INTEGRATED PROJECT REPORT

Work Done By :

Yasmine Ben Larbi Sarah Ben Yahmed

Ichrak Boulakleka Sayda Bouhoula

Ali Ben Zaied Ibtihelle Ben Yahmed

Work Supervised By :

Mustapha Trabelsi
Manel Khamassi
Ines Mhaya
Abir Mrabet

Table of Contents

Introduction

Launching the La Rose Blanche and ALCO project represents a strategic shift towards maximizing supply chain efficiency and refining product quality prediction within our organizations. By placing a paramount emphasis on the development of user-friendly dashboards, advanced analytics models, and optimization algorithms, we aim to revolutionize our approach to decision-making processes.

In elevating supply chain efficiency, our goal is to streamline operations, reduce lead times, and minimize costs while ensuring the seamless flow of goods and services throughout our networks. Concurrently, enhancing product quality prediction entails leveraging cutting-edge technologies and data-driven insights to anticipate and mitigate potential defects or deviations, thereby safeguarding our reputation and customer satisfaction levels.

Moreover, this initiative signifies our unwavering commitment to strategic decision-making. By harnessing the power of data visualization and predictive analytics, we seek to empower decision-makers with actionable insights, enabling them to make informed choices that drive sustainable growth and competitive advantage.

As we embark on this transformative journey, the successful implementation of these measures holds the promise of significant benefits for both organizations. Not only will it empower decision-makers with real-time visibility and predictive capabilities, but it will also optimize operational processes, enhance resource allocation, and foster innovation across our supply chains.

Ultimately, the culmination of these efforts is poised to propel both La Rose Blanche and ALCO towards long-term growth and heightened customer satisfaction. By embracing innovation and prioritizing strategic decision-making, we are poised to unlock new opportunities, overcome challenges, and create lasting value for our stakeholders.

II. Business Understanding

1. Presentation of the Enterprise

The STPA, also known as the Tunisian Company of Food Products, has carved a niche for itself in the market as a specialized producer of flour and semolina, marketed under the esteemed brand name Sboula. In addition to this, the company enriches its offerings by manufacturing animal feed under the renowned brand Alco. This diversification underscores STPA's commitment to catering to various consumer needs within the food industry.

Now, let's delve into the rich history of the Rose Blanche Group, spanning over a century. For generations, the Rose Blanche Group has remained steadfast in its dedication to delivering top-notch quality products. Starting from the cultivation of cereals, they have meticulously transformed these raw materials into delectable pasta and other cereal-based products. This steadfast commitment to excellence and innovation has enabled them to expand their product portfolio, offering a diverse array of cereal products to satisfy every culinary preference.

Despite their century-long legacy of success, the Rose Blanche Group faces formidable challenges, particularly within their supply chain operations. Issues such as the quality of raw materials, escalating costs, regulatory compliance, discrepancies in truck quantities, overloading of trucks, and prolonged waiting times for loading demand urgent attention. Addressing these challenges is imperative to uphold operational efficiency and uphold the high standards of customer satisfaction that the Rose Blanche Group is renowned for.

2. Problematic

How to improve the reliability of La Rose Blanche company's supply chain and make it more efficient?

3. Decision Makers

- CEO
- Production Manager
- Sales Manager

4. Business Goals

- ***STPA :***

- Enable decision makers to specify precise filtering criteria, such as date, client, and transporter, to obtain targeted and relevant analyses.
- Design an interactive dashboard with a clean, user-friendly interface, clear dynamic charts for optimal user experience, and overview of key performance indicators.
- Develop more in-depth, comparative, and summarizing charts, providing detailed analyses, meaningful comparisons across various data points, and summaries for an overall understanding of the presented information.
- Display evolution rates and other relevant metrics for a comprehensive presentation of data, including the evolution of the difference and CPK and CP indices.

- ***Alco :***

- Predict product quality:
    - Identify components of a high-quality product
    - Identify the best component for each product
    - Segment products based on components
- Visualize components by product:
    - Compare products based on their components
    - Identify top-selling products
    - Identify the best-selling product group

5. Technical Goals

6. STPA :

- Develop and deploy a regression model incorporating maximum values, minimum values, and filling duration as input parameters to precisely forecast Process Capability (CP) Process Capability Index (CPK).
- Apply a clustering algorithm (K-Means) on sales data to group products based on sales patterns, identifying clusters with the highest sales volume as the best-selling product groups.

- Implement dynamic ordering recommendations based on real-time data, historical trends, and external factors to optimize order quantities and schedules.
- Develop a classification model to categorize carriers based on their filling time.
- Display evolution rates and other relevant metrics for a comprehensive presentation of data, including the evolution of the difference and CPK and CP indices.

- ***Alco :***

- Predict product quality :

K-Nearest Neighbors (KNN) algorithm:

Develop a KNN model to predict product quality.

Identify the optimal parameters for the model.

Evaluate the model's performance.

Support Vector Machine (SVM) algorithm:

Develop an SVM model to predict product quality.

Identify the optimal parameters for the model.

Evaluate the model's performance.

- Identify key product components:

Analysis of variance (ANOVA) algorithm:

Identify components significantly impacting product quality.

Determine the contribution of each component to product quality.

Feature selection algorithm:

Identify the most important components for product quality.

Reduce data dimensionality.

- Optimize production:

Linear programming algorithm:

- Determine the optimal combination of components to maximize product quality.

- Define production constraints.
- Find the optimal solution.

6. Key Performance Indicator

- **STPA**

- Deviation/Qx = Sum ([Net Weight Scale/Qx]) - Sum ([BL Quantity/Qx]) -Sum ([Empty Bag Weight])
- Filling duration = [Exit/h] - [Entry/h] / [Net Weight Scale]
- Number of units sold = Sum (BL quantity/QX)
- Product distribution ratio = (Quantity of a specific product distributed / Total quantity distributed) * 100
- Quantity sold evolution rate = BL quantity in month x of year n - BL quantity in month x of year n-1 / BL quantity of year n-1 in month x
- Carrier Loading Ratio = number of distinct delivery notes / Total quantity delivered
- Process Capability CP = min ((USL - LSL) / (6 * filling duration))
- Process Capability Index CPK = min (((USL - average_filling_duration) / (3 * filling_duration)), ((average_filling_duration - LSL) / (3 * filling_duration)))

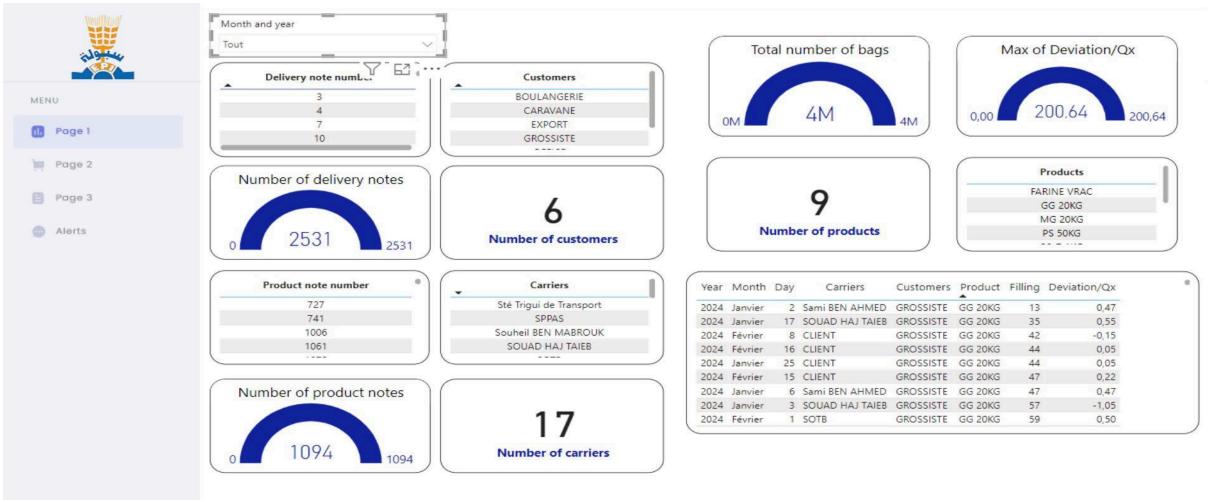
- **Alco :**

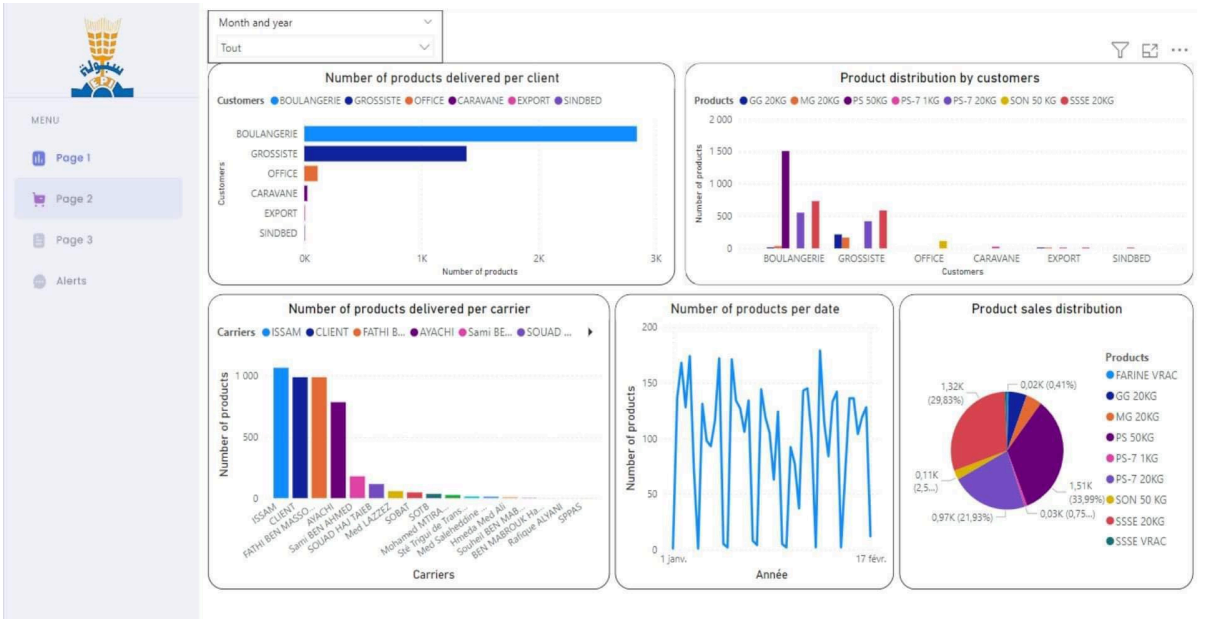
- Average X content = (Sum of X values for all products) / Number of products : Average X Content: This is the average quantity of X in each product manufactured by ALCO. It's calculated by summing up all X values for all products and then dividing that sum by the total number of products
- Raw material cost = Unit price of raw material : Raw Material Cost: This represents the unit price of raw materials used in manufacturing composite foods. It helps monitor production costs.
- cp = (USL - LSL) / (6 * ecart_type_humidite) : Cp (Process Capability Index): Cp measures the production process's capability to maintain product quality, typically in relation to the variability of humidity compared to specifications. A higher Cp indicates a more stable process.
- cpk = min(((USL - moyenne_humidite) / (3 * ecart_type_humidite)), ((moyenne_humidite - LSL) / (3 * ecart_type_humidite))): Cpk (Process Performance Index): Cpk evaluates both the process's capability to meet specifications and its

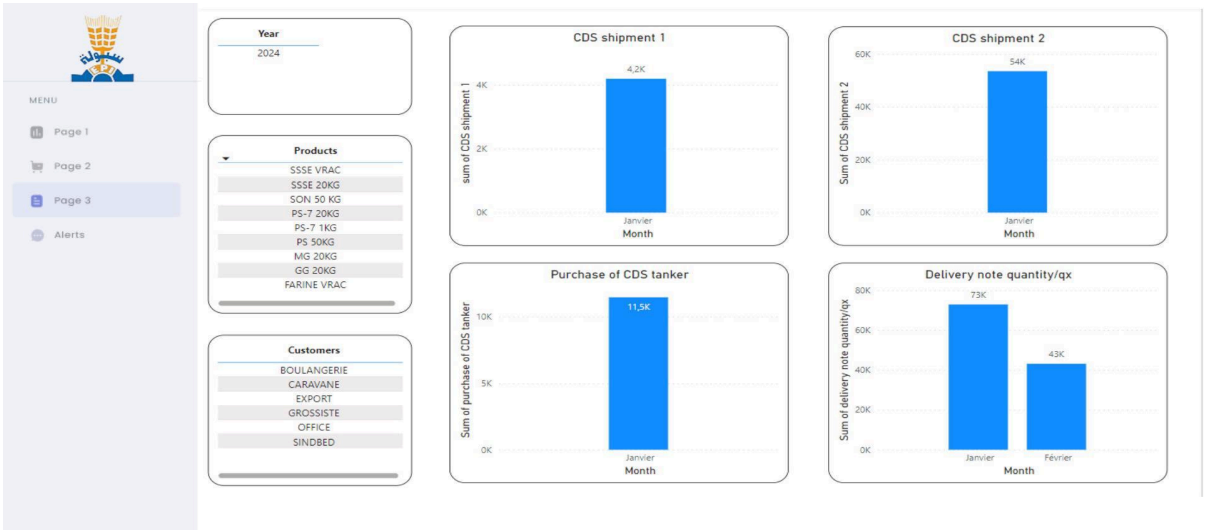
ability to approach the specified target. A higher Cpk suggests better quality control of the product.

