**ETL Project**

**Summary – Group 7**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Mohamed Jouhari (DE) | Vincent Pinneau (DA) | Christine Ismail (DE) | Abdifatah Farah (DE) |
| STA Phase | * **(technical)** Resolved the encoding problem that prevented the order\_reviews\_dataset to be loaded in the STA phase | * There were no debates on how to do most of the STA phase. Everyone agreed to do the Foreach Loop Container to put the three orders csv files into a unique table. * The part where we had the most difficulties was solving the encoding issue with the order\_reviews\_dataset. Mohamed solved it first and shared with rest of the group. | * did a table in the database for each dataset and used the foreach loop for orders * used utf-8 for all datasets | * In the Staging phase, I set up SQL tables to match our raw data files (orders, customers, etc.) and used SSIS to automate data imports, especially with multiple files for orders. We hit a technical issue with special characters in the order\_reviews\_dataset, so I adjusted the encoding to UTF-8, ensuring all data loaded smoothly. This gave us a clean starting point for transformations ; |
| ODS Phase |  | * **(technical)** In the original final template of the project, the ADM Rejects table was truncated at the beginning of each ODS. In my ETL I didn’t include an ADM Reject truncate because I preferred to keep a ’tracking’ of the past rejects, and filter by date in SQL. I said that we could keep the truncating in the final version but that the truncation had to be only done once, because the different steps of the ODS all shared the same Rejects table. Also, initially, in the sequence container, the ODS phases were not linked together, so I asked that we link them together and put the ODS (ODS Orders) that had the truncation of the ADM reject table at the top of the ODS pipeline, to be sure that the rejects from all the ODS packages were taken into account when launching the entire pipeline. * **(Modelling)** In my initial model, only one date was kept and converted to date: order\_approved\_at. The reasoning behind this was that the information given by the dates ‘order\_delivered\_carrier\_date’ and ‘order\_delivered\_customer\_date’ was already partially given by the ‘order\_status’ column. I considered that the combination of ‘order\_approved\_at’ and ‘order\_status’ were enough for a database destined to be used by sales teams. The other dates, especially the estimated\_delivery\_date or the shipping\_limit\_date were more for a customers relations/logistics team that would need to build a visualization table to follow in real time the progress of the deliveries (some type of KPI-like table where rows or symbols would turn red if the delivery was made after the ‘estimated\_delivery\_date’). In the final version of the group ETL, purchase date and delivered\_date were kept to be able to answer questions like what parameters can influence the length of a delivery. * **(Modelling)** For the payments, a similar reasoning was adopted when choosing not to keep the payment\_sequential or payment\_installments, considering that this would be more useful for an accounting team than a sales team. | * Created the tables in the ods database exactly like the sta without removing certain columns * Only put added the date and col of the reject without specifying further description | * Throughout the project, I used SQL for organizing, cleaning, and analyzing data. In the ODS phase, I created a **Rejects table** to track data errors, which made troubleshooting easier. In the Data Warehouse phase, I wrote SQL queries to pull insights, like identifying top product categories and analyzing payment methods’ impact on order times. SQL was key to making our data accurate and valuable for business use. |
| DWH Phase | * **(Modelling)** Wanted to have all the geolocation information of customers and sellers regrouped in a geolocation dimension table to reduce the size of the data but also to make the model more logical. * **(Technical)** Modifications made on the GPS values * **(Technical/Modelling)** Creation of the column Total\_invoiced, payment\_value deleted, and columns price + freight\_value moved to products table | * **(Modelling)** Moving the city/state information from the customer/seller tables to the geolocation dimension to avoid duplicates. Also, from a data analyst point of view, keeping all the gps geolocations in one table and not splitting them between the customer and seller tables, can allow us to do a visualization comparing customer et seller location on a unique map more easily. The only thing that would need to be added is a category column in the geolocation table differentiating customers and sellers to filter or do a legend on the map. * **(Modelling)** Proposed to put most information regarding orders (and not orderitems) in a separate dimension table to prevent duplicating information in the orderitems fact table since some orders appear in several rows of the fact table (those that have multiple items). In the final form of the project, a mixed strategy was adopted. The strategy previously stated was kept for the orders information that could take a lot of memory and processing power such as comments, so a separate DimReview table was created. On the other hand, the rest of the orders information was put in the main fact table to make it easier to query, but there was still a strategy of maintaining the size to a minimum by taking only the most relevant columns (choices made in ODS phase) | * At first did not follow the snowflake scheme but after the meetings agreed to the fact that this scheme will help us retrieve data in a better way so followed the team suggestion and linked the DimGeo table with DimCustomer and DimSeller and the DimDate with DimRev * Did not add a lot of details such as Day, Day of Year, Suffix…to the DimDate Table I preferred keeping it simple | * I shared the team’s vision for the Data Warehouse schema, including ideas like organizing geolocation details into a single dimension, but I didn’t have the time to work on the implementation directly. |
| Data analysis | * SQL * Report comments/analysis | * Report comments/analysis | * SQL | * SQL |