المملكة المغربية
ROYAUME DU MAROC
المندوبية السامية للتخطيط
HAUT-COMMISSARIAT AU PLAN

INSEA

Master of Information Systems and Intelligent Systems

Data Warehouse Project

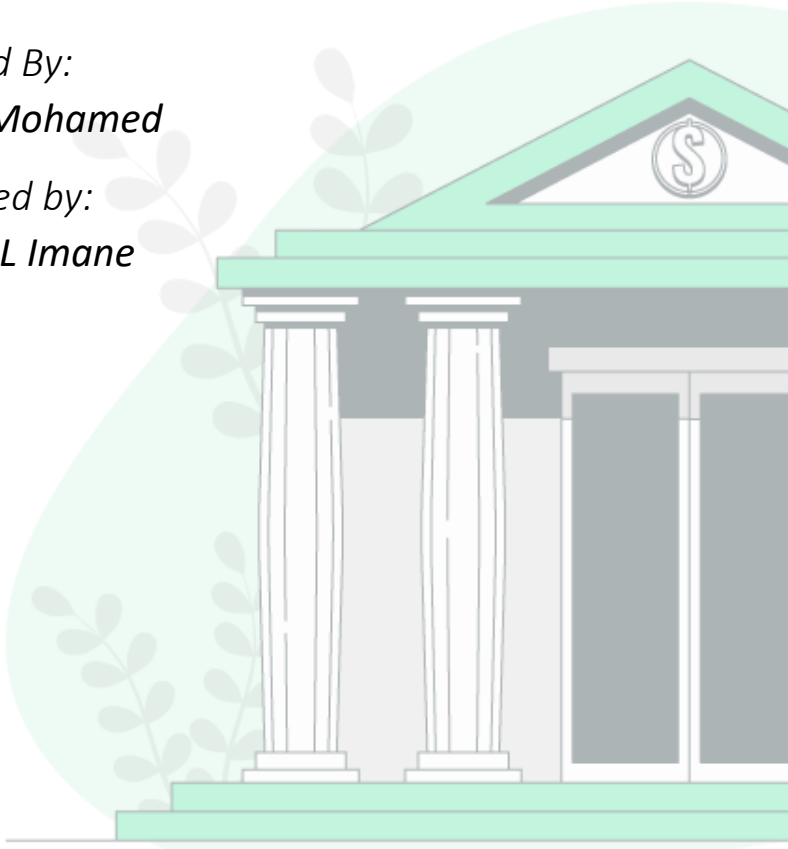
Extract, Transform & Load Process Implementation on Bank Transactions Dataset

Realised By:

ELAZZAOUI Mohamed

Supervised by:

Mme. HILAL Imane



Academic Year: 2022/2023

Table of Contents

Introduction

Data warehousing is the process of collecting, storing, and managing large amounts of data from various sources to support business intelligence and analytics. The goal of data warehousing is to provide a centralized, integrated view of an organization's data, making it easier for business users to access and analyze the data they need to make informed decisions.

A data warehouse typically includes data from a variety of sources, such as transactional systems, external data sources, and other databases. The data is then cleaned, transformed, and integrated into a single, consistent format. This process is known as data integration.

Once the data is integrated, it is then loaded into the data warehouse for storage. The data warehouse is typically designed to support high-performance querying and reporting, and may include features such as indexing, partitioning, and compression to optimize performance.

Data warehousing also includes the process of creating a data mart, which is a subset of the data warehouse that is tailored to the specific needs of a particular business unit or department. Data marts are often used to provide a more focused view of the data for specific groups of users, such as marketing or finance.

I. General Notions:

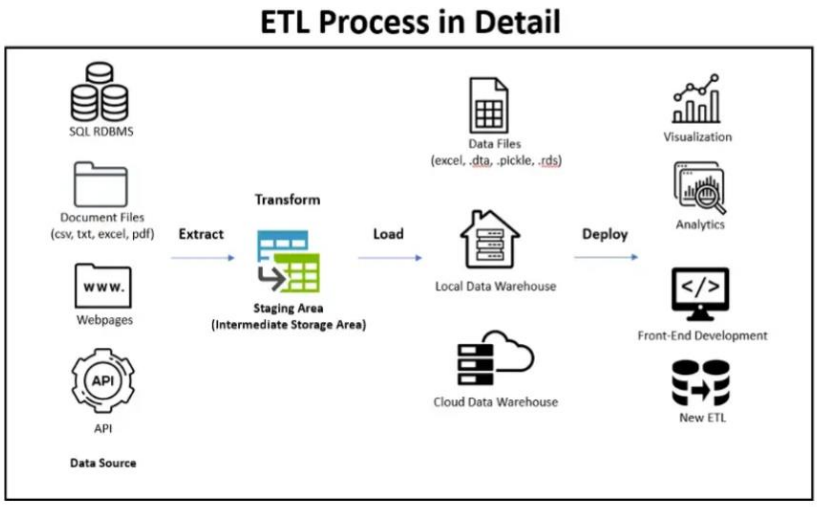
1. What is Data Warehouse?

A data warehouse is a system used for reporting and data analysis, and is considered a core component of business intelligence. It is a relational database that is designed for query and analysis rather than for transaction processing. It typically contains historical data derived from transaction data, but can also include data from other sources. Data is stored in a way that is optimized for reporting and analysis, rather than for transaction processing. The data is then used to create informative and actionable business reports and data visualizations for decision-making purposes.

2. What is ETL?

Extract, Transform & Load Process (ETL) is an important process that is used to populate a data warehouse with the data that is needed for reporting and analysis. It is used to extract data from various sources, clean and transform it into a format that is compatible with the data warehouse, and then load it into the data warehouse. This process is critical for ensuring that the data in the data warehouse is accurate, complete, and up-to-date, and that it can be used to create meaningful and actionable reports and data visualizations.

- Extract: The first step in the ETL process is to extract data from a variety of sources. These sources can be structured or unstructured, and can include databases, flat files, or even social media feeds.
- Transform: The extracted data is then transformed to fit the specific format and structure required by the target system. This can include tasks such as data cleaning, data validation, and data normalization.
- Load: The final step is to load the transformed data into the target system, such as a data warehouse or a data mart.



3. What is Talend?

Talend is a popular ETL (Extract, Transform, Load) tool that is used for data integration. It offers a wide range of pre-built connectors and transformation components that make it easy to extract data from various sources, transform it to fit the target system's structure and load it into the target system.

II. Modeling And Creation of a Data Warehouse for a Bank Clients Datasets:

1. Datasets:

The datasets provided consist of four different types of information in different formats (csv & json).

The "clients.json" dataset contains basic information about bank clients, including their ID, full name, address, phone number, email, workplace, birthdate, registration date, gender, income, expenses, credit, and deposit.

This dataset can be used to understand the demographics and financial information of the bank's client base.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
19	18	Зоя Никифоровна Силина	п. Моршанск, бул. Ватутина, д. 6, 829748	8 446 399 2976	timofeevevdokim@	с. Ногинск (Моск.), наб. Чайковского, д. 80 стр. 513, 574767	1972-06-16	2015-05-06	F		75635		1	
20	19	Клавдия Феликсовна Гурьева	д. Тихвин, бул. Тюленина, д. 2/5, 159920	+7 977 432 6073	nikiforkabanov@ra	д. Белогорск (Амур.), алл. Нефтяников, д. 93, 851164	1972-09-30	2014-10-07	F	224368	77545	1		
21	20	Агата Ефимовна Королева	с. Калач-на-Дону, бул. Базарный, д. 104 к. 45, 103883	82167746073	flukina@yandex.ru	д. Владивосток, ш. Парковое, д. 9/1, 528054	1970-10-13	2019-04-11	F		75361			
22	21	Елизавета Богдановна Громова	к. Нарткала, наб. Коммуны, д. 5/6 стр. 768, 755228	8 (849) 518-4640	nturova@gmail.co	к. Кунгур, ш. Ключевое, д. 529 к. 66, 162780	1992-12-09	2017-08-28	F		79467		1	
23	22	Громов Автоном Ерофеевич	клх Киров (Вятка), ул. Сельская, д. 5/9 к. 9, 275499	+7 (735) 435-83-28	trofim2013@gmail	клх Красноуфимск, ш. Тургенева, д. 775 к. 7, 137756	1995-02-28	2015-04-01	M		89454			

The "transactions.csv" dataset contains information about individual transactions made by bank clients, including the transaction ID, client ID, product category, product company, subtype, amount, date and transaction type.