7. *Dashboard Mockups*

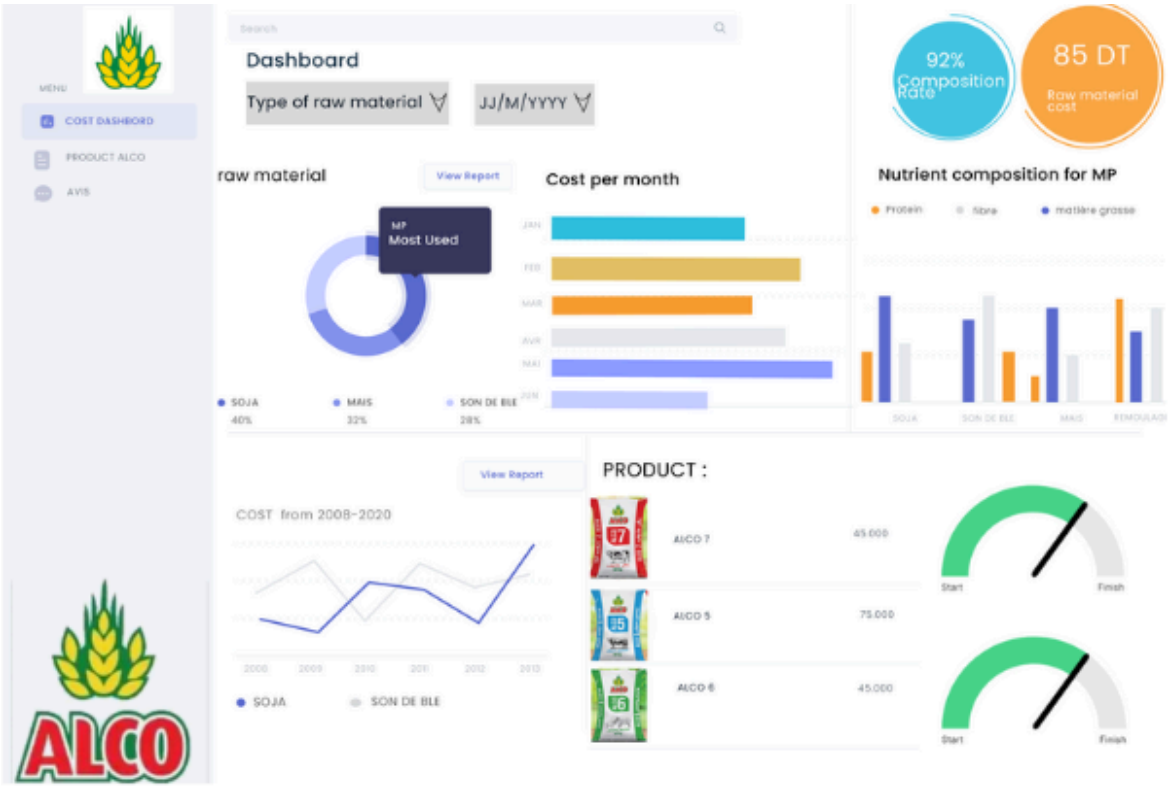
- ***STPA***






MENU
Page 1
Page 2
Page 3
Alerts
Year
2024
Products
SSSE VRAC
SSSE 20KG
SON 50 KG
PS-7 20KG
PS-7 1KG
PS 50KG
MG 20KG
GG 20KG
FARINE VRAC
Customers
BOULANGERIE
CARAVANE
EXPORT
GROSSISTE
OFFICE
SINDBED
CDS shipment 1
sum of CDS shipment 1
4,2K
Janvier
Month
CDS shipment 2
Sum of CDS shipment 2
54K
Janvier
Month
Purchase of CDS tanker
Sum of purchase of CDS tanker
11,5K
Janvier
Month
Delivery note quantity/qx
Sum of delivery note quantity/qx
73K
43K
Janvier
Février
Month

- ***Alco :***


MENU
COST DASHBORD
PRODUCT ALCO
AVIS
Dashboard
Type of raw material
JJ/M/YYYY
92% Composition Rate
85 DT Raw material cost
raw material
View Report
MP Most Used
Cost per month
Nutrient composition for MP
Protein
fibre
matière grasse
SOJA 40%
MAIS 32%
SON DE BLE 28%
View Report
PRODUCT :
COST from 2008-2020
ALCO 7
45.000
ALCO 5
75.000
ALCO 6
45.000
SOJA
SON DE BLE
ALCO


ALCO
MENU
Dashboard
Composant ALCO
Dashboard
THIS YEAR
46
number of products with problems
4
number of groups
125
number of products
CPE CP
protein humidity durete grasse starch
Component rates for each product
starch
protein
humidity
ALCO7 ALCO8 ALCO6 ALCO5 P0V2 C0V2 0
Problem by product
high fine rate
durability below target
non-compliant protein
Distribution of problems
starch below target
starch above target
non-compliant protein
high fine rate
compliance levels
no-compliant
35%
65%
compliant

III. Environment and Methodology

1. Software and Techniques Used

Technique/Software	***Definition***
	Excel is a widely used spreadsheet software renowned for its versatility and user-friendly interface. In our project, we employed Excel as a pivotal tool for managing and analyzing our data. Its intuitive features enabled us to organize large datasets efficiently, perform calculations seamlessly, and generate insightful visualizations. By leveraging Excel's capabilities, we streamlined our data processing workflows, facilitating collaboration and driving informed decision-making.
	In our project, we utilized Visual Paradigm to create visual representations of our software system. This tool helped us design and model various aspects of our project, enabling us to communicate complex ideas in a clear and concise manner. With Visual Paradigm, we were able to streamline the development process and ensure alignment with project requirements.
	Talend is a powerful data integration software that played a crucial role in our project. With its intuitive interface and robust features, Talend streamlined our data management processes. We utilized Talend to extract, transform, and load data from diverse sources, ensuring accuracy and efficiency throughout. Its automation capabilities and scalability made it indispensable for handling large datasets and achieving project goals seamlessly.

	Power BI is a dynamic business analytics tool that significantly contributed to our project's success. With its intuitive interface and powerful visualization capabilities, Power BI allowed us to create compelling dashboards and reports from our data. We leveraged Power BI to gain valuable insights, monitor key metrics, and make informed decisions. Its user-friendly features and seamless integration with various data sources made it an invaluable asset for our project, empowering us to drive actionable insights and enhance decision-making processes effortlessly.
	Angular is a popular JavaScript framework that played a vital role in our project's frontend development. With its robust architecture and extensive features, Angular enabled us to create dynamic and responsive user interfaces. We leveraged Angular's component-based structure and two-way data binding to build interactive and scalable web applications. Its modular design and dependency injection made code organization and maintenance straightforward. Angular proved to be an invaluable tool for enhancing user experience and accelerating frontend development in our project.
	Python is a versatile programming language that served as a cornerstone in our project's development. With its simple syntax and vast ecosystem of libraries and frameworks, Python facilitated rapid prototyping and streamlined development processes. We leveraged Python for various tasks, including data analysis, machine learning, and backend development. Its readability and flexibility made it easy to collaborate on code and iterate quickly. Python's extensive community support and rich documentation were invaluable resources that contributed to the success of our project.

	Power Apps played a pivotal role in our project by empowering us to build custom applications tailored to our specific needs. With its intuitive interface and drag-and-drop functionality, Power Apps enabled us to create robust and user-friendly applications without the need for extensive coding knowledge. Leveraging Power Apps, we were able to accelerate the development process and deliver solutions that seamlessly integrated with our existing systems. Its versatility and scalability made it a valuable tool for enhancing productivity and meeting the diverse requirements of our project.
	Power Automate was instrumental in automating workflows in our project. With its user-friendly interface and extensive range of connectors, we streamlined processes, eliminated manual tasks, and improved overall efficiency. Leveraging Power Automate's automation capabilities, we enhanced productivity and achieved seamless integration with various applications and services, driving operational excellence in our project.
	Anaconda is a comprehensive platform for data science and machine learning that played a pivotal role in our project. With its bundled collection of Python packages and environments management tools, Anaconda simplified the setup and configuration of our development environment. We utilized Anaconda to manage dependencies, create virtual environments, and install packages seamlessly. Its integrated development environment (IDE) and Jupyter Notebook support provided a user-friendly interface for data exploration, analysis, and model development. Anaconda's robust capabilities and ease of use significantly accelerated our workflow and enhanced productivity in our project.

2. Methodology Used

The GIMSI method, short for Goal, Information, Method, Solutions, Implementation, is a systematic approach used for problem-solving and decision-making in various domains. Each step in the GIMSI method corresponds to a specific phase in the problem-solving process, guiding practitioners through a structured and organized framework.

In the first step, Goal, the focus is on clearly defining the objectives or goals that need to be achieved. This involves identifying the desired outcomes and understanding the purpose of addressing the problem at hand. Next, in the Information phase, relevant data and information are gathered and analyzed to gain insights into the underlying factors contributing to the problem.

Following the Information phase is the Method phase, where various methodologies, techniques, and tools are employed to analyze the data and develop potential solutions. This phase emphasizes critical thinking and creative problem-solving to explore different approaches and identify the most effective strategies.

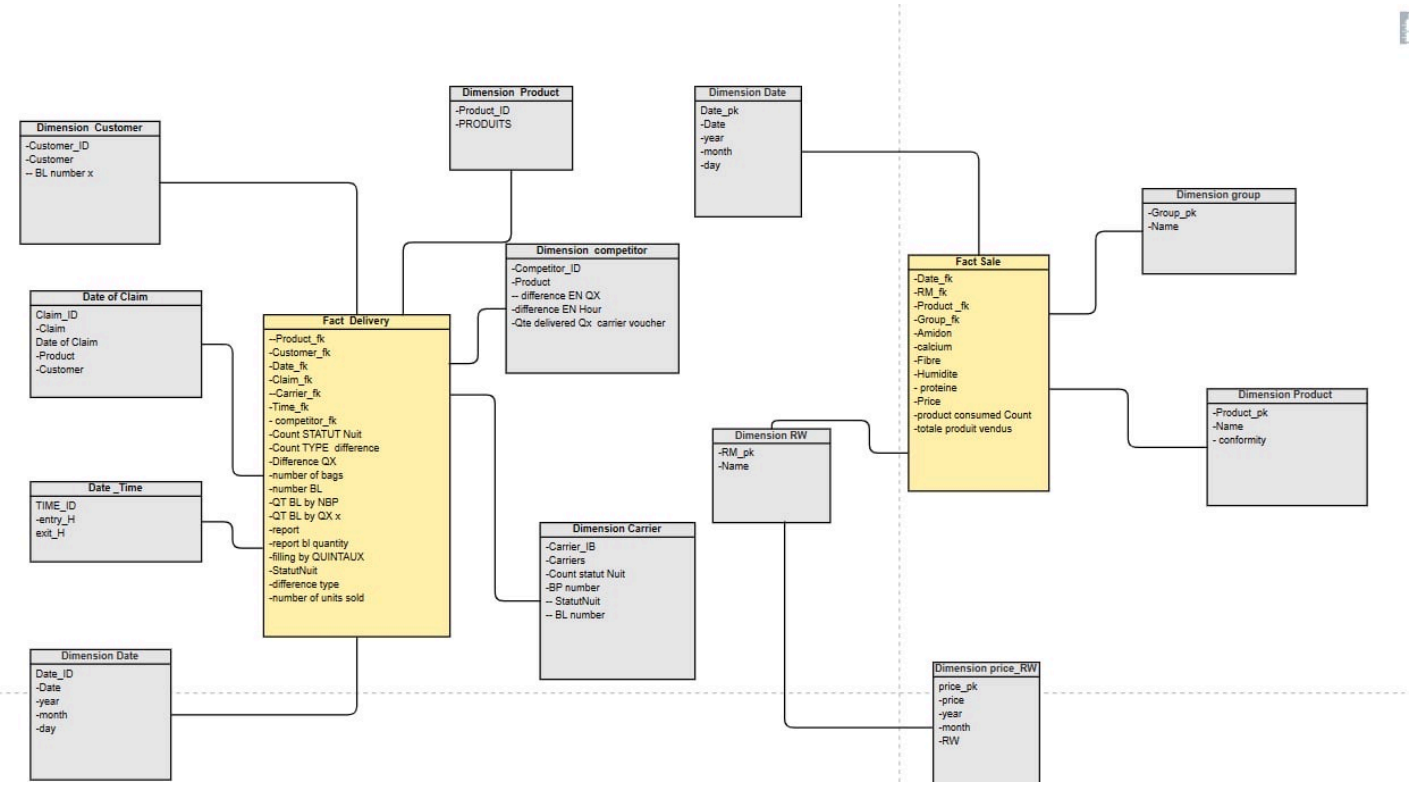
Once potential solutions have been generated, the Solutions phase involves evaluating and selecting the best course of action based on predefined criteria and constraints. This step requires careful consideration of the feasibility, practicality, and impact of each proposed solution.

After selecting the optimal solution, the Implementation phase focuses on putting the chosen solution into action. This involves developing an action plan, allocating resources, and executing the implementation strategy effectively.

In our project, we adopted the GIMSI method as a structured approach to problem-solving and decision-making. By following this systematic framework, we were able to define clear objectives, gather relevant data, analyze various methodologies, evaluate potential solutions, and implement effective strategies to address the challenges encountered.

Using the GIMSI method in our project enabled us to approach problem-solving in a methodical and organized manner, ultimately leading to more informed decisions and successful outcomes. Incorporating this approach into our project report ensures transparency and clarity in documenting our problem-solving process and the rationale behind our decisions.

3. Data Warehouse



In our project, we have implemented two distinct data warehousing solutions tailored to the specific needs of STPA and ALCO. Firstly, for STPA, we have adopted a star schema design. This schema architecture revolves around a central fact table, "fact-delivery," capturing delivery-related data. This fact table is intricately linked to multiple dimension tables, including "product," "customer," and "time," enabling comprehensive analysis from various perspectives. This star schema configuration simplifies query execution and enhances performance by minimizing joins and optimizing data retrieval.

