Data Warehousing: Lab 2 Extract-Transform-Load Design

Javier Ferrando, Ricard Monge Calvo

November 2019

The aim of this exercise is to design an extract-transform-load process for the *ACME-flying* use case.

1. Considerations

First of all, we consider as fixed the structure of the target tables, provided in the problem statement. In that sense, we conform to the names, types, lengths and number precision of the target tables.

Moreover, we only consider the tables of AIMS and AMOS which provide the desired information to fulfill the target requirements. Accordingly, the data quality and robustness checks that we perform following to the business rules are only for the considered tables.

2. Data quality

Regarding the connection between *Flights* and *OperationInterruption*, we ensure that all maintenance events are correctly matched with its corresponding flight by the *scheduledeparture* field, which we assume (and checked) is identical to the date specified in the *flightID*; and by the *aircraftRegistrationID* and *delayCode*.

Furthermore, we checked the expected length of each maintenance event according to its kind, and observed that none of them fulfilled the expected duration. Thus, for the sake of the exercise we do not apply any action on this rule.

In the case of the data quality of flight information, we ensure that flights have a departure timestamp earlier than an arrival timestamp for both schedule and actual cases, by checking the differences of the *scheduledeparture/arrival* and *actualdeparture/arrival*. In addition, we checked that the consecutive time slots of a given airplane do not overlap, by checking that the arrival time of a slot happens earlier than the departure time of the next.

Finally, although every *Maintenance* that appears in AIMS has to appear in AMOS, we assume the reverse is not true and so we use the more complete tables in AMOS.

3. Robustness

In order to achieve robustness in our ETL we impose a logical order in our job, by first checking the connections and existence of the tables/files, to then load the dimensions and end up with the facts.

Furthermore, we ensure that the foreign keys in the fact tables are present in the dimensions, removing those that are not.

Finally, we ensure that successive loads do not duplicate data or generate primary key violation errors, by checking changes and updating instead of inserting.

4. Instructions

To run the ETL process, one has to open the job named as DW_Lab2 and execute it. This job first runs a $job_{check-input-outputs}$, then a $job_{dimensions}$ and finally, job_{facts}. Inside $job_{dimensions}$ we have a transformation for each dimension, respecting the order of loading first *months* dimension and then *temporaldimension* to ensure foreign key restrictions. Equally, inside job_{facts} we have a transformation for each fact table.