It can be used to understand the spending patterns and transaction behavior of the bank's clients.

	A	B	C	D	E	F	G
1	id	client_id	product_category	product_company	amount	date_start	date_end
2	766278	825	29		3990	2012-01-25	
3	658393	709	4	Яндекс.Музыка	199	2012-01-27	2020-02-27
4	354548	383	4	Boom	149	2012-01-28	
5	515830	556	4	Spotify	169	2012-02-01	
6	799560	862	4	YouTube Music	169	2012-02-20	2020-05-20
7	597370	643	4	YouTube Music	169	2012-02-28	
8	522160	562	29		1778	2012-03-07	
9	692999	747	4	Apple Music	169	2012-03-09	
10	781838	843	4	Яндекс.Музыка	199	2012-03-28	
11	521377	562	4	Яндекс.Музыка	199	2012-05-09	2020-10-09
12	86701	93	29		6595	2012-05-14	
13	464228	501	4	Google Play Музыка	159	2012-05-22	
14	414164	447	4	Яндекс.Музыка	199	2012-05-24	
15	306278	331	4	Google Play Музыка	159	2012-06-16	2020-08-16

The "subscriptions.csv" dataset contains information about recurring transactions made by bank clients, including the ID, client ID, product category, product company, amount, date start, and date end. This dataset can be used to understand the clients' recurrent expenses and help the bank to optimize their revenue.

	A	B	C	D	E	F	G
1	id	client_id	product_category	product_company	amount	date_start	date_end
2	766278	825	29		3990	2012-01-25	
3	658393	709	4	Яндекс.Музыка	199	2012-01-27	2020-02-27
4	354548	383	4	Boom	149	2012-01-28	
5	515830	556	4	Spotify	169	2012-02-01	
6	799560	862	4	YouTube Music	169	2012-02-20	2020-05-20
7	597370	643	4	YouTube Music	169	2012-02-28	
8	522160	562	29		1778	2012-03-07	
9	692999	747	4	Apple Music	169	2012-03-09	
10	781838	843	4	Яндекс.Музыка	199	2012-03-28	
11	521377	562	4	Яндекс.Музыка	199	2012-05-09	2020-10-09
12	86701	93	29		6595	2012-05-14	
13	464228	501	4	Google Play Музыка	159	2012-05-22	

The "amount" column in the "transactions.csv" dataset and "subscriptions.csv" datasets is in Ukrainian hryvnia (UAH) currency.

Lastly, the "categories.csv" dataset contains information about standard transaction categories used by many banks worldwide, including the ID, name, description and mcc-code. This dataset can be used to classify transactions and understand how clients are spending their money.

	A	B	C	
1	id	name	description	mcc-code
2	1	Каршеринг	Краткосрочная аренда авто с оплатой по минутам или часам — не включая услуги такси и аренду в дилерских центрах	7512, 4121
3	2	Супермаркеты	Покупки в супермаркетах и продуктовых магазинах	5297, 5298, 5300, 5411, 5412, 5422, 5441, 5451, 5462, 5499, 5715, 5921
4	3	Такси	Услуги такси (каршеринг не входит в данную категорию)	4121
5	4	Музыка	Покупки в магазинах музыки и музыкальных инструментов	5733, 5735
6	5	Фастфуд	Покупки в ресторанах быстрого питания	5814

Overall, these datasets provide a comprehensive view of the bank's clients and their financial activities, which can be used to inform business decisions and improve the bank's operations.

2. KPIs:

KPI stands for Key Performance Indicator. It is a metric used to measure and track the performance of a business, process, or specific activity. KPIs are used to evaluate the effectiveness of an organization in achieving its strategic goals and objectives. They provide a way to measure progress and success, and can be used to identify areas where improvements are needed.

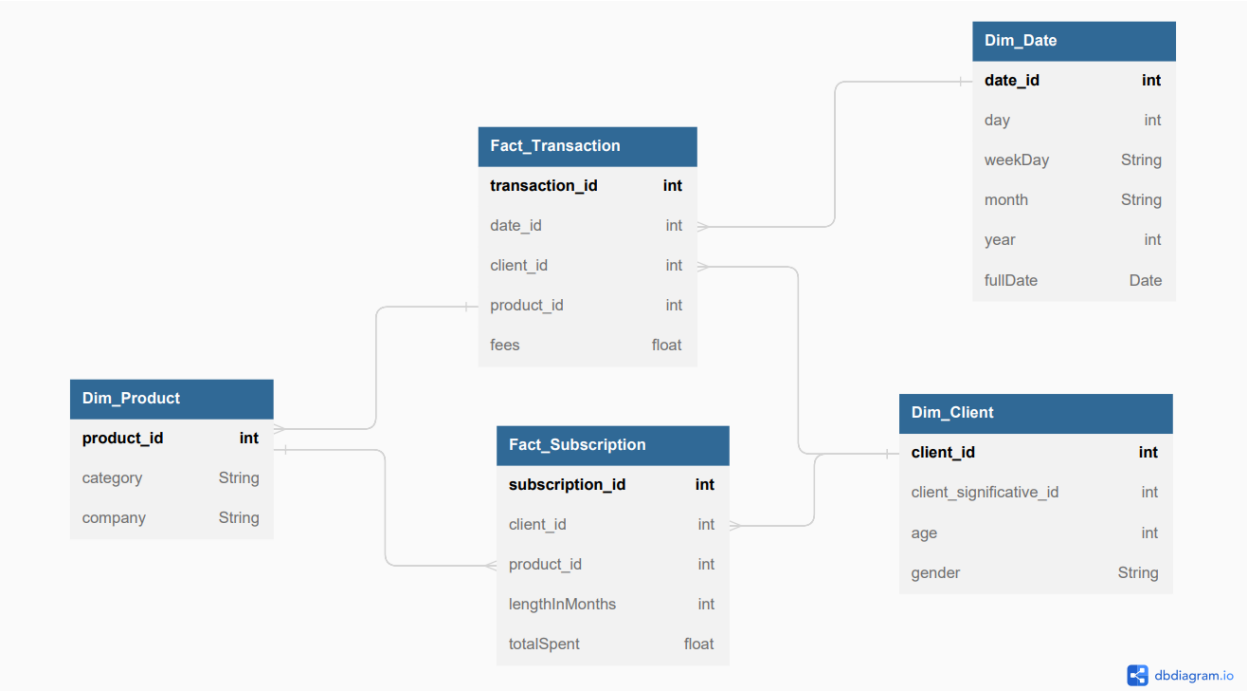
Our goal is to analyze the provided datasets in order to calculate the following KPIs:

- Transaction fee: the fee for each transaction can be done by using the "transactions.csv" dataset. This metric can be used as a KPI to evaluate how much revenue is generated by clients' transactions, including fees. The fee can be calculated in various ways depending on the bank's policy, it can be a fixed amount per transaction, a percentage of the transaction amount, or a combination of both. In our case we considered that transactions fee can be calculated using this formula: Transaction fee = amount*0.0349+17.99
- Subscription length: By using the "subscriptions.csv" dataset, we can calculate the length of a subscription by subtracting the "date_start" column from the "date_end" column for each subscription. This will give us the number of days, months or years that a client has had a subscription. This metric can be used as a KPI to evaluate how long clients retain their subscriptions, this information can be used to understand how to retain clients, and how to offer them new services.
- Total amount spent on subscription: By using the "subscriptions.csv" dataset and the subscription length calculated, we can calculate the total amount spent on subscriptions by multiplying the "amount" column by his subscription length. This will give us the total amount of money spent by clients on each subscription. This metric can be used as a KPI to evaluate how much expenses is generated by clients' subscriptions, and how much clients are loyal to this product.