Secondly, for ALCO, we have employed a snowflake schema approach. In this schema, the primary fact table, "fact_sale," records sales-related information. This fact table is associated with dimension tables representing entities such as "product," "group," and "date." Additionally, a sub-dimension further refines the granularity of certain attributes, providing a more detailed analysis of sales data. The snowflake schema's normalized structure reduces data redundancy and improves data integrity, making it ideal for scenarios where data integrity and storage efficiency are paramount.

By utilizing star and snowflake schema designs for STPA and ALCO, respectively, we ensure that each data warehousing solution is optimized to meet the unique requirements of its corresponding organization. These schema designs lay the foundation for efficient data storage, retrieval, and analysis, empowering decision-makers with actionable insights to drive business growth and performance.

IV. Project Implementation

1. Our Source of Data

1.1 Internal

In our project, the primary source of data originates from the esteemed enterprise, La Rose Blanche. As a leading Tunisian company specializing in the production of flour, semolina, and animal feed, La Rose Blanche serves as the principal supplier of raw data for our analytical endeavors. Leveraging its extensive operational activities and rich dataset, we gather a comprehensive range of data points spanning various aspects of the production, distribution, and sales processes.

Through close collaboration with La Rose Blanche, we gain access to crucial datasets encompassing production volumes, sales transactions, inventory levels, and logistical operations. This data serves as the cornerstone of our analysis and decision-making processes, enabling us to derive actionable insights and drive strategic initiatives.

By tapping into the wealth of data provided by La Rose Blanche, we are able to uncover hidden patterns, trends, and correlations that inform our business strategies and operational optimizations. Moreover, our partnership with La Rose Blanche underscores our commitment to leveraging real-world data to drive innovation and excellence in our project outcomes.

- ***STPA***

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Date	ansporteu	Produits	nbre/qx	sac unitaire	Clients	N° BP	Entrée/h	Sortie/h	Net Bascu	N° BL_x	Qt BL/qx_x	Nbre sacs	Poids sacs vide	Envoi CDS 1
03/01/2024		SSSE VRAC										0	0	256
01/01/2024		SSSE VRAC										0	0	501,88
02/01/2024		SSSE VRAC										0	0	750,67
												0	0	
												0	0	
11/01/2024	ISSAM	PS 50KG	2	0,0008	BOULANG	2851			55,4	1241	20	40	0,032	
11/01/2024	ISSAM	PS 50KG	2	0,0008	BOULANG	2851				1240	10	20	0,016	
11/01/2024	ISSAM	PS-7 20KG	5	0,0006	BOULANG	2851				1240	3	15	0,009	
11/01/2024	ISSAM	PS 50KG	2	0,0008	BOULANG	2851				1239	20	40	0,032	
11/01/2024	ISSAM	PS-7 20KG	5	0,0006	BOULANG	2851				1239	2	10	0,006	
11/01/2024	CLIENT	PS-7 20KG	5	0,0006	GROSSISTE	2843			10	1248	5	25	0,015	
11/01/2024	CLIENT	SSSE 20KG	5	0,0006	GROSSISTE	2843				1248	5	25	0,015	
11/01/2024	AYACHI	PS 50KG	2	0,0008	BOULANG	2824			157	1232	20	40	0,032	
11/01/2024	AYACHI	SSSE 20KG	5	0,0006	BOULANG	2824				1232	5	25	0,015	
11/01/2024	AYACHI	PS 50KG	2	0,0008	BOULANG	2824				1230	30	60	0,048	
11/01/2024	AYACHI	SSSE 20KG	5	0,0006	BOULANG	2824				1230	5	25	0,015	
11/01/2024	AYACHI	PS 50KG	2	0,0008	BOULANG	2824				1229	20	40	0,032	
11/01/2024	AYACHI	PS-7 20KG	5	0,0006	BOULANG	2824				1229	3	15	0,009	
11/01/2024	AYACHI	SSSE 20KG	5	0,0008	BOULANG	2824				1229	10	50	0,04	
11/01/2024	AYACHI	SSSE 20KG	5	0,0006	BOULANG	2824				1228	1	5	0,003	
11/01/2024	AYACHI	PS 50KG	2	0,0008	BOULANG	2824				1227	40	80	0,064	
11/01/2024	AYACHI	PS 50KG	2	0,0008	BOULANG	2824				1231	10	20	0,016	
11/01/2024	AYACHI	PS-7 20KG	5	0,0006	BOULANG	2824				1231	2	10	0,006	
11/01/2024	AYACHI	SSSE 20KG	5	0,0006	BOULANG	2824				1231	10	50	0,03	
11/01/2024	SOTB	SON 50 KG	2	0,0008	OFFICE	2823			301	1251	300	600	0,48	
11/01/2024	CLIENT	PS-7 20KG	5	0,0006	GROSSISTE	2816			50,4	1221	30	150	0,09	
11/01/2024	CLIENT	SSSE 20KG	5	0,0006	GROSSISTE	2816				1221	20	100	0,06	
11/01/2024	ISSAM	PS 50KG	2	0,0008	BOULANG	2814			75,6	1238	20	40	0,032	
11/01/2024	ISSAM	PS-7 20KG	5	0,0006	BOULANG	2814				1237	10	50	0,03	

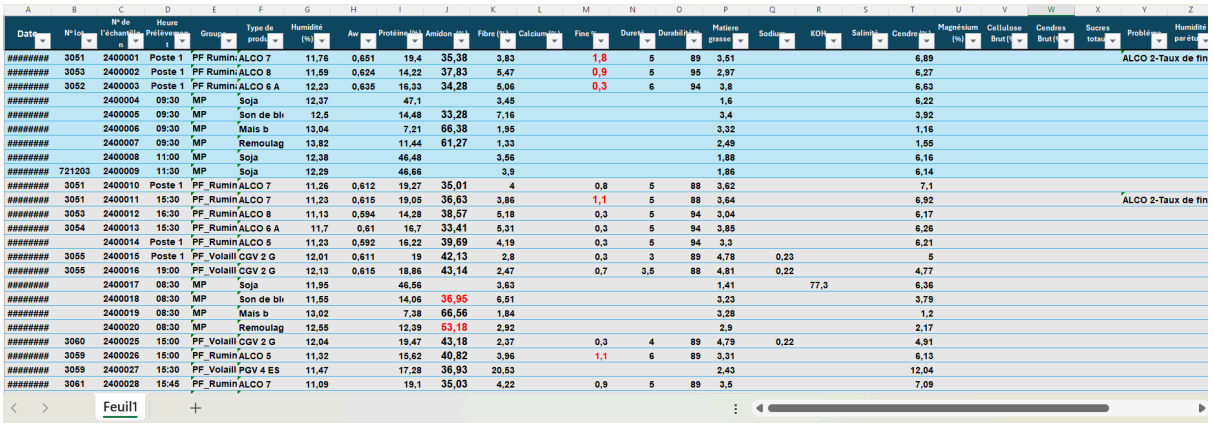
Sheet1

P	Q	R	S	T	U	V	W	X	Y
Envoi CDS 2	Achat Citerne CDS	Ecart/Qx	StatutNuit	Count_Statut_Nuit	mplisssage_par_quintal	Type_Ecart	Count_Type_Ecart	NB Entree Transporteur	Qt BL/NI
1572,75			0						
2306,58			0						
2138,74			0						
			0						
			0						
		0,305	0			Positif	127	4	
		0,305	0			Positif	127	4	
		0,305	0			Positif	127	4	
		0,305	0			Positif	127	4	
		0,305	0			Positif	127	4	
		-0,03	0			Négatif	168	14	
		-0,03	0			Négatif	168	14	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,69	0			Positif	94	3	
		0,52	0			Positif	17	1	
		0,25	0			Positif	328	14	
		0,25	0			Positif	328	14	
		0,447	0			Positif	127	4	
		0,447	0			Positif	127	4	

Sheet1

W	X	Y	Z	AA	A
Count_Type_Ecart	NB Entree Transporteur	Qt BL/NBP	Nombre_BL	Rapport	
127	4	55	3	0,054545	
127	4	55	3	0,054545	
127	4	55	3	0,054545	
127	4	55	3	0,054545	
127	4	55	3	0,054545	
168	14	10	1	0,1	
168	14	10	1	0,1	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
94	3	156	6	0,038462	
17	1	300	1	0,003333	
328	14	50	1	0,02	
328	14	50	1	0,02	
127	4	75	6	0,08	
127	4	75	6	0,08	

- ***Alco***



Date	N° lot	N° de l'échantillon	Heure Prélèvement	Groupe	Type de produ	Humidité (%)	Aw	Protéine (%)	Amidon	Fibre (%	Calcium	Fine	Dureté	Durabilité	Matiere grasse	Sodium	KOH	Salinité	Cendre	Magnésium (%)	Cellulose Brut (%	Cendres Brut (%	Sucres totaux	Problème	Humidité par étu
########	3051	2400001	Poste 1	PF Rumin	ALCO 7	11,76	0,651	19,4	35,38	3,83		1,8	5	89	3,51				6,89					ALCO 2-Taux de fin	
########	3053	2400002	Poste 1	PF Rumin	ALCO 8	11,59	0,624	14,22	37,83	5,47		0,9	5	95	2,97				6,27						
########	3052	2400003	Poste 1	PF Rumin	ALCO 6 A	12,23	0,635	16,33	34,28	5,06		0,3	6	94	3,8				6,63						
########		2400004	09:30	MP	Soja	12,37		47,1		3,45					1,6				6,22						
########		2400005	09:30	MP	Son de blé	12,5		14,48	33,28	7,16					3,4				3,92						
########		2400006	09:30	MP	Mais b	13,04		7,21	66,38	1,95					3,32				1,16						
########		2400007	09:30	MP	Remoulag	13,82		11,44	61,27	1,33					2,49				1,55						
########		2400008	11:00	MP	Soja	12,38		46,48		3,56					1,88				6,16						
########	721203	2400009	11:30	MP	Soja	12,29		46,66		3,9					1,86				6,14						
########	3051	2400010	Poste 1	PF_Rumin	ALCO 7	11,26	0,612	19,27	35,01	4		0,8	5	88	3,62				7,1						
########	3051	2400011	15:30	PF_Rumin	ALCO 7	11,23	0,615	19,05	36,63	3,86		1,1	5	88	3,64				6,92					ALCO 2-Taux de fin	
########	3053	2400012	16:30	PF_Rumin	ALCO 8	11,13	0,594	14,28	38,57	5,18		0,3	5	94	3,04				6,17						
########	3054	2400013	15:30	PF_Rumin	ALCO 6 A	11,7	0,61	16,7	33,41	5,31		0,3	5	94	3,85				6,26						
########		2400014	Poste 1	PF_Rumin	ALCO 5	11,23	0,592	16,22	39,69	4,19		0,3	5	94	3,3				6,21						
########	3055	2400015	Poste 1	PF_Volaill	CGV 2 G	12,01	0,611	19	42,13	2,8		0,3	3	89	4,78	0,23			5						
########	3055	2400016	19:00	PF_Volaill	CGV 2 G	12,13	0,615	18,86	43,14	2,47		0,7	3,5	88	4,81	0,22			4,77						
########		2400017	08:30	MP	Soja	11,95		46,56		3,63					1,41		77,3		6,36						
########		2400018	08:30	MP	Son de blé	11,55		14,06	36,95	6,51					3,23				3,79						
########		2400019	08:30	MP	Mais b	13,02		7,38	66,56	1,84					3,28				1,2						
########		2400020	08:30	MP	Remoulag	12,55		12,39	53,18	2,92					2,9				2,17						
########	3060	2400025	15:00	PF_Volaill	CGV 2 G	12,04		19,47	43,18	2,37		0,3	4	89	4,79	0,22			4,91						
########	3059	2400026	15:00	PF_Rumin	ALCO 5	11,32		15,62	40,82	3,96		1,1	6	89	3,31				6,13						
########	3059	2400027	15:30	PF_Volaill	PGV 4 ES	11,47		17,28	36,93	20,53					2,43				12,04						
########	3061	2400028	15:45	PF_Rumin	ALCO 7	11,09		19,1	35,03	4,22		0,9	5	89	3,5				7,09						

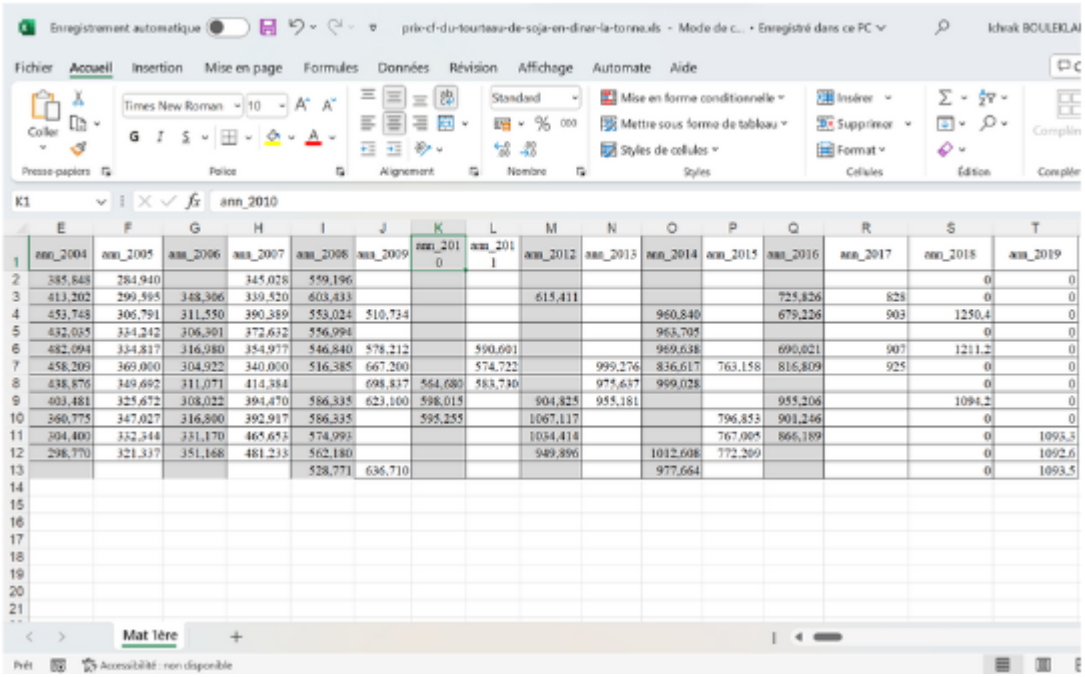
Feuil1

1.2 External

In addition to the data sourced internally from La Rose Blanche, our project also incorporates valuable external data obtained from the Ministry. Collaborating with the Ministry has provided us access to supplementary datasets that enrich our analysis and broaden the scope of our insights. These external datasets offer a broader perspective on industry trends, regulatory changes, and market dynamics, complementing the internal data from La Rose Blanche.

