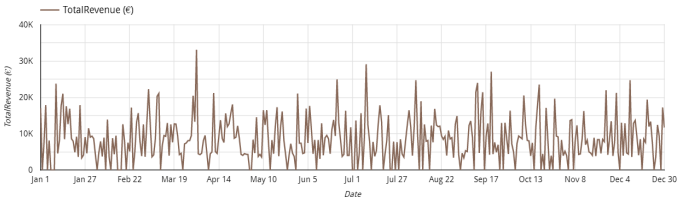

Fig. 4. Daily revenue for year 2020.

For better readability, this visualization was managed by generating three daily revenue charts separately for each year stored in the Data Warehouse (2019, 2020 and 2021), rather than generating a single massive chart for the whole available date range. The adoption of a time series chart allows to study the yearly revenue trends in a familiar way, spotting potential off-target shifts. Unfortunately, no interesting patterns seem to emerge (since data was generated randomly). Actual data may differ (e.g. revenues may be higher on weekends).

B. Query visualization

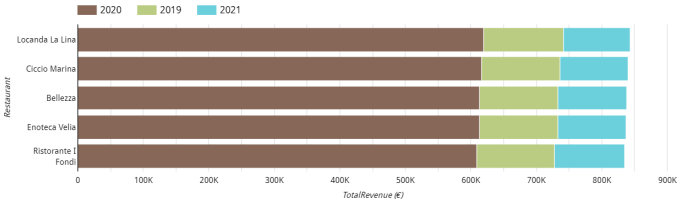

Fig. 5. Total and yearly revenue by restaurant.

This visualization was managed in two ways:

- A first approach consisted in generating a single synthetic chart (Fig. 5), reporting the total and yearly revenue for each restaurant. The adoption of a stacked bar chart allows for an immediate comparison of the total revenues, also highlighting the yearly revenues for each restaurant. Unfortunately, this representation does not allow to meet the requirement of sorting results by descending yearly revenue (only by total revenue).
- A second approach consisted in generating three yearly revenue column charts separately for each year. This visualization choice, although sacrificing the synoptic view, allows for a more fine grained analysis of restaurants revenue performance for each year, also allowing to sort results by descending yearly revenue

It can be noted that *"Locanda La Lina"* turns out to be the restaurant with the highest total revenue, despite being the one that earned the least in year 2021.

C. Query visualization

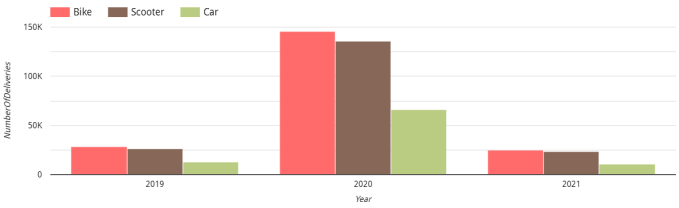

Fig. 6. Total number of deliveries by year and transport mode.

This visualization was managed by generating a chart reporting the total number of deliveries for each year, breaking down each year by transport mode. The adoption of a column chart allows for an immediate comparison of the adoption level of the different transport modes for each year. *Bike* and *Scooter* result as being more widely adopted (since they were assigned higher probability of occurring than to *Car*). An alternative but equivalent version of the described chart was also generated, reporting the total number of deliveries for each transport mode, breaking down each transport mode by year.

D. Query visualization

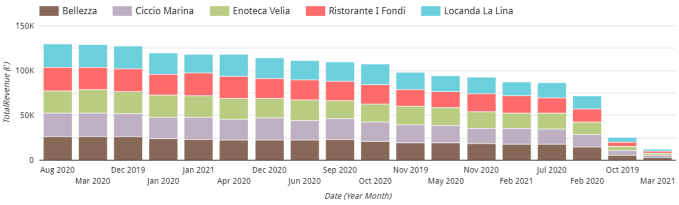

Fig. 7. Monthly revenue by restaurant using *bike* as transport mode.

This visualization was managed in two ways, considering only the records where *bike* was used as the transport mode:

- A first approach consisted in generating a single synthetic chart (Fig. 7), reporting the monthly revenue for each restaurant in descending order. The adoption of a stacked column chart allows for an immediate comparison of the most profitable months, highlighting data composition in a vibrant fashion (which restaurant also performed best for each month?). Unfortunately, this representation does not allow to meet the requirement of displaying the average delivery time.
- A second approach consisted in generating five monthly revenue combo charts separately for each restaurant, sorted by date in ascending order. This visualization choice, although sacrificing the synoptic view, allows for a more fine grained comparison of monthly revenues separately for each restaurant, also allowing to analyze average delivery time trends. The adoption of a combo chart enables to analyze the joint evolution of the two data over time, allowing to spot potential correlations (do longer delivery times always mean higher revenues? Or maybe there are some situations of transport inefficiency?).

E. Query visualization

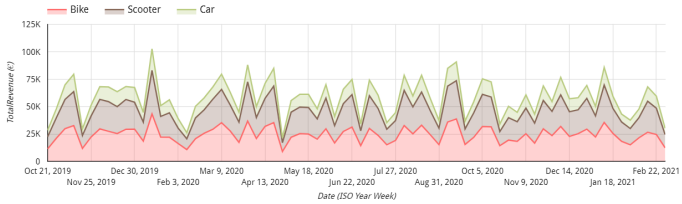

Fig. 8. Weekly revenue by transport mode.

This visualization was managed in two ways, aggregating dates by week for better readability:

- A first approach consisted in generating a single synthetic chart (Fig. 8), reporting the weekly revenue for each transport mode. The adoption of a stacked area chart allows for an immediate comparison between revenues generated by the different transport modes (which are the most profitable ones?), also allowing to analyze, at the same time, their trends over the time dimension (are some transport modes becoming unprofitable? Should they be improved or eliminated?). Unfortunately, this representation does not allow to meet the requirement of displaying the maximum delivery time.
- A second approach consisted in generating three weekly revenue combo charts separately for each transport mode. This visualization choice, although sacrificing the synoptic view, allows for a more fine grained analysis of weekly revenues separately for each transport mode, also allowing to study maximum delivery time trends (which transport modes are the fastest/slowest? Do peaks in the delivery times affect weekly revenues?).

F. Query visualization

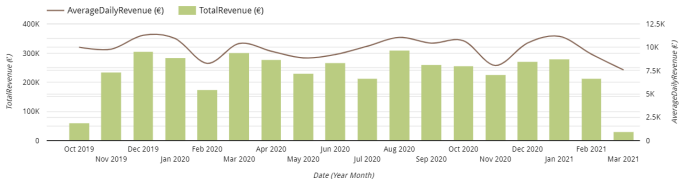

Fig. 9. Total revenue and average daily revenue by month.

This visualization was managed by generating a single synthetic combo chart (Fig. 9), reporting the total revenue and the average daily revenue for each month. This visualization choice allows for an immediate comparison of the monthly revenues, also allowing to study average daily revenue trends. The adoption of a combo chart enables to analyze the joint evolution of the two data over time, allowing to spot potential correlations (do higher average daily revenues always mean higher total revenues? Or maybe there are months characterized by just a few days of very profitable work?).

REFERENCES

[1] M. Golfarelli and S. Rizzi, *Data Warehouse: Teoria e pratica della progettazione*. McGraw-Hill New York City, 2006.