3. Star Schema:

A star schema is a type of data modeling technique that is commonly used in data warehousing to organize data for efficient querying and analysis. In a star schema, the main table is called the fact table and it contains the measures or facts of the business. The fact table is connected to one or more-dimension tables, which contain the attributes that describe the facts.

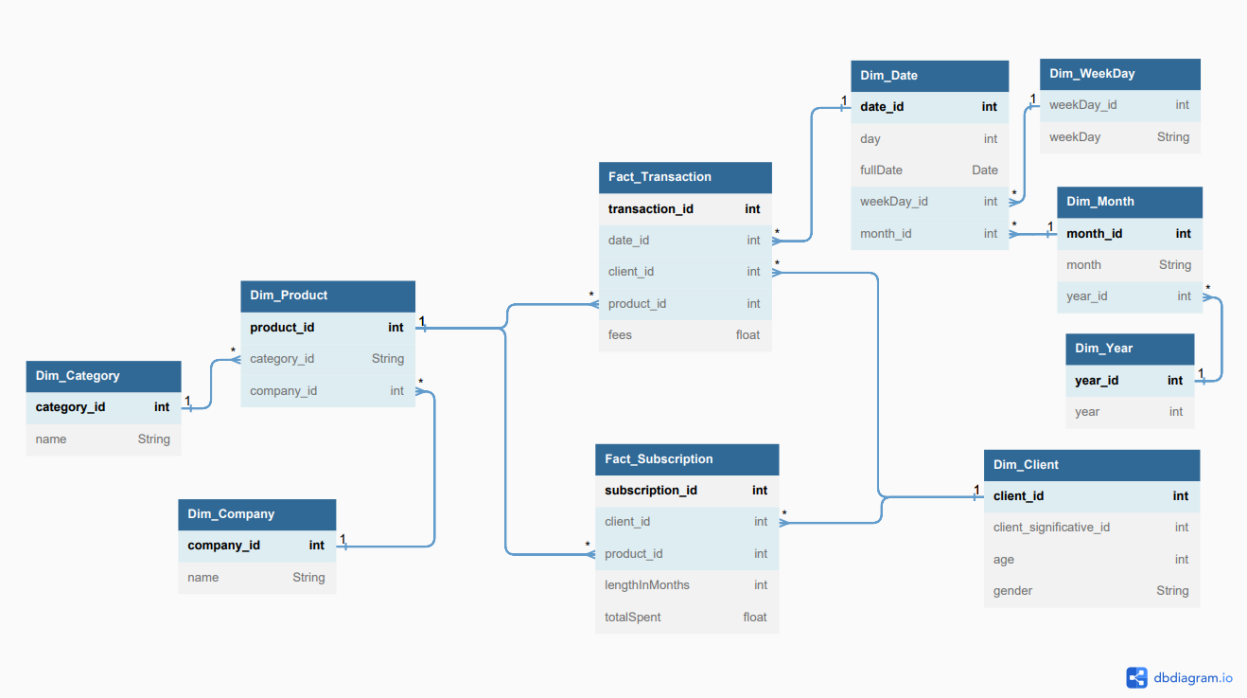
To calculate the KPIs described in the previous paragraph, we could use the following star schema:



4. Snowflake Schema

A snowflake schema is a star schema with normalized dimension tables.

To calculate the KPIs, we could use the following snowflake schema:



5. Star Schema or Snowflake Schema?

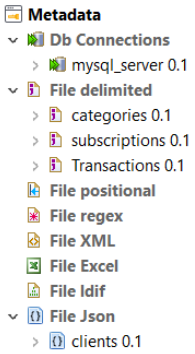
Our project is simple and light in size, so it is preferable to use a star schema because a star schema is easier to design and implement. It can be more efficient to query than a snowflake schema because there are fewer joins between tables.

Because of the denormalized data, a star schema can require more storage space than a snowflake schema. but in our case, we don't have much data that needs a lot of storage space.

So, I choose to work with a star schema.

6. ETL Implementation:

In the beginning, we need to define data sources & targets for Talend:

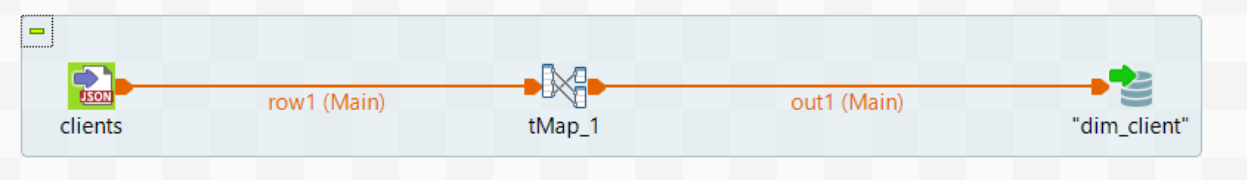


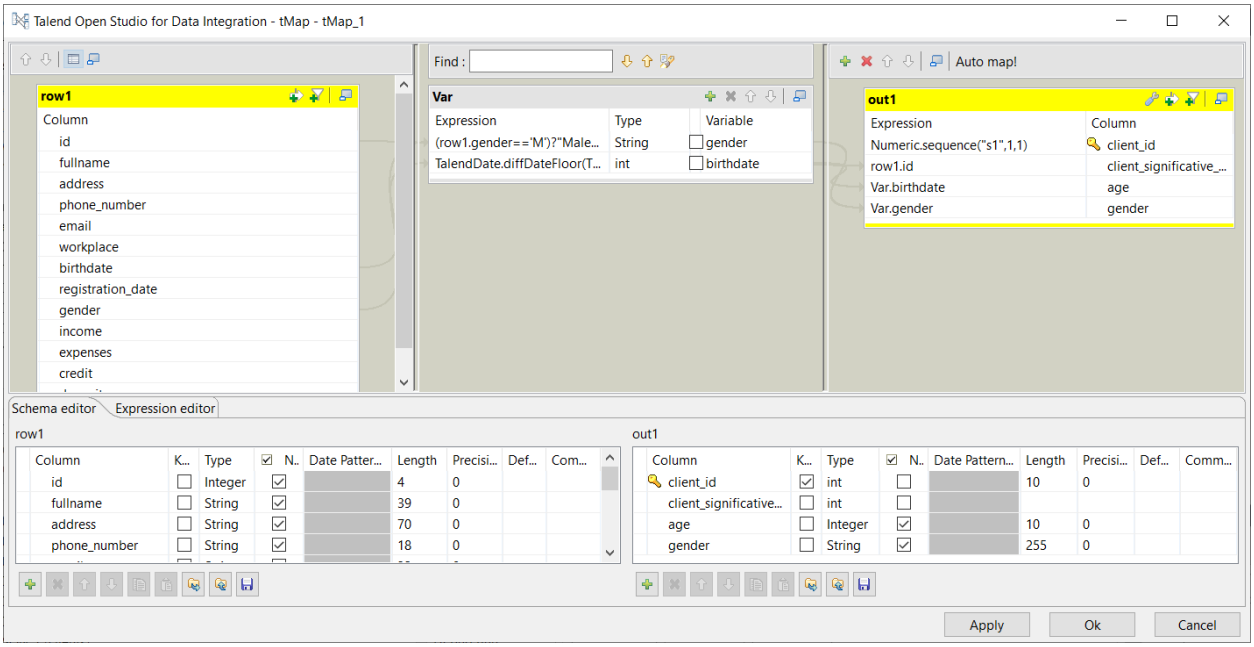
And before performing ETL, we should know about some used components:

- tMap : Used to transform and map data between input and output sources. It allows for the manipulation and joining of data from various sources, as well as the application of various data quality rules and transformations. It is a versatile component that can be used to perform a wide range of data integration tasks.
- tUniqRow : Used to remove duplicate records from an input flow of data. It compares the input rows against each other and removes any that have the same values in the selected columns. It can be used to ensure that the output data is unique and does not contain any duplicates. The component can be configured to compare specific columns of the input data, and can also be set to output both unique and duplicate rows if desired.

1. Creating Dimensions:

- Client Dimension:





Client_id auto incremented integer generated using function: Numeric.sequence("s1",1,1).