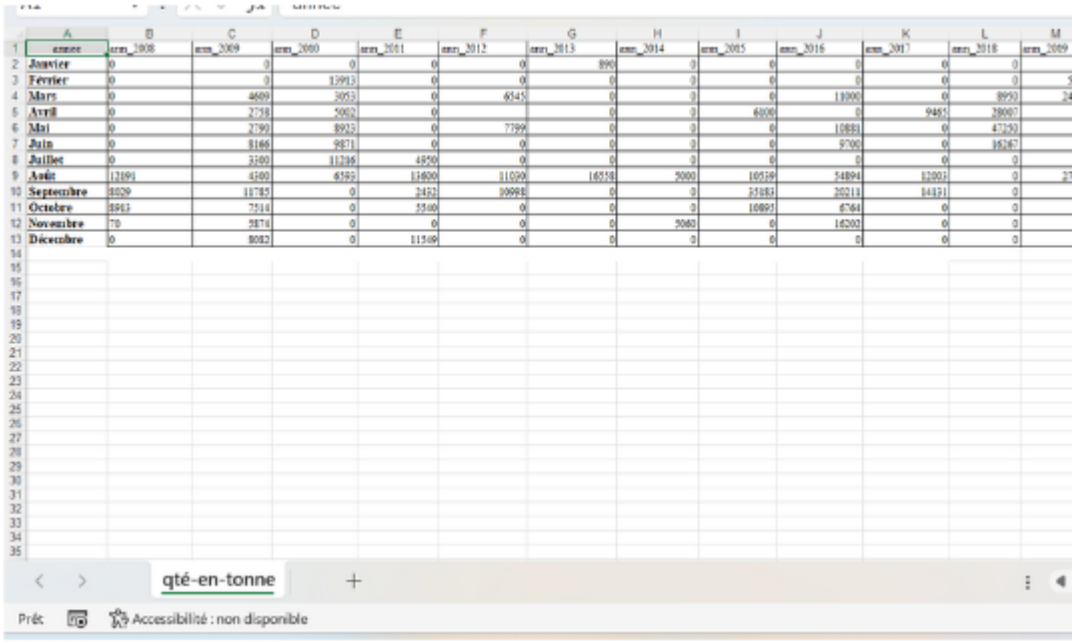
By incorporating external data from authoritative sources such as the Ministry, we enhance the robustness and reliability of our analyses, ensuring a more comprehensive understanding of the business landscape. This collaborative approach not only enriches our project outcomes but also strengthens the validity and relevance of our findings.

The integration of external data from the Ministry underscores our commitment to leveraging diverse sources of information to inform our decision-making processes and drive actionable insights. This collaborative effort exemplifies the synergy achieved through cross-sector partnerships and highlights the value of accessing a wide array of data sources to achieve our project objectives.



	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	ann_2004	ann_2005	ann_2006	ann_2007	ann_2008	ann_2009	ann_2010	ann_2011	ann_2012	ann_2013	ann_2014	ann_2015	ann_2016	ann_2017	ann_2018	ann_2019
2	385,848	284,940		345,028	559,196										0	0
3	413,202	299,595	348,306	339,520	603,433				615,411				725,826	828	0	0
4	453,748	306,791	311,550	390,389	553,024	510,734					960,840		679,226	903	1250,4	0
5	432,035	334,242	306,301	372,632	556,994						963,705				0	0
6	482,094	334,817	316,980	354,977	546,840	578,212		590,601			969,638		690,021	907	1211,2	0
7	458,209	369,000	304,922	340,000	516,385	667,200		574,722		999,276	836,617	763,158	816,809	925	0	0
8	438,876	349,692	311,071	414,384		698,837	564,680	583,730		975,637	999,028				0	0
9	403,481	325,672	308,022	394,470	586,335	623,100	598,015		904,825	955,181			955,206		1094,2	0
10	360,775	347,027	316,800	392,917	586,335		595,255		1067,117			796,853	901,246		0	0
11	304,400	332,344	331,170	465,653	574,993				1034,414			767,005	866,189		0	1093,3
12	298,770	321,337	351,168	481,233	562,180				949,896		1012,608	772,209			0	1092,6
13					528,771	636,710					977,664				0	1093,5

Mat 1ère



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	année	ann_2008	ann_2009	ann_2010	ann_2011	ann_2012	ann_2013	ann_2014	ann_2015	ann_2016	ann_2017	ann_2018	ann_2019
2	Janvier	0	0	0	0	0	890	0	0	0	0	0	
3	Février	0	0	13993	0	0	0	0	0	0	0	0	5
4	Mars	0	4609	3053	0	6545	0	0	0	13000	0	8950	24
5	Avril	0	2738	5002	0		0	0	6000	0	9465	28007	
6	Mai	0	2790	8923	0	7789	0	0	0	10883	0	47250	
7	Juin	0	8166	9871	0	0	0	0	0	9700	0	16247	
8	Juillet	0	3160	11236	4850	0	0	0	0	0	0	0	
9	Août	12191	4160	6593	13600	11090	16558	5000	10519	54894	12003	0	27
10	Septembre	8029	11785	0	2432	10998	0	0	23183	20213	14321	0	
11	Octobre	8913	7514	0	5500	0	0	0	10895	6764	0	0	
12	Novembre	70	5874	0	0	0	0	5060	0	16202	0	0	
13	Décembre	0	8082	0	11509	0	0	0	0	0	0	0	

qté-en-tonne

Source:

الجمهوريّة التونسيّة

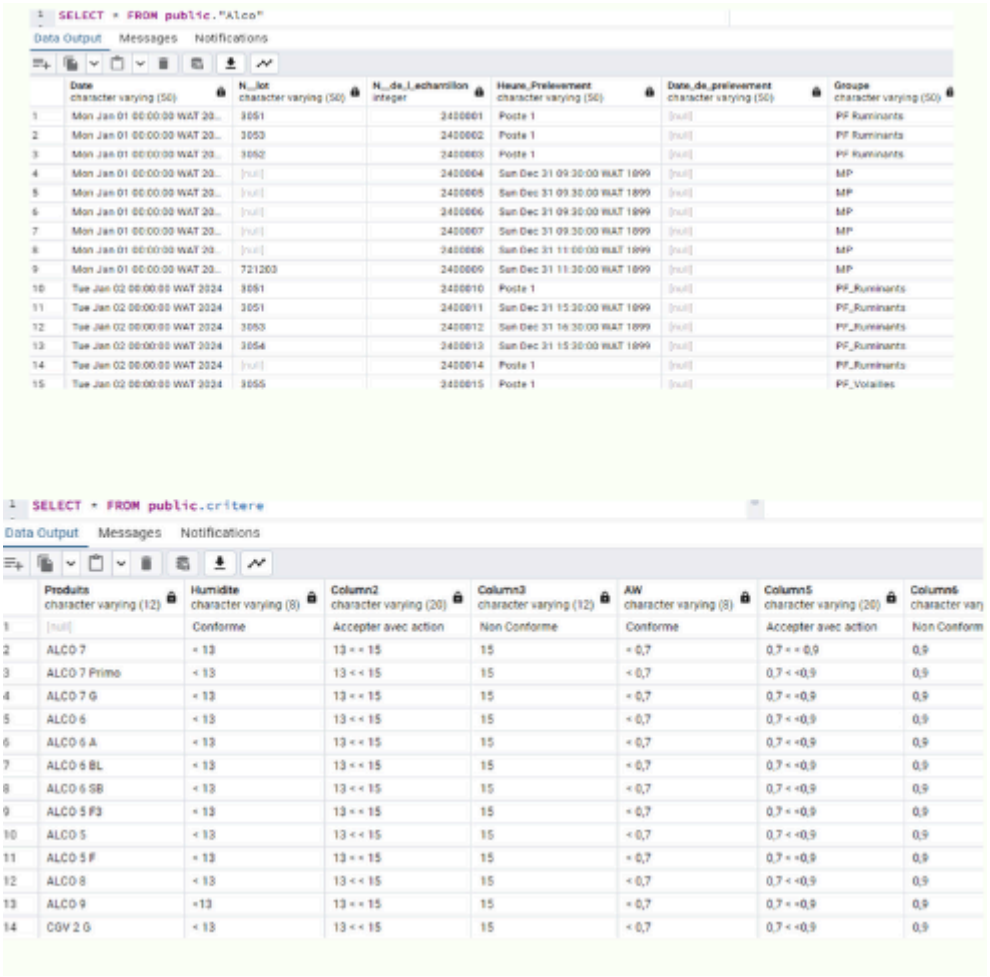
وزارة الفلاحة والموارد المائيّة والصيد البحري

2. ***Talend***

2.1 Alco

<u>Step 1 : Database Importation in Talend:</u>

Staging area : We have created a staging area using the Alco Excel file containing the company's data, along with a criteria file containing the specifications for each component. Additionally, we have incorporated four separate files for feed, corn, soybean, and wheat prices, each providing annual data.



SELECT * FROM public."Alco"

Data Output | Messages | Notifications

	Date character varying (50)	N__lot character varying (50)	N__de_l_echantillon integer	Heure_Prelevement character varying (50)	Date_de_prelevement character varying (50)	Groupe character varying (50)
1	Mon Jan 01 00:00:00 WAT 20...	3051	2400001	Poste 1	[null]	PF Ruminants
2	Mon Jan 01 00:00:00 WAT 20...	3053	2400002	Poste 1	[null]	PF Ruminants
3	Mon Jan 01 00:00:00 WAT 20...	3052	2400003	Poste 1	[null]	PF Ruminants
4	Mon Jan 01 00:00:00 WAT 20...	[null]	2400004	Sun Dec 31 09:30:00 WAT 1899	[null]	MP
5	Mon Jan 01 00:00:00 WAT 20...	[null]	2400005	Sun Dec 31 09:30:00 WAT 1899	[null]	MP
6	Mon Jan 01 00:00:00 WAT 20...	[null]	2400006	Sun Dec 31 09:30:00 WAT 1899	[null]	MP
7	Mon Jan 01 00:00:00 WAT 20...	[null]	2400007	Sun Dec 31 09:30:00 WAT 1899	[null]	MP
8	Mon Jan 01 00:00:00 WAT 20...	[null]	2400008	Sun Dec 31 11:00:00 WAT 1899	[null]	MP
9	Mon Jan 01 00:00:00 WAT 20...	721203	2400009	Sun Dec 31 11:30:00 WAT 1899	[null]	MP
10	Tue Jan 02 00:00:00 WAT 2024	3051	2400010	Poste 1	[null]	PF_Ruminants
11	Tue Jan 02 00:00:00 WAT 2024	3051	2400011	Sun Dec 31 15:30:00 WAT 1899	[null]	PF_Ruminants
12	Tue Jan 02 00:00:00 WAT 2024	3053	2400012	Sun Dec 31 16:30:00 WAT 1899	[null]	PF_Ruminants
13	Tue Jan 02 00:00:00 WAT 2024	3054	2400013	Sun Dec 31 15:30:00 WAT 1899	[null]	PF_Ruminants
14	Tue Jan 02 00:00:00 WAT 2024	[null]	2400014	Poste 1	[null]	PF_Ruminants
15	Tue Jan 02 00:00:00 WAT 2024	3055	2400015	Poste 1	[null]	PF_Volailles

SELECT * FROM public.critere

Data Output | Messages | Notifications

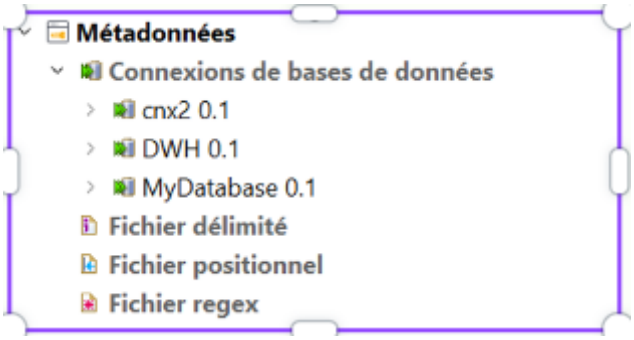
	Produits character varying (12)	Humidite character varying (8)	Column2 character varying (20)	Column3 character varying (12)	AW character varying (8)	Column5 character varying (20)	Column6 character var
1	[null]	Conforme	Accepter avec action	Non Conforme	Conforme	Accepter avec action	Non Conform
2	ALCO 7	< 13	13 < < 15	15	< 0,7	0,7 < < 0,9	0,9
3	ALCO 7 Primo	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
4	ALCO 7 G	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
5	ALCO 6	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
6	ALCO 6 A	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
7	ALCO 6 BL	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
8	ALCO 6 SB	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
9	ALCO 5 F3	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
10	ALCO 5	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
11	ALCO 5 F	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
12	ALCO 8	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
13	ALCO 9	<13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9
14	CGV 2 G	< 13	13 < < 15	15	< 0,7	0,7 < <0,9	0,9

Database Importation in Talend :

Before commencing the construction of the data warehouse staging area, we established an intermediate database using PostgreSQL, acting as a bridge between the source systems and the DW. This database is named "DWH."