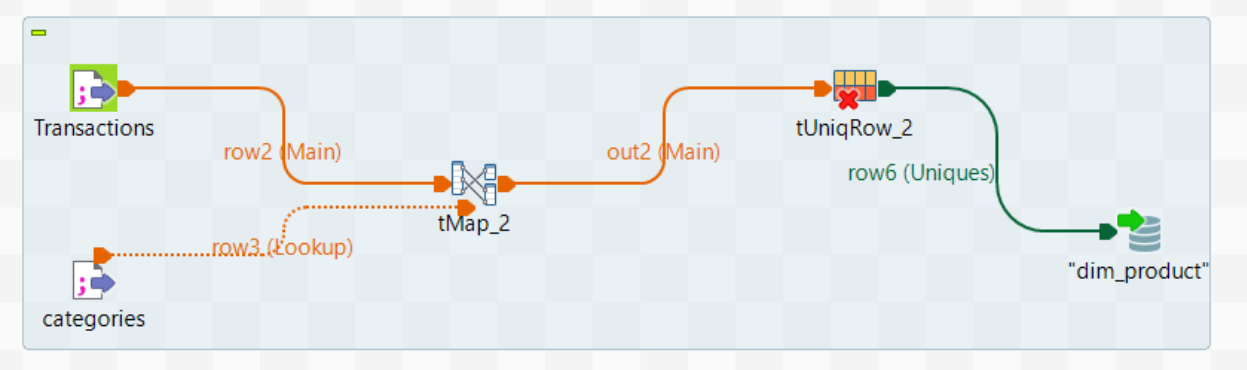
Client_significative_id= same client id on “clients.json“ file.

Gender transformed to clear text using instruction: (row1.gender=='M')?"Male":"Female"

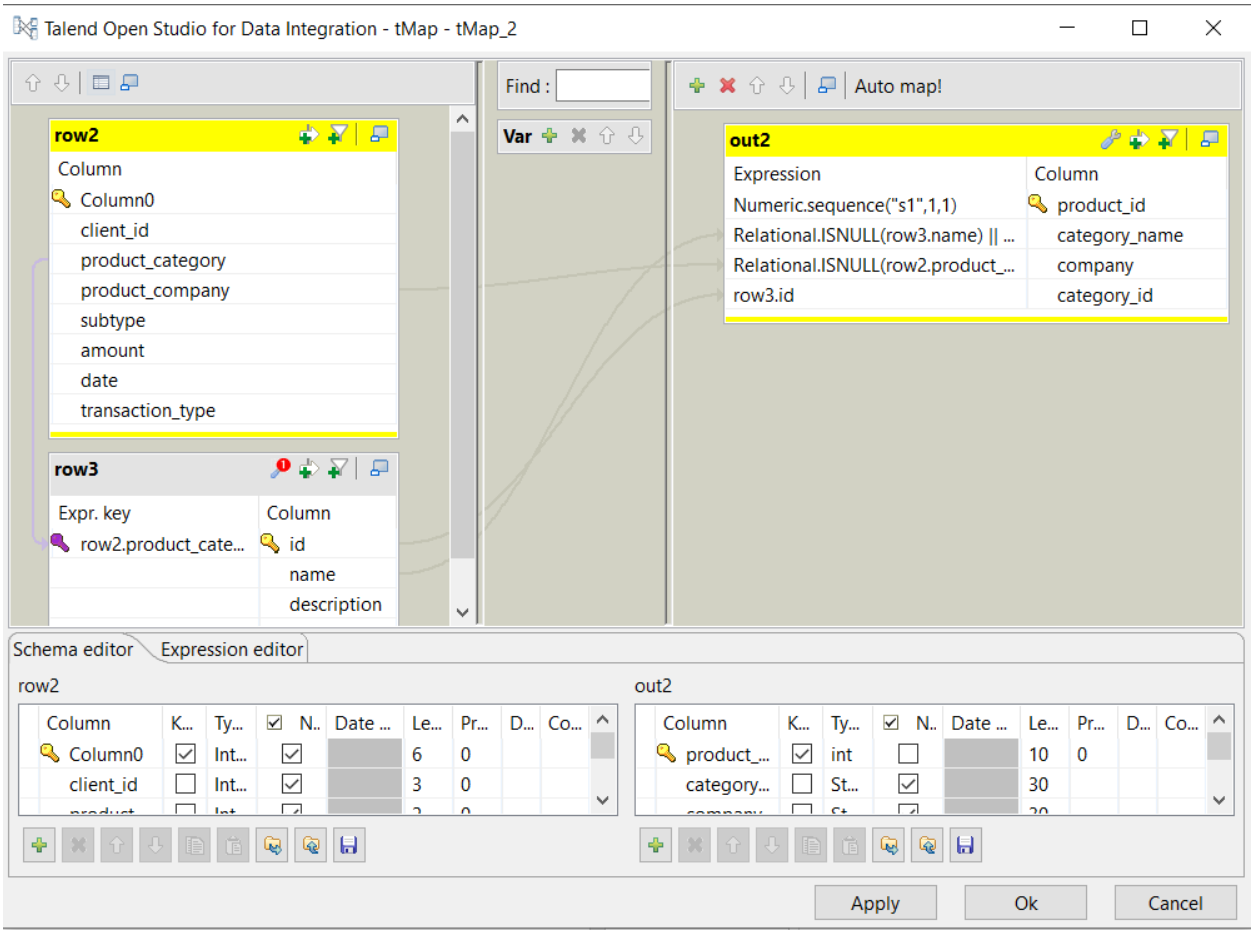
Age calculated using diffDateFloor built-In function:

TalendDate.diffDateFloor(TalendDate.getCurrentDate(),row1.birthdate,"yyyy").

- Product Dimension:



Transactions file contains only category id, so we add categories file to lookup for category name.



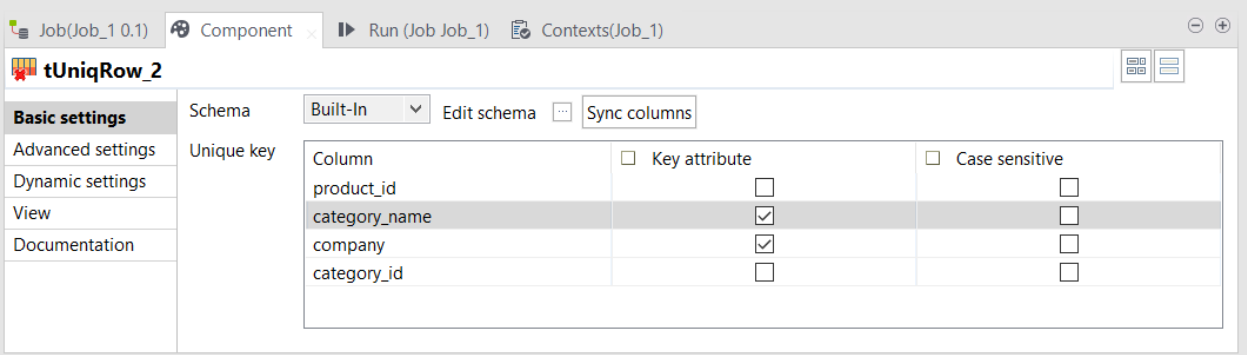
We should match product id in both files with each other to perform a correct inner join between files.

Product_id auto incremented integer generated using function: Numeric.sequence("s1",1,1).

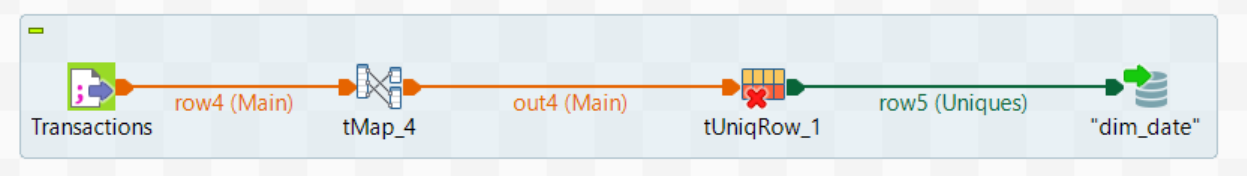
Category_name or Company will be replaced with “Unknown” if they are null.

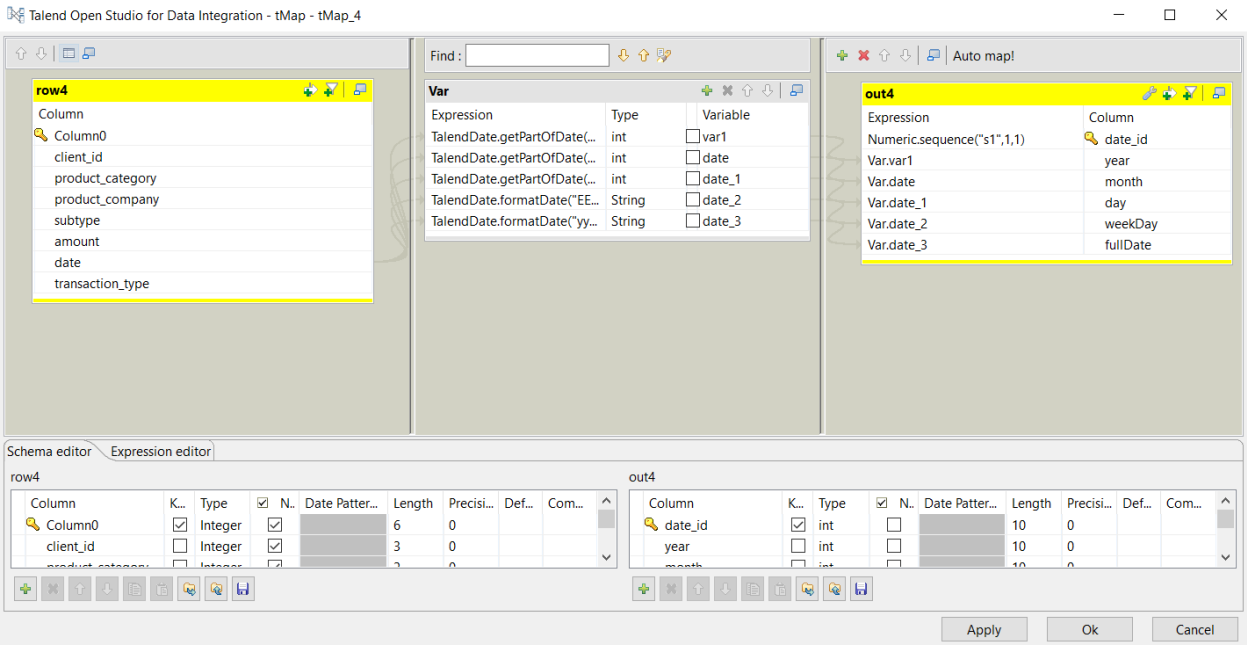
You will ask, why I kept Category_id? I will need it to match the product when the fact table creation.

Then we remove products with identic category & company because we mustn’t keep duplicate rows.



- Date Dimension:





In this stage, we extracted year, month, day (1, 2, …, 31) & day of the week (Monday,Tuesday …) from date, and we preserved the full date to use it in “lookup” when fact table creation.

Date_id auto incremented integer generated using function: Numeric.sequence("s1",1,1).

Year extracted using function: TalendDate.getPartOfDate("YEAR", row4.date)

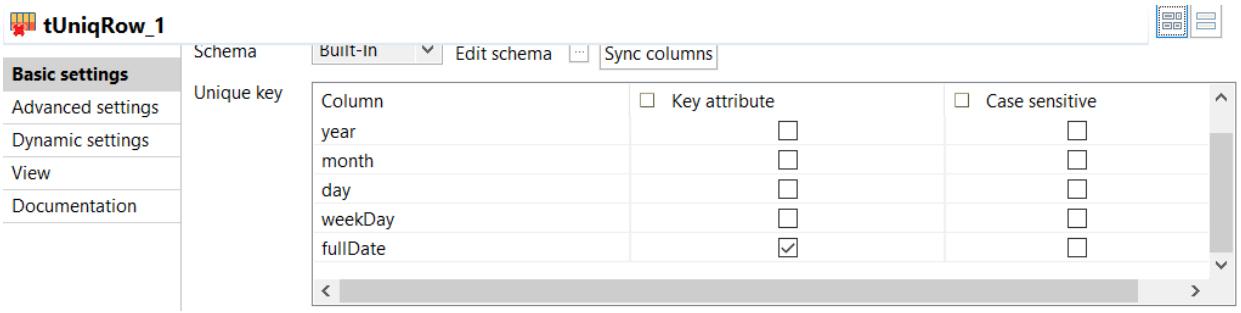
Month extracted using instruction: TalendDate.getPartOfDate("MONTH", row4.date)+1

Day extracted using function: TalendDate.getPartOfDate("DAY_OF_MONTH", row4.date)

WeekDay extracted using function: TalendDate.formatDate("EEEE",row4.date)

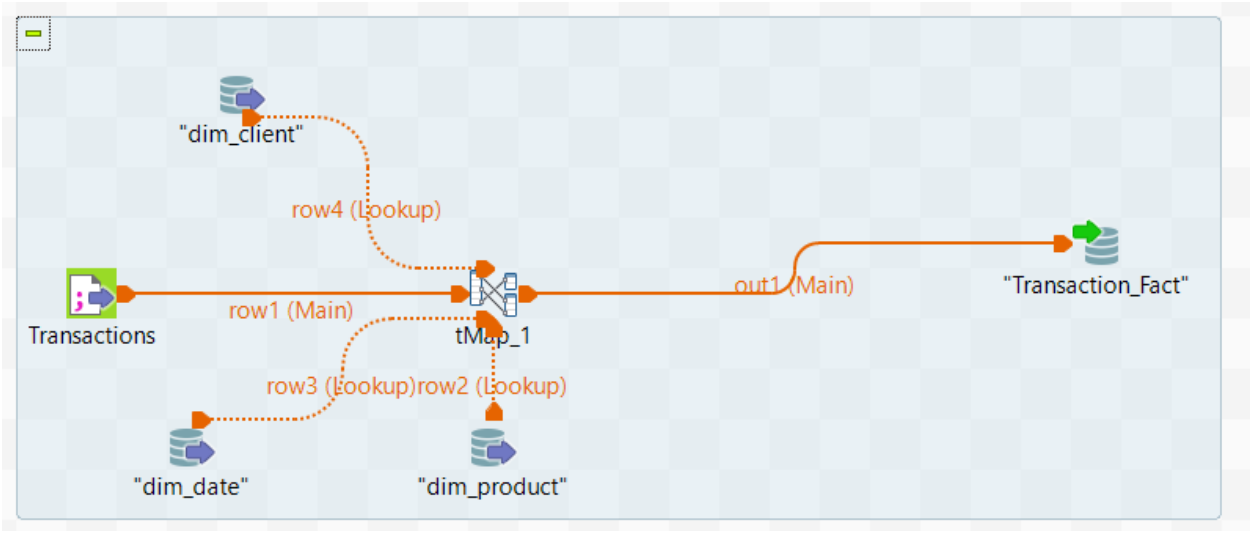
FullDate transformed using function: TalendDate.formatDate("yyyy-MM-dd",row4.date)

Then we delete repeated dates before loading it:



2. Creating Fact Tables:

- Transaction Fact Table:



In this step, for each row we should match his attributes values with attributes values in dimensions to perform the “Inner Join” between data sources.