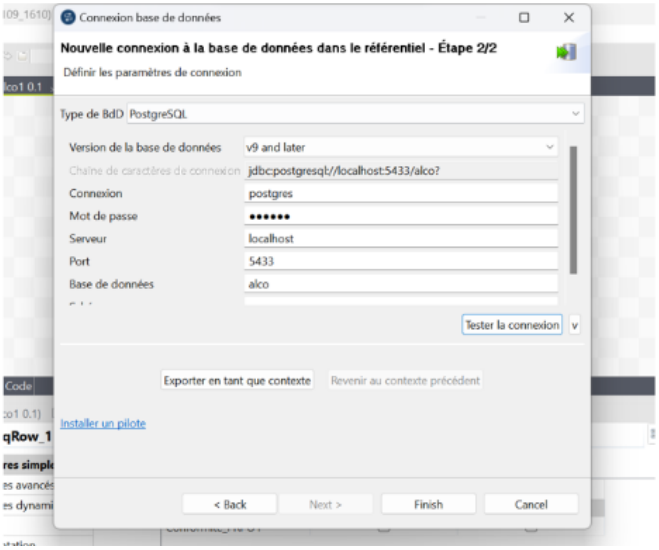
Create the metadata:

Figure illustrates the metadata created for implementing the solution.



Editing a database connection:

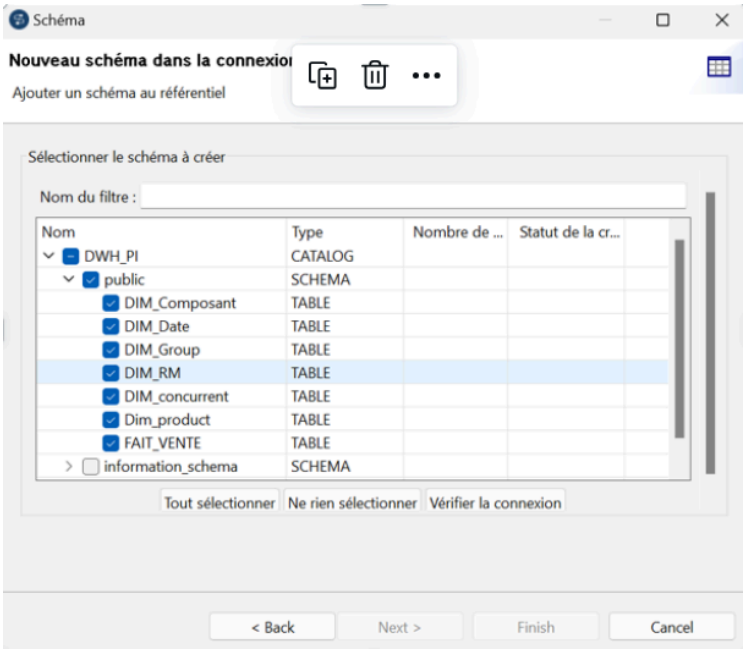
We need to establish the connection to our previously created database by entering various credentials, such as the username, server, and database name. This is illustrated in Figure .



Once completed, we test the connection to check if our information is correct or not and finally our database is ready to use.

<u>Retrieve Database Schema:</u>

The figure shows the retrieval of the schema from the DW tables.

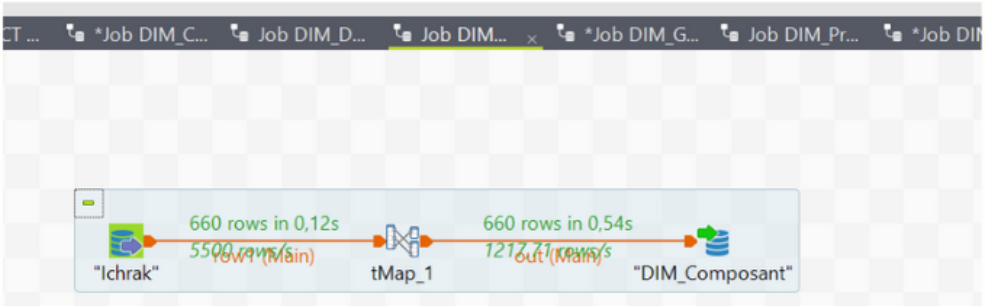


Step 2 : Feeding Dimensions

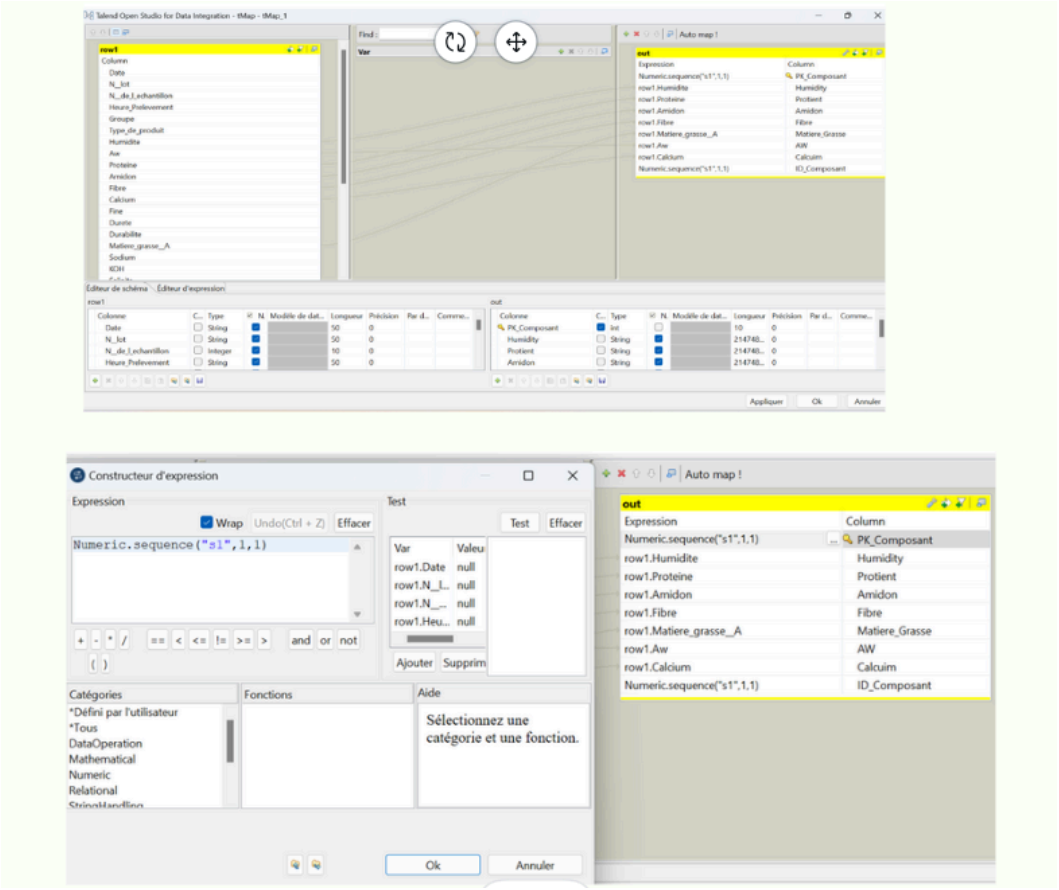
To feed a table in our BD, we need a few components:

- **TPostgresqlInput:** This component is used to read data from a PostgreSQL database. TFilterColumns: This component allows you to select specific columns.
- **Tmap**: This component transforms data from one or more data sources to a destination.
- **TPostgresqlOutput:** This component is used to write transformed data to a PostgreSQL database.

- ***DIM Composant :***



Once finished, with tmap we drag the relevant attributes from the file to the target table "dim_Composant":



For <<PK_Composant >>we use the numeric function “Numeric.sequence(“s1”,1,1)”to have automatic incrementing

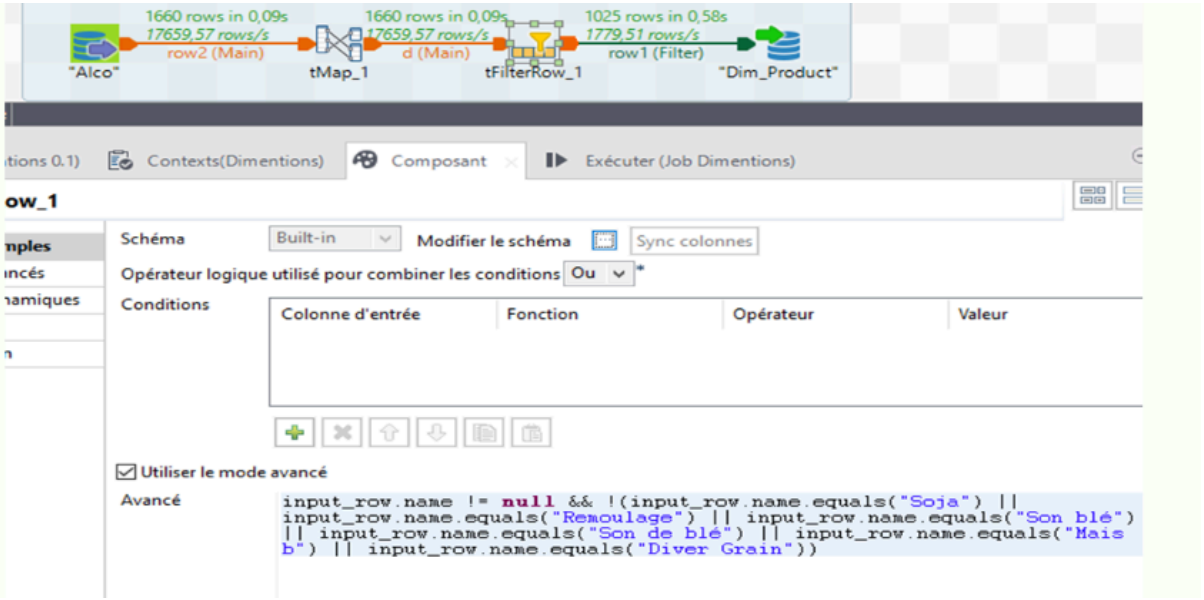
After the execution of this job, here is our"dim_Composant" table created in the comic

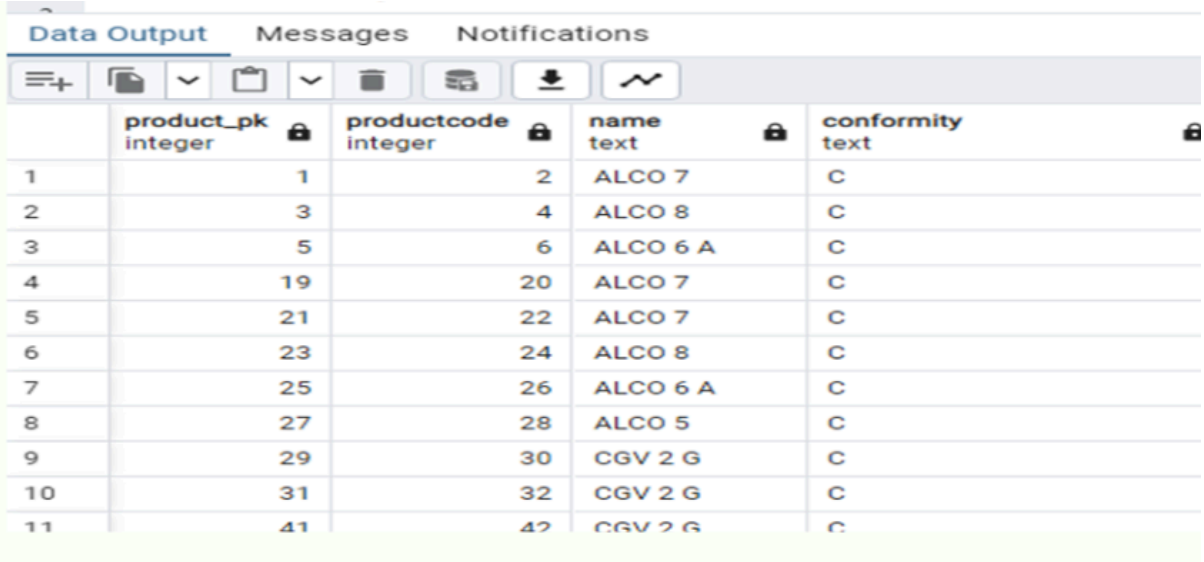
Data Output | Messages | Notifications

	PK_Composant [PK] integer	Humidity text	Protient text	Amidon text	Fibre text	Matiere Grasse text	AW text	Calcuim text	ID_Composant integer
116	231	11,77	46,01	[null]	3,94	2,99	[null]	[null]	232
117	233	11,51	22,72	37,18	2,9	4,09	[null]	0,78	234
118	235	11,83	22,35	36,47	2,9	4,25	[null]	0,81	236
119	237	10,62	19,63	36,63	3,34	3,42	[null]	1,11	238
120	239	10,73	19,23	35,69	3,73	3,64	[null]	1,02	240
121	241	11,04	19,5	36,69	3,18	3,49	[null]	1,12	242
122	243	11,6	16,22	38,56	4,35	3,14	[null]	1,04	244
123	245	11,44	11,11	47,27	4,78	2,91	[null]	0,64	246
124	247	11,39	16,16	39,64	2,58	2,88	[null]	4,43	248
125	249	11,51	47,87	[null]	3,06	1,34	[null]	[null]	250
126	251	13,25	7,49	66,8	1,8	2,99	[null]	[null]	252

- ***DIM Product :***

For the product dimension, when we have a table Alco in the source containing data from Alco, there is a column labeled "product_type." However, this column contains both products and raw materials such as soybean, bran, wheat bran, etc. Therefore, we utilize a TFilterRow component to retrieve only the products, excluding raw materials. I have implemented a condition to achieve this.





	product_pk integer	productcode integer	name text	conformity text
1	1	2	ALCO 7	C
2	3	4	ALCO 8	C
3	5	6	ALCO 6 A	C
4	19	20	ALCO 7	C
5	21	22	ALCO 7	C
6	23	24	ALCO 8	C
7	25	26	ALCO 6 A	C
8	27	28	ALCO 5	C
9	29	30	CGV 2 G	C
10	31	32	CGV 2 G	C
11	41	42	CGV 2 G	C

- ***DIM MP :***

For the raw material (MP) dimension, I've added a TFilterRow component to extract only the raw materials, excluding the products.