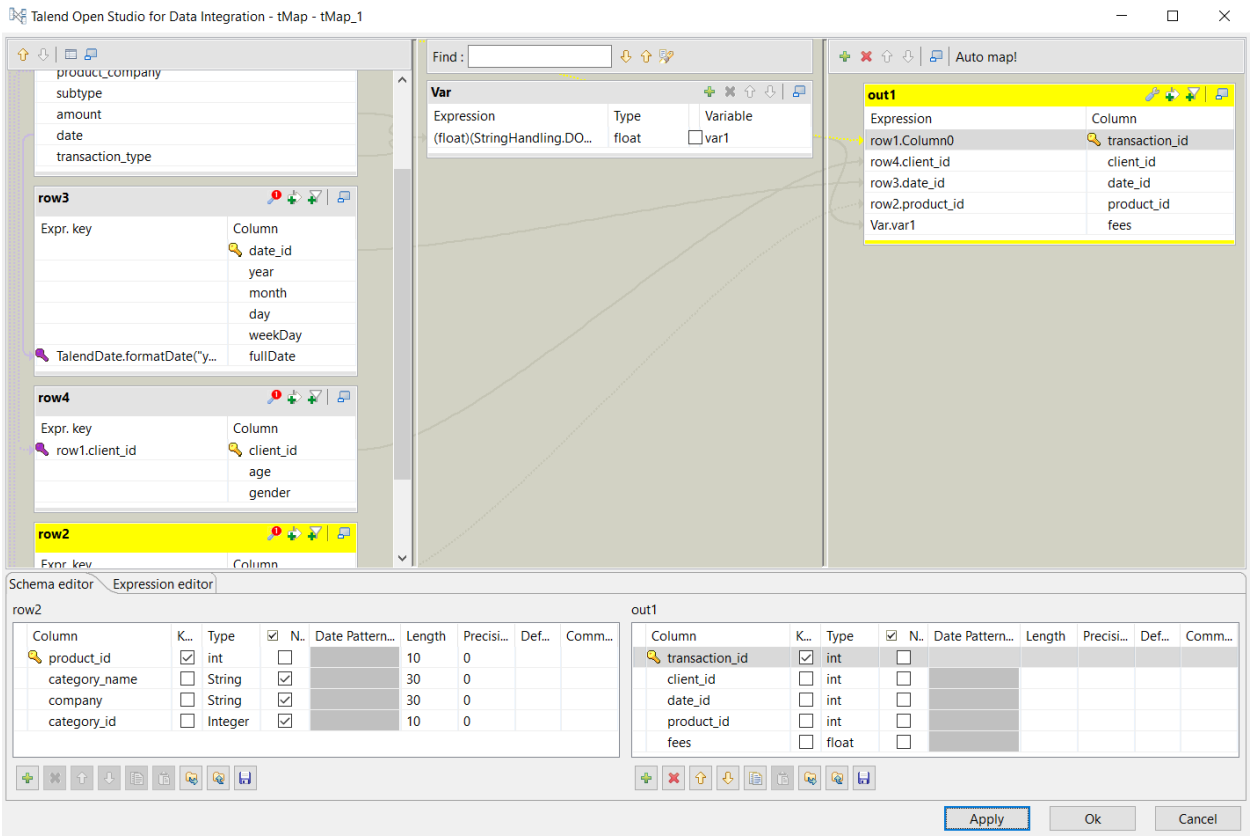
To match date, we format it first using function:

TalendDate.formatDate("yyyy-MM-dd",row1.date).

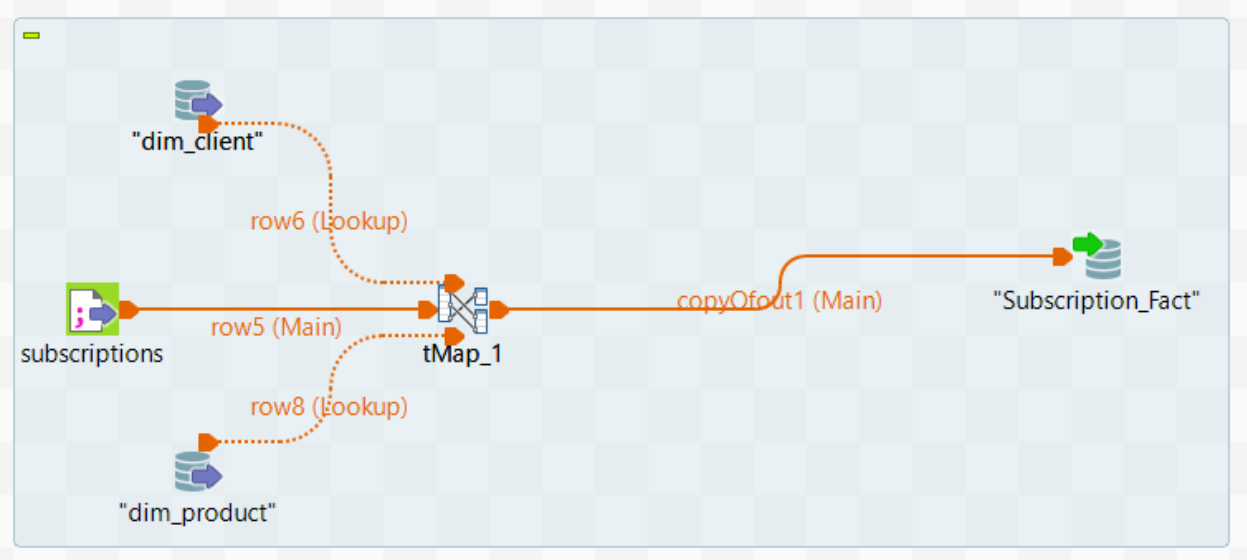
We should verify if company name if they are null then they should be replaced by “Unknown”: Relational.ISNULL(row1.product_company)?"Unknown":row1.product_company

Then we calculate fees for each transaction that debit the balance using the following instruction:

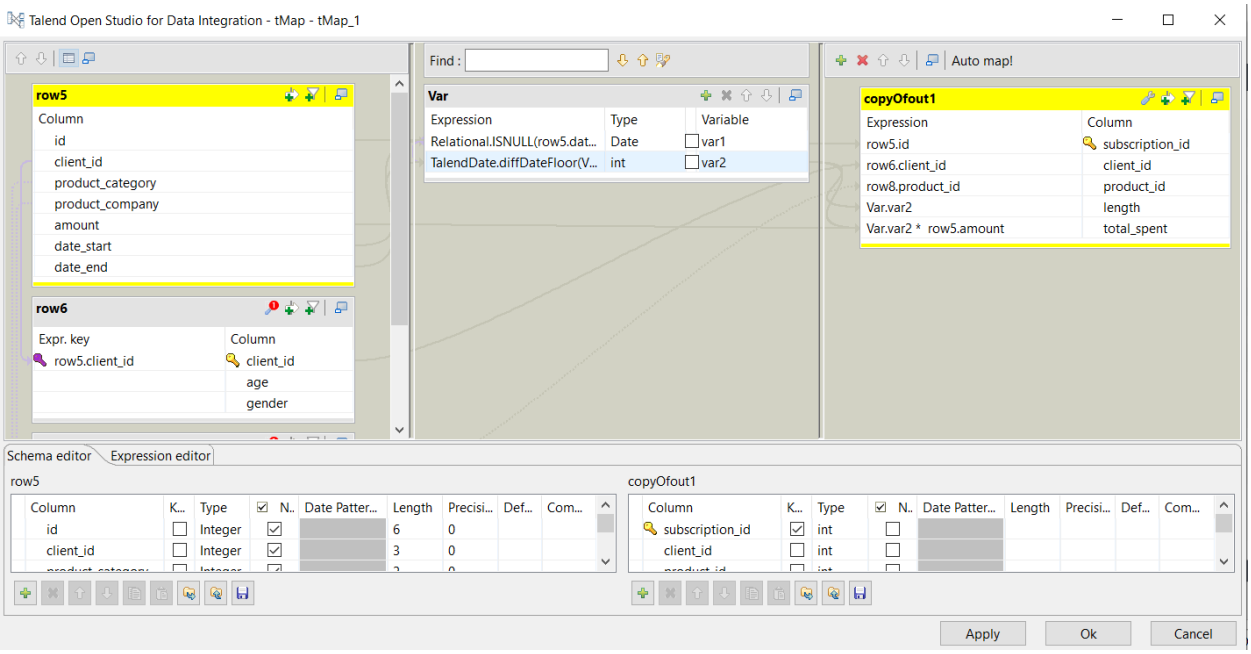
(float)(StringHandling.DOWNCASE(StringHandling.TRIM(row1.transaction_type)).equals("negative")?row1.amount*0.0349+17.99:0)



- Subscription Fact Table:



To create Subscription Fact Table, we do the same steps like the previous fact table creation.



First, we calculate the length for each subscription using instruction:

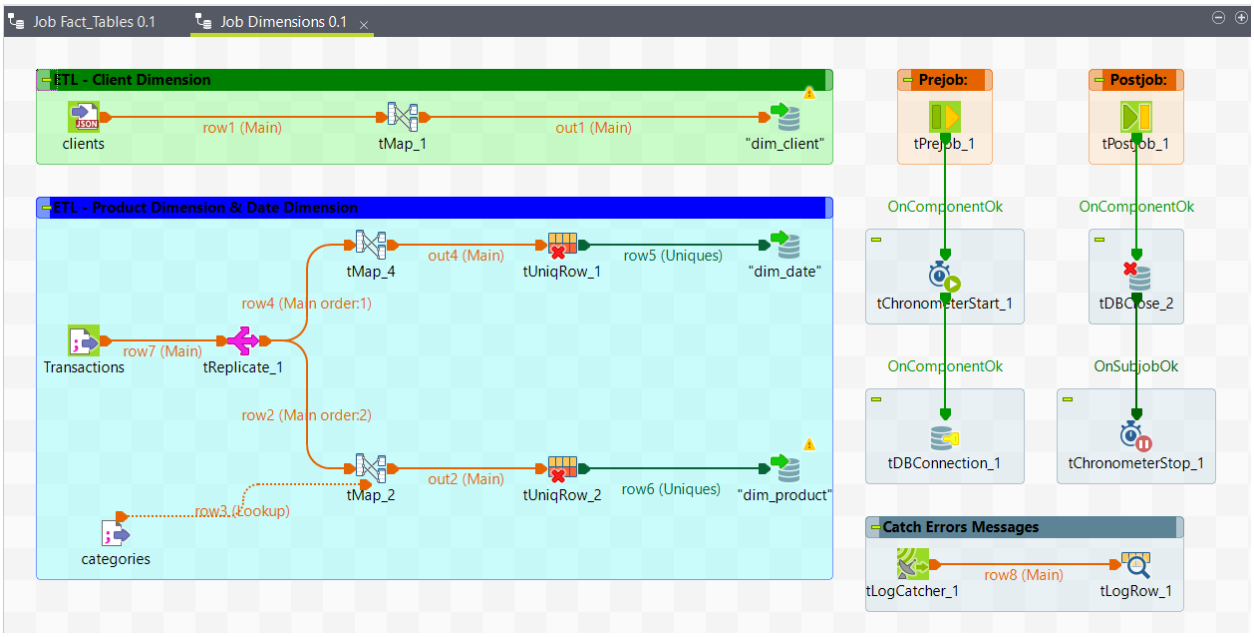
Relational.ISNULL(row5.date_end)?TalendDate.getCurrentDate():row5.date_end
If the end date is null, that means the subscription has not been end yet and should be replaced with the current date.

After that, we calculate the total amount spent on this subscription by multiplying the length by amount: TalendDate.diffDateFloor(Var.var1,row5.date_start,"MM")

After Creating all these tables, we load it in our MySQL database as it shown in the Jobs schemas. For example, the table Transaction_Fact:

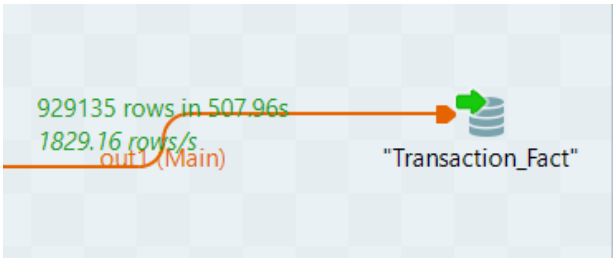
7. Best Practices & Organization:

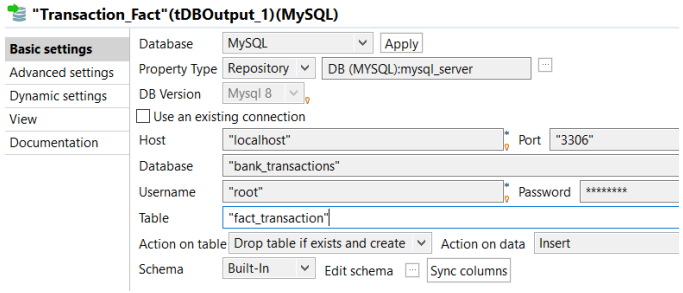
In this part, I am going to organize my project on SubJobs, add some colors and add some useful components that should be in each project.



"tReplicate" allows us to create multiple copies of the same data instead of extracting it each time.

"tPreJob" is a routine that is executed before the main Job starts. preJob routines are used to

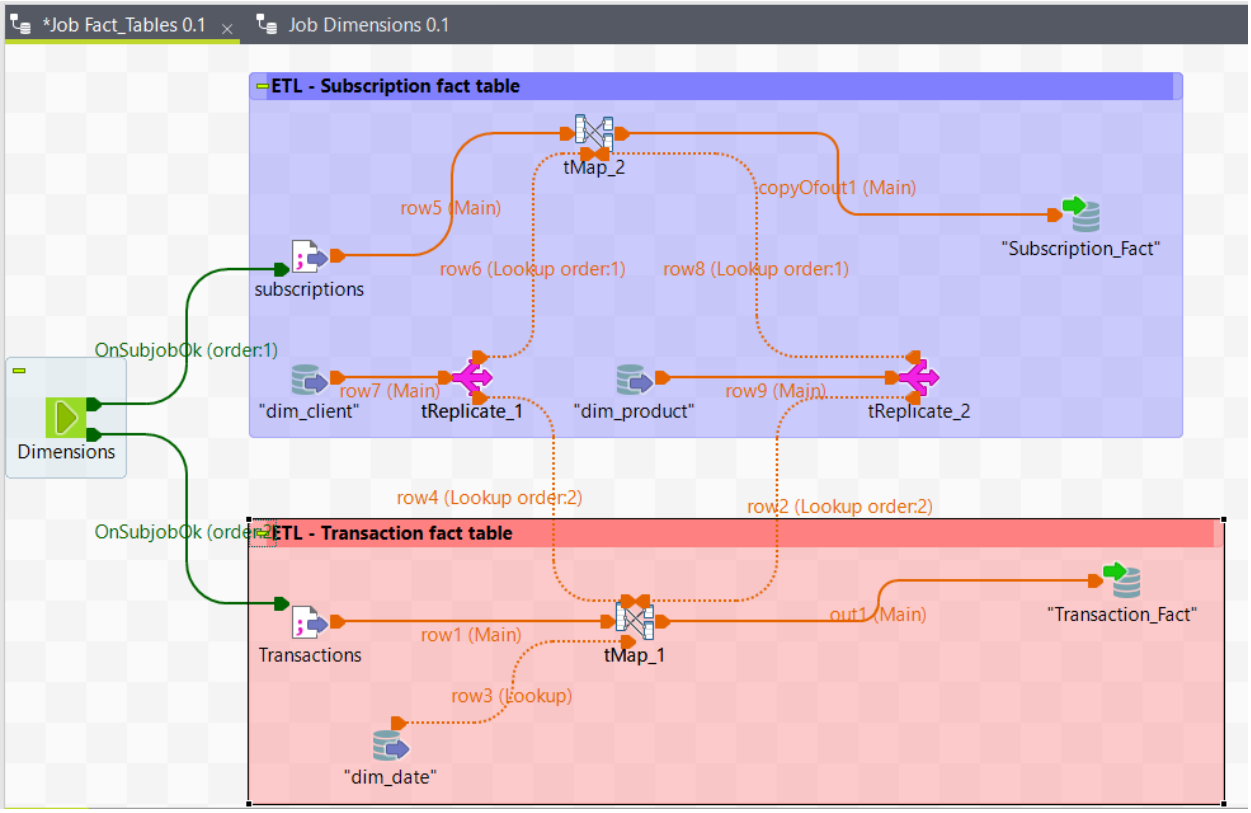




perform setup tasks that need to be done
before the main Job execution, such as in our project ; setting up connections (tDbConnection) or starting the chronometer to calculate the process time of the job.

"tPostJob" is the opposite of "tPreJob", routine that is executed after the main Job finishes. used to perform cleanup tasks that need to be done after the main Job execution, such as closing connections, stopping the chronometer.