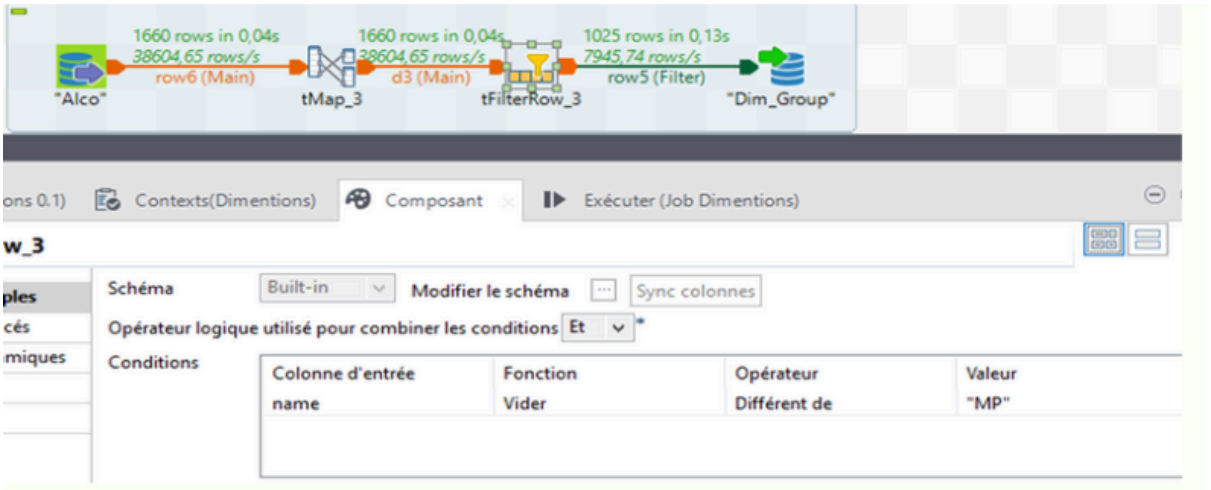
```
1  SELECT * FROM public."Dim_MP"
```

Data Output | Messages | Notifications

	mp_pk integer	mpcode integer	name text
1	3327	3328	Soja
2	3329	3330	Son de blé
3	3331	3332	Mais b
4	3333	3334	Remoulage
5	3335	3336	Soja
6	3337	3338	Soja
7	3353	3354	Soja
8	3355	3356	Son de blé
9	3357	3358	Mais b

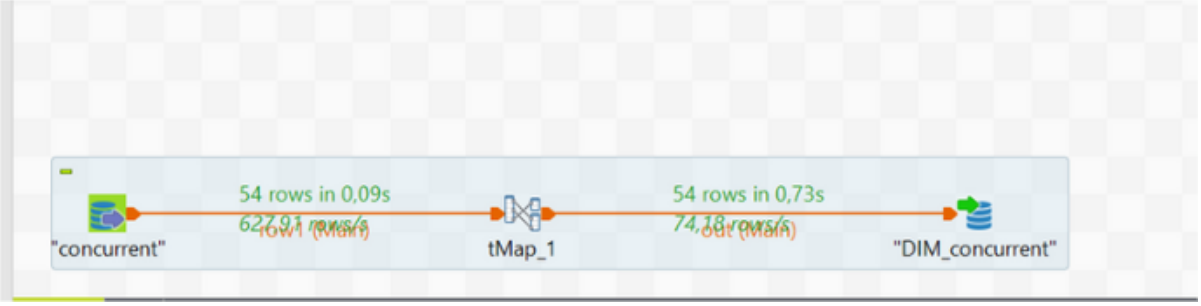
- ***DIM Group :***

For the group dimension, I've implemented a TFilterRow component with a single condition. In the source table, there is a column labeled "group" containing both groups and the term "MP". To exclude the term "MP" and retrieve only the groups, I've applied a condition accordingly.

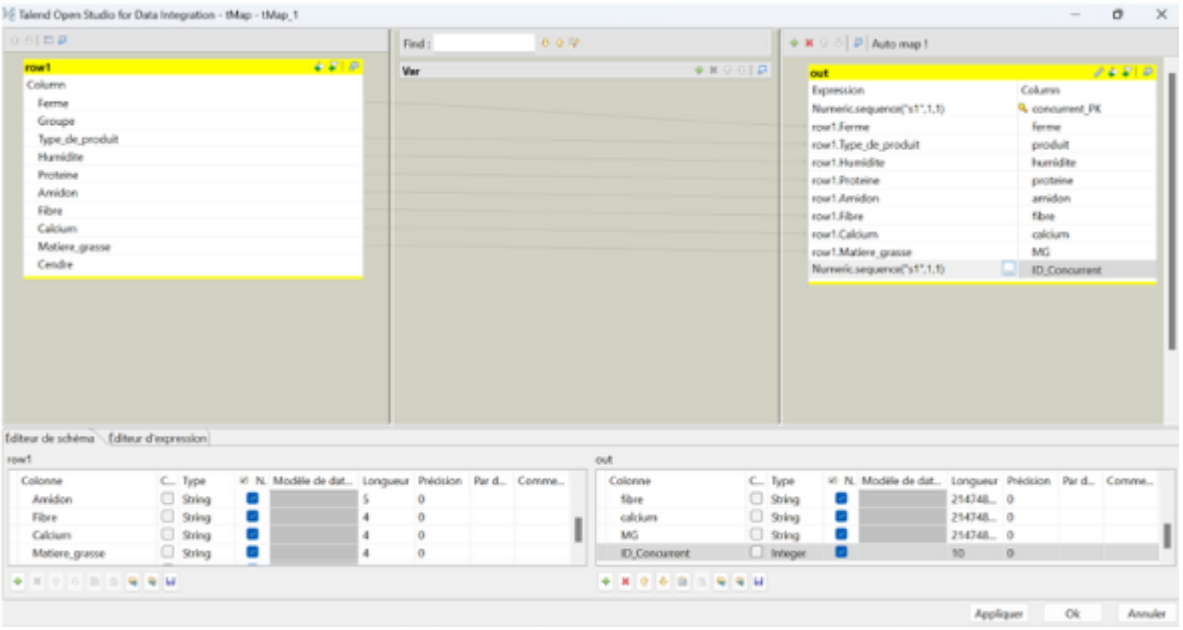


- ***DIM Concurrent :***

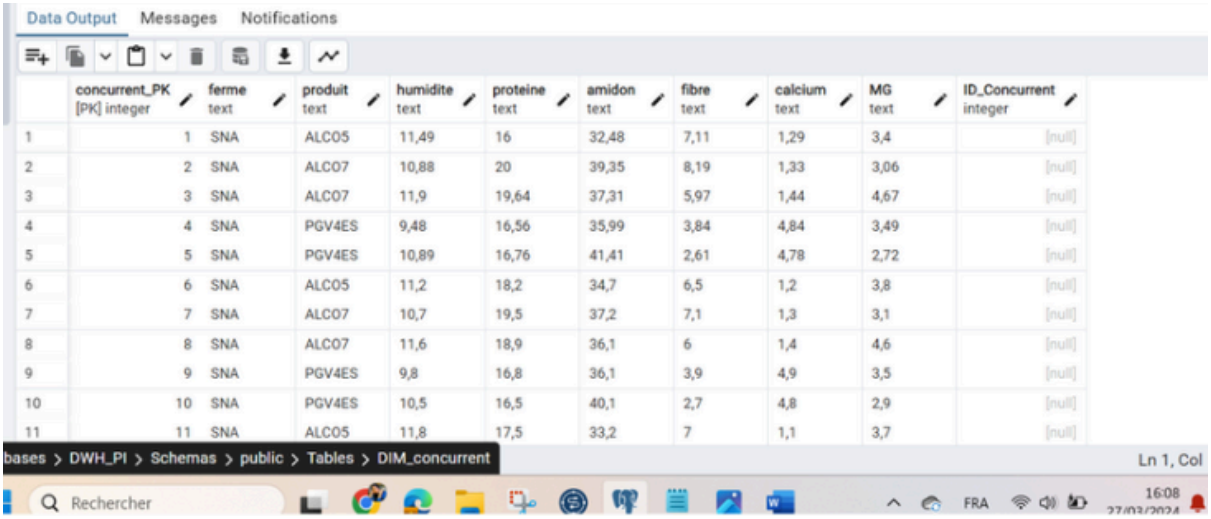
In the same way as for the previous dimension, we place each element in its place in order to complete the job configuration:



Once finished, with tmap we drag the relevant attributes from the file to the target table "dim_Concurrent":



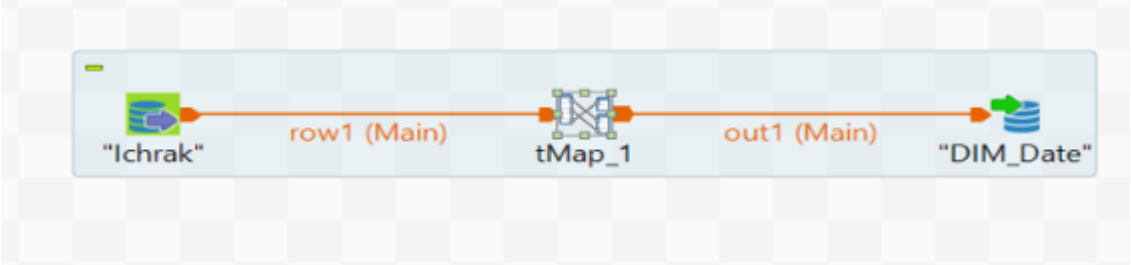
And after the execution of this job, here is our "dim_Concurrent " table created in BD:



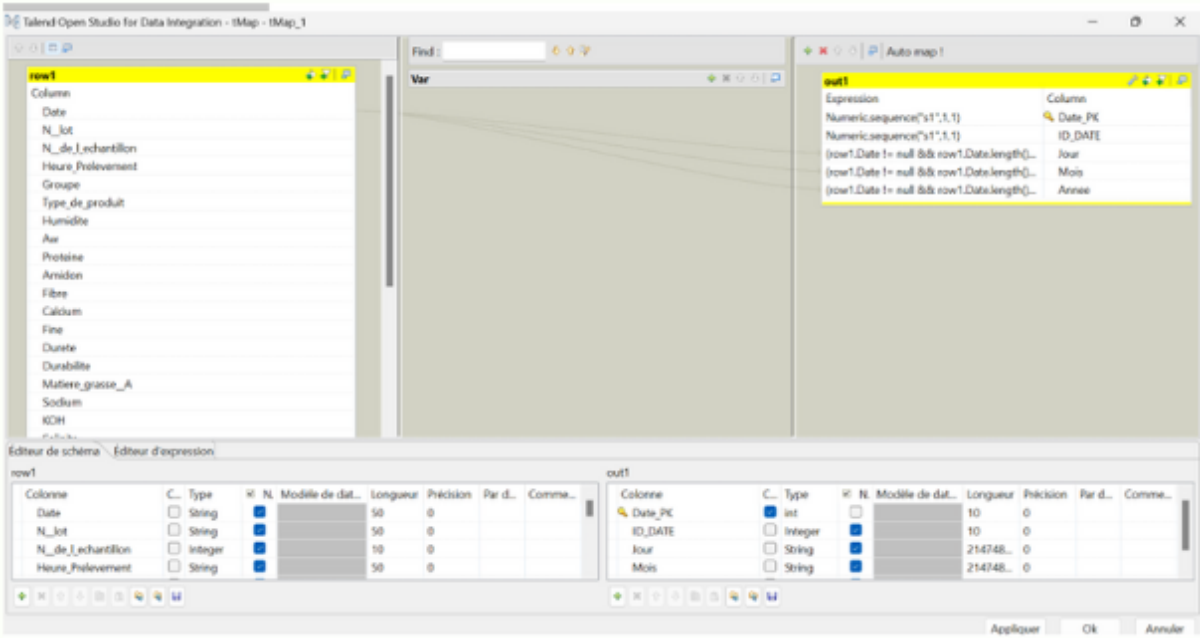
	concurrent_PK [PK] integer	ferme text	produit text	humidite text	proteine text	amidon text	fibre text	calcium text	MG text	ID_Concurrent integer
1	1	SNA	ALC05	11,49	16	32,48	7,11	1,29	3,4	[null]
2	2	SNA	ALC07	10,88	20	39,35	8,19	1,33	3,06	[null]
3	3	SNA	ALC07	11,9	19,64	37,31	5,97	1,44	4,67	[null]
4	4	SNA	PGV4ES	9,48	16,56	35,99	3,84	4,84	3,49	[null]
5	5	SNA	PGV4ES	10,89	16,76	41,41	2,61	4,78	2,72	[null]
6	6	SNA	ALC05	11,2	18,2	34,7	6,5	1,2	3,8	[null]
7	7	SNA	ALC07	10,7	19,5	37,2	7,1	1,3	3,1	[null]
8	8	SNA	ALC07	11,6	18,9	36,1	6	1,4	4,6	[null]
9	9	SNA	PGV4ES	9,8	16,8	36,1	3,9	4,9	3,5	[null]
10	10	SNA	PGV4ES	10,5	16,5	40,1	2,7	4,8	2,9	[null]
11	11	SNA	ALC05	11,8	17,5	33,2	7	1,1	3,7	[null]

- ***DIM Date :***

In the same way as for the previous dimension, we place each element in its place in order to complete the job configuration:



After completion, using tmap, we transfer the pertinent attributes from the file to the destination table called "dim_Date."



Following the execution of this task, we observe the creation of our "dim_Date" table within the database.