“tLogCatcher” component acts as a catch-all for exceptions that occur during the execution of a Job. When an exception is caught, it can be logged and the Job can continue to run, allowing you to capture detailed information about the error and identify the cause of the issue. This information can then be used to troubleshoot and fix the problem, improving the overall reliability and stability of the Job.



With the “tRunJob” component, you can execute another Job as a subprocess within the main Job. This allows you to modularize your data processing logic, breaking down complex Jobs into smaller, reusable parts.

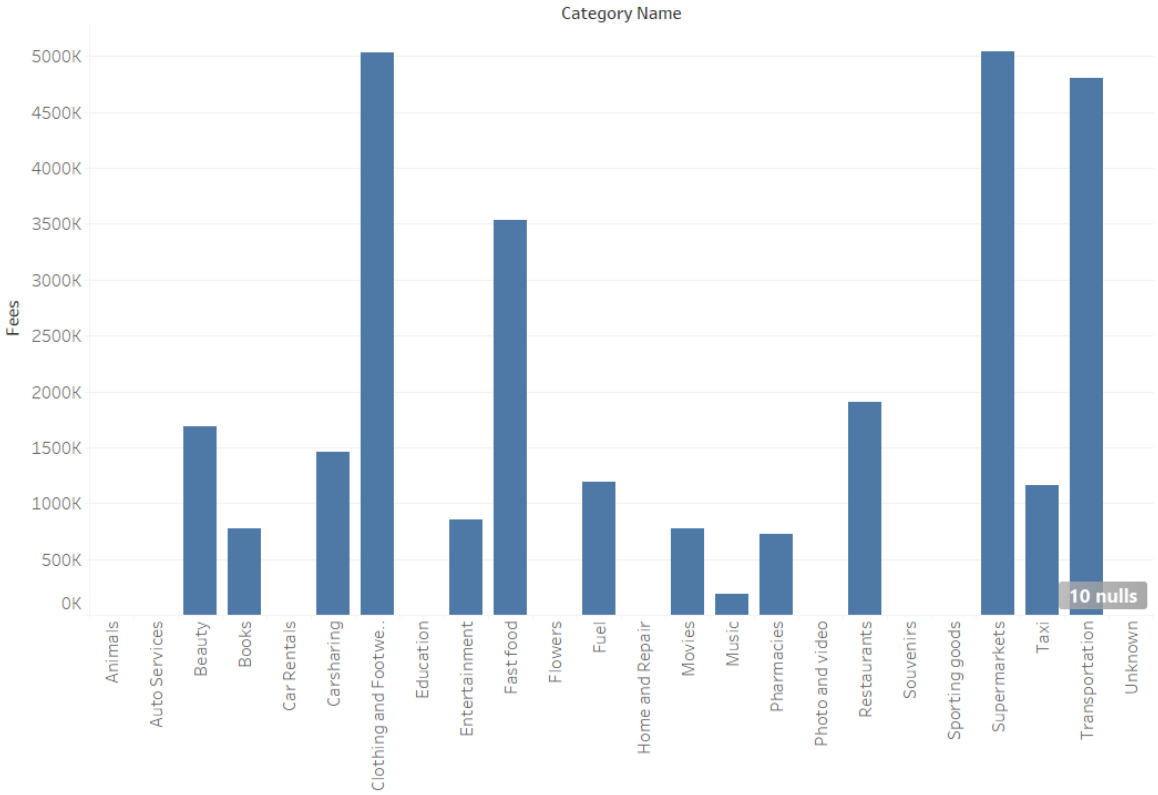
Instead of running separately each job each time, we can run directly “Fact_Tables” job while tRunJob will run “Dimensions” job as first step before running “Fact_Tables”.

III. Data Visualization:

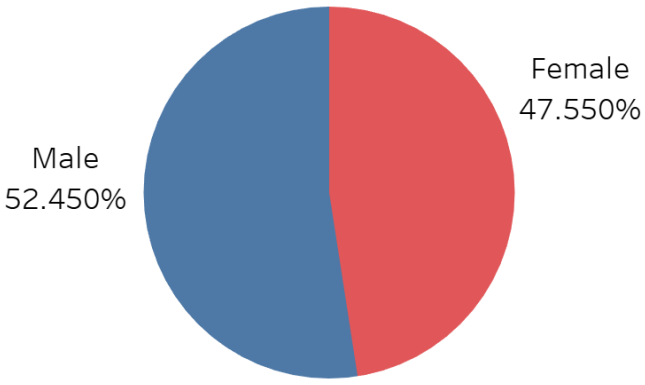
After Creating our Data Warehouse, now we can visualize it using any visualization tool like Power BI, Tableau…

In my case, I used Tableau and we I performed the following graphs:

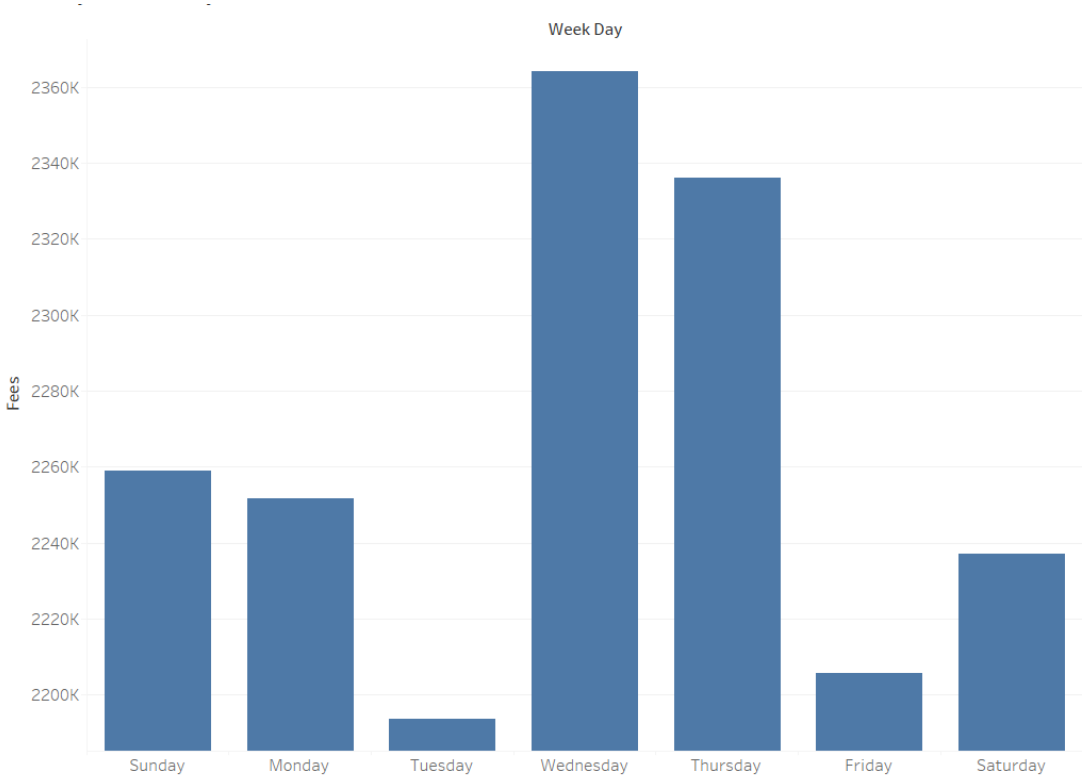
- Fees By Category:



- Fees By Gender:



- Fees By Week Day



We could also visualize sum of subscriptions length by companies, or total spent by group of ages.

We could visualize KPIs by more than one dimension, sorting them, and why not selecting just the top 10.

Conclusion

In conclusion, this data warehouse project has provided valuable insights and hands-on experience in data integration. Through the implementation of a data warehouse, I have learned the importance of data governance and the necessary steps to ensure data quality. The ETL process has taught me the various techniques used to extract, transform and load data from multiple sources into a centralized location. The use of Talend, a powerful ETL tool, has further enhanced my understanding of the capabilities and potential of ETL in solving real-world data integration challenges. Overall, this project has been a valuable learning experience that has provided a strong foundation for future endeavors in data warehousing.