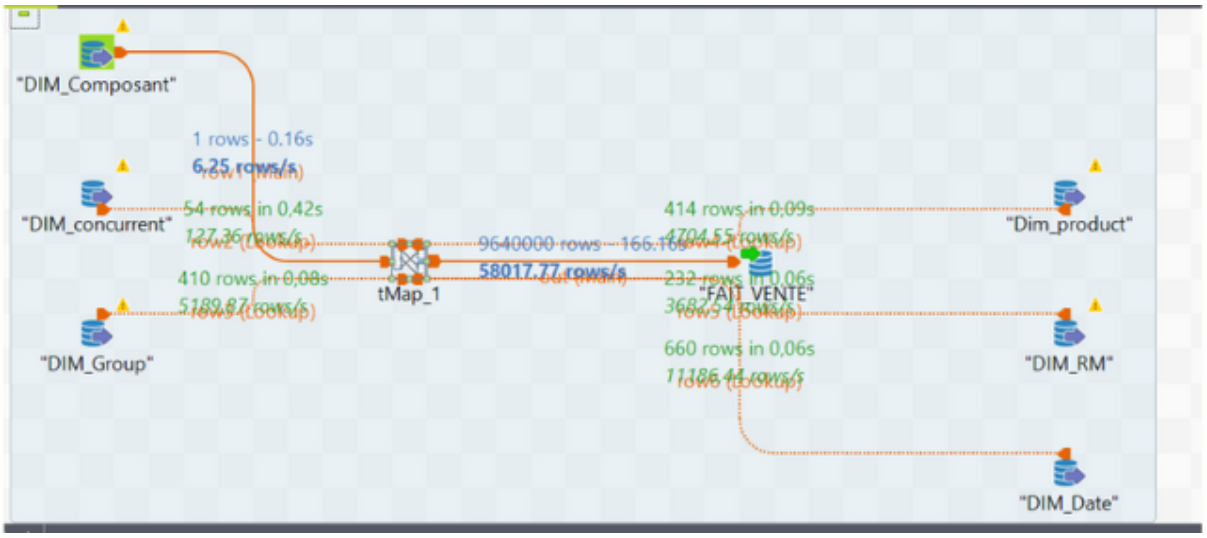
	Date_PK [PK] integer	ID_DATE integer	Jour text	Mois text	Annee text
1	1	2	01	Jan	2024
2	3	4	01	Jan	2024
3	5	6	01	Jan	2024
4	7	8	01	Jan	2024
5	9	10	01	Jan	2024
6	11	12	01	Jan	2024
7	13	14	01	Jan	2024
8	15	16	01	Jan	2024
9	17	18	01	Jan	2024
10	19	20	02	Jan	2024
11	21	22	02	Jan	2024

ases > DWH_PI > Schemas > public > Tables > DIM_Date

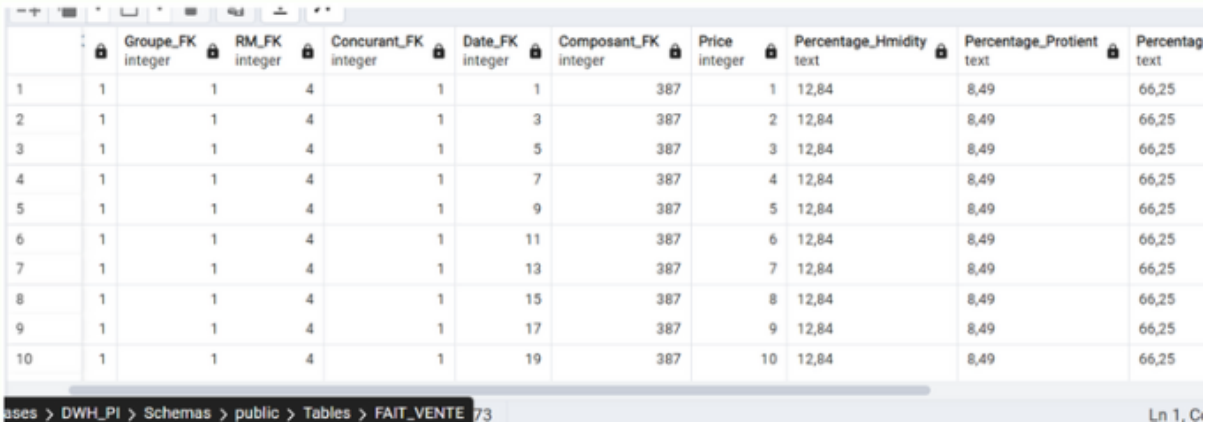
Step 3 : Feeding Fact table

- ***FACT Vente :***

We populate the fact table by integrating relevant data from a variety of sources, creating a consolidated database that will be used for future analyses.



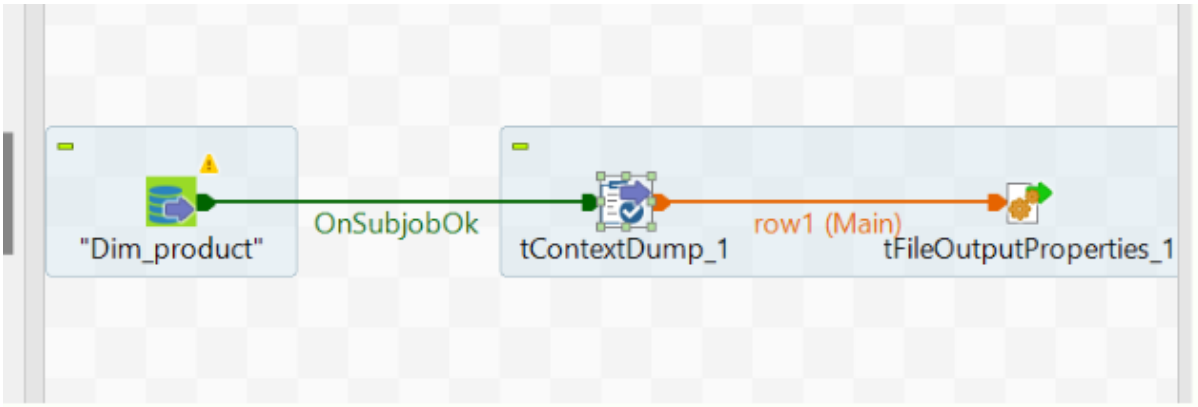
In this step, we're going to load our target table which is the table made with the different attributes already defined earlier:



	Groupe_FK integer	RM_FK integer	Concurant_FK integer	Date_FK integer	Composant_FK integer	Price integer	Percentage_Hmidity text	Percentage_Protient text	Percentag text
1	1	1	4	1	1	387	1	12,84	8,49	66,25
2	1	1	4	1	3	387	2	12,84	8,49	66,25
3	1	1	4	1	5	387	3	12,84	8,49	66,25
4	1	1	4	1	7	387	4	12,84	8,49	66,25
5	1	1	4	1	9	387	5	12,84	8,49	66,25
6	1	1	4	1	11	387	6	12,84	8,49	66,25
7	1	1	4	1	13	387	7	12,84	8,49	66,25
8	1	1	4	1	15	387	8	12,84	8,49	66,25
9	1	1	4	1	17	387	9	12,84	8,49	66,25
10	1	1	4	1	19	387	10	12,84	8,49	66,25

ases > DWH_PI > Schemas > public > Tables > FAIT_VENTE 73 Ln 1, C

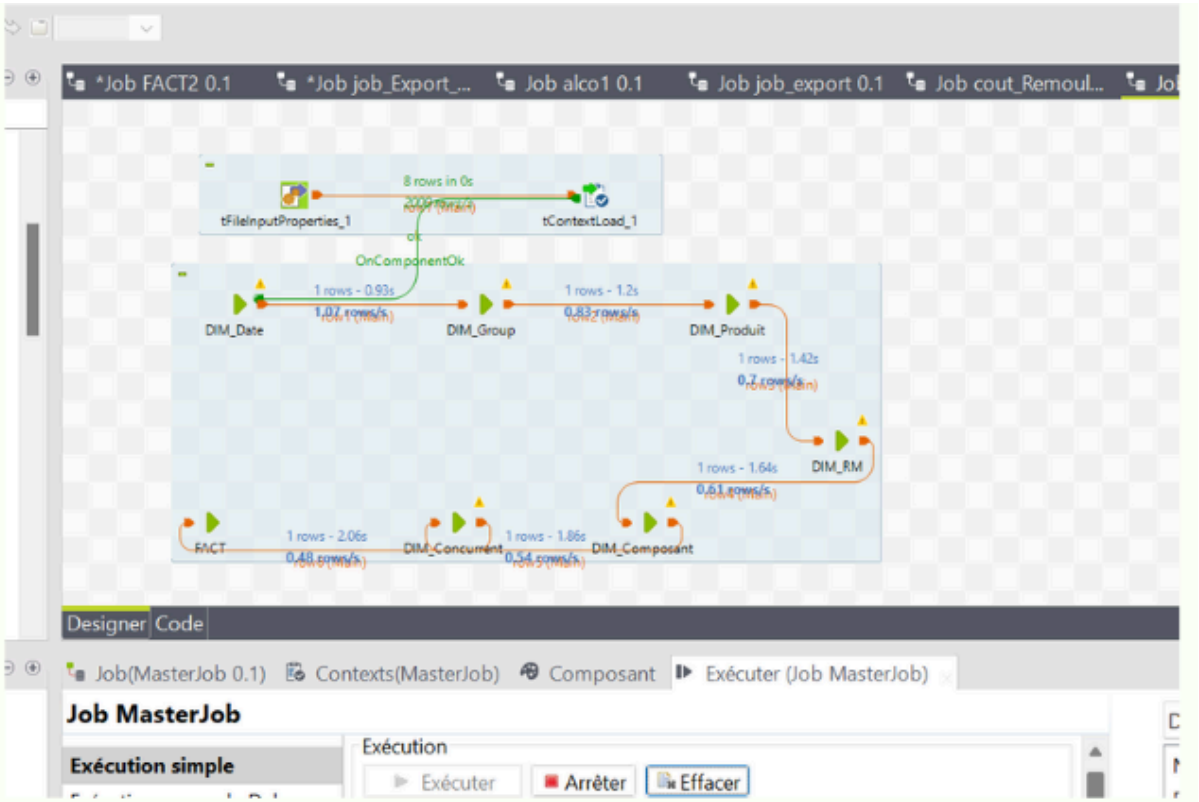
Export Database Connection as Context



Initially, the database connection details are configured within the Metadata repository. These details are then exported as context variables, allowing for their easy reuse across different components or workflows.

Exporting Database Parameters in a Properties File: Subsequently, a job is set up to export database parameters into a properties file.

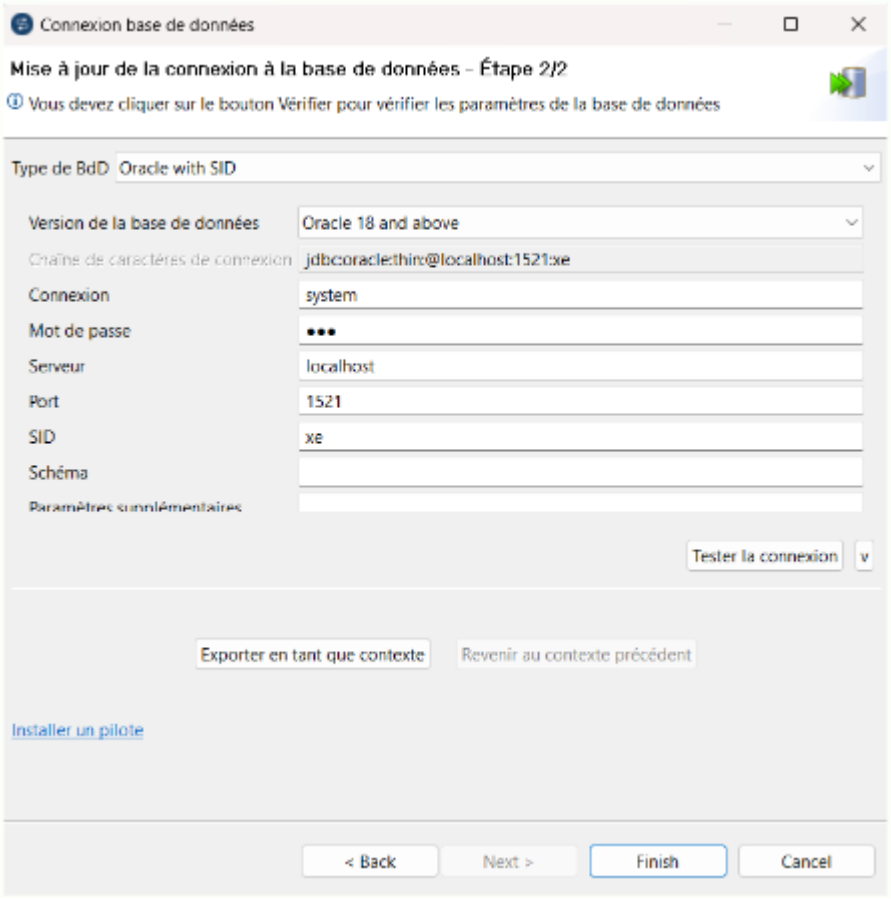
This job utilizes three key components: tDBInput retrieves database parameters, tContextDump exports them as context variables, and tFileOutputProperties stores them in a properties file.

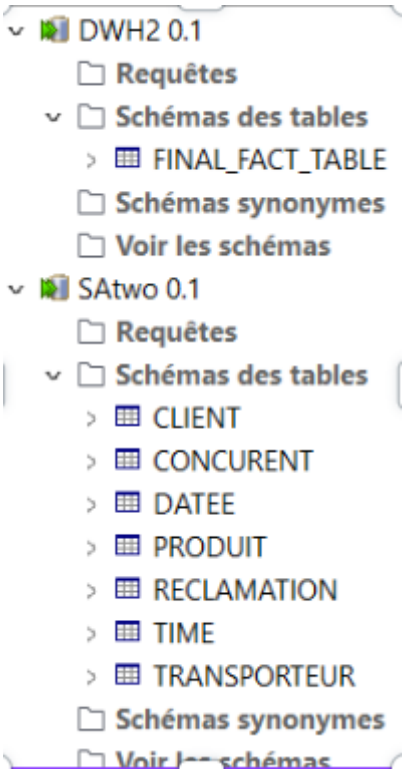


The process involves creating a master job to oversee and execute individual tasks sequentially. This master job integrates with the main database connection and source data paths through two components: tFileInputProperties and tContextLoad. tFileInputProperties: This component connects to the "Connection.properties" file, housing crucial database connection details. We configure it to switch the "Property Type" to "Build-In" and combine the source data path with the file name for smooth integration. tContextLoad: Used to import variables from the "Connection.properties" file, ensuring all necessary context variables are available throughout the master job.

2.2 STPA

Step 1 : Database Importation in Talend:


Connexion base de données
Mise à jour de la connexion à la base de données – Étape 2/2
Vous devez cliquer sur le bouton Vérifier pour vérifier les paramètres de la base de données
Type de BdD
Oracle with SID
Version de la base de données
Oracle 18 and above
Chaîne de caractères de connexion
jdbc:oracle:thin:@localhost:1521:xe
Connexion
system
Mot de passe
Serveur
localhost
Port
1521
SID
xe
Schéma
Paramètres supplémentaires
Tester la connexion
Exporter en tant que contexte
Revenir au contexte précédent
Installer un pilote
< Back
Next >
Finish
Cancel


DWH2 0.1
Requêtes
Schémas des tables
FINAL_FACT_TABLE
Schémas synonymes
Voir les schémas
SAtwo 0.1
Requêtes
Schémas des tables
CLIENT
CONCURENT
DATEE
PRODUIT
RECLAMATION
TIME
TRANSPORTEUR
Schémas synonymes

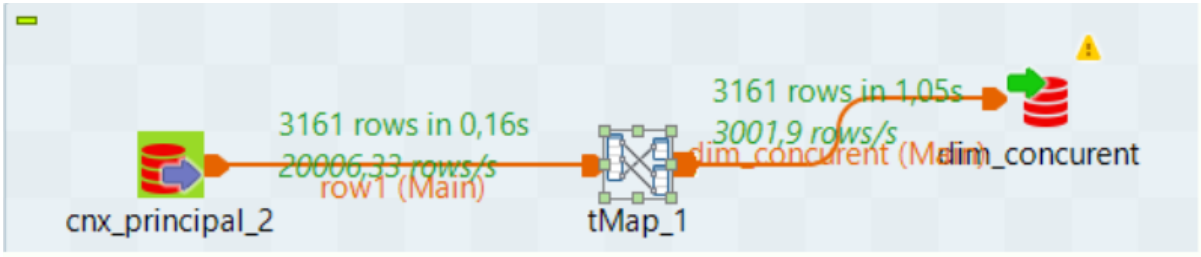
We've linked SQL Developer with Talend to simplify importing our working database, named FINAL_FACT_TABLE. This setup streamlines access and manipulation within Talend's environment, ensuring seamless data transfer and synchronization. With this connection, we're well-equipped to maximize the capabilities of both platforms for efficient data management and analysis, aiding us in achieving our goals.

We specifically linked SQL Developer to create the various dimensions and our fact table, as depicted in the figure. This connection streamlines the process of designing and implementing our database schema directly within Talend's environment.

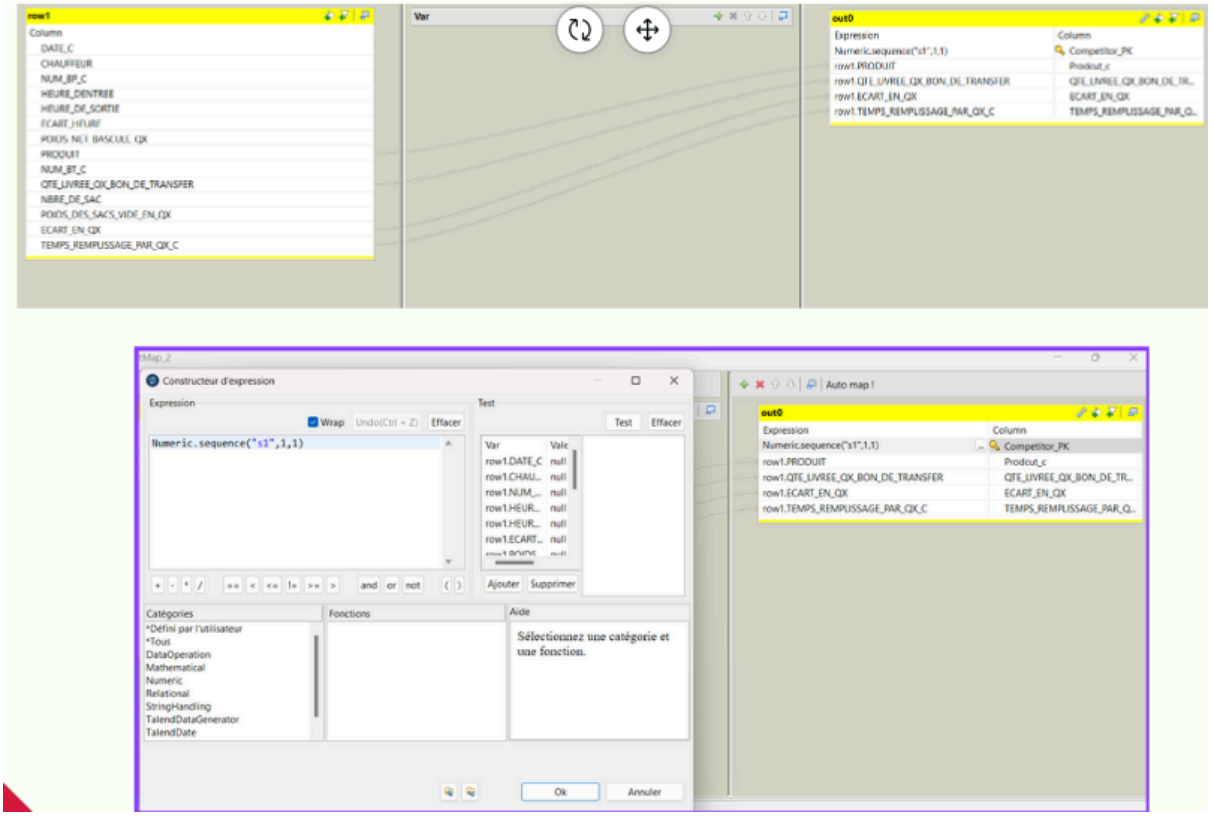
Step 2 : Feeding Dimensions

To feed a table in our BD, we need a few components: TDBInput: This component is used to read data from an Oracle SQl Developer database. TFilterRow: This component allows you to filter specific rows. Tmap: This component transforms data from one or more data sources to a destination. TDBOutput: This component is used to write transformed data to an Oracle SQl Developer database..

- ***DIM Competitor :***

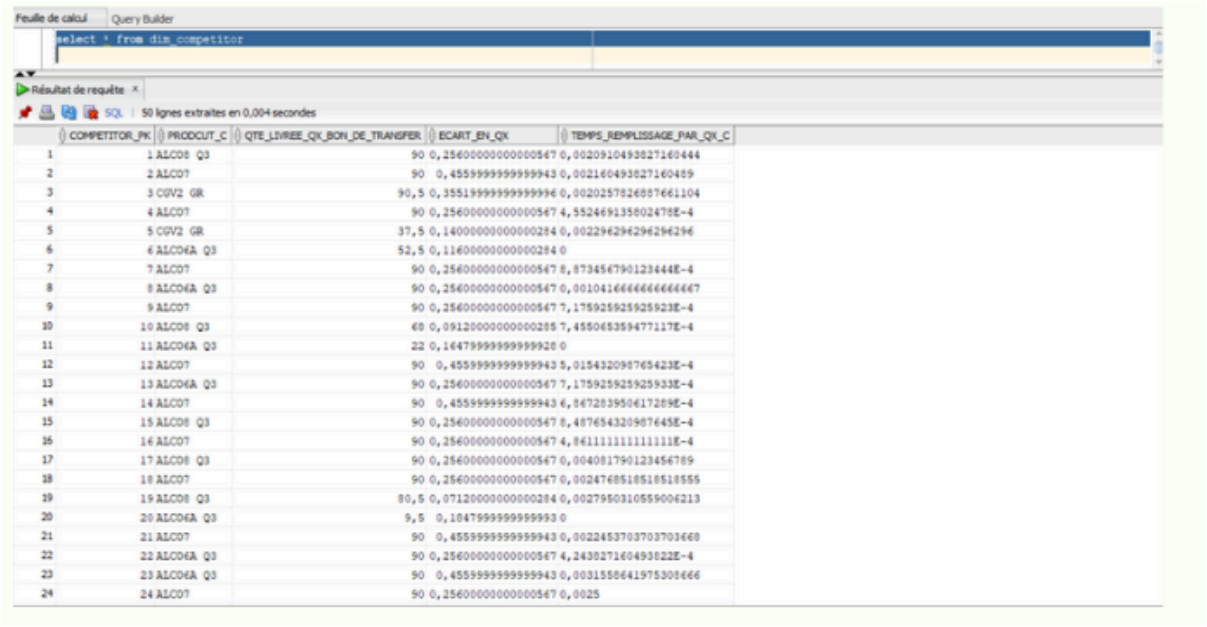


Once finished, with tmap we drag the relevant attributes from the file to the target table "dim_concurrent":



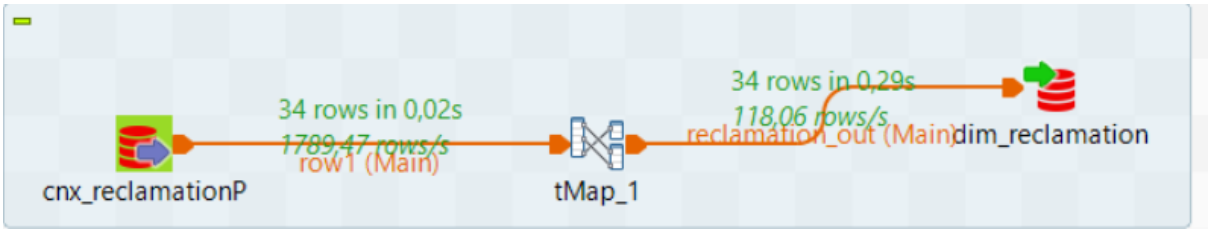
For <<Competitor_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

After the execution of this job, here is our "dim_Concurrent" table created in SQL Developer.

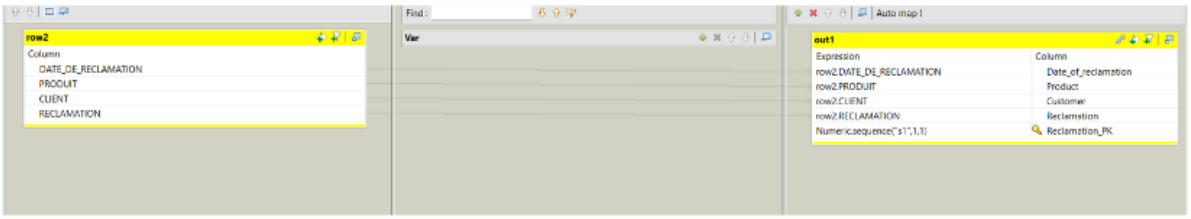


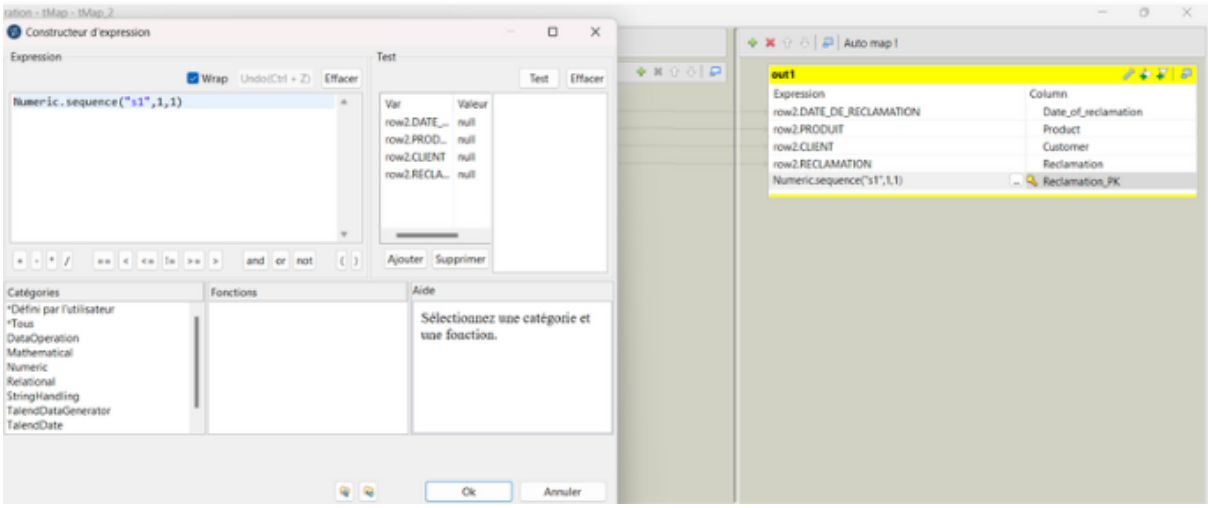
	COMPETITOR_PK	PRODCUT_C	QTE_LIVREE_QX_BON_DE_TRANSFER	ECART_EN_QX	TEMPS_REMPLISSAGE_PAR_QX_C
1	1	ALCO8 Q3	90	0,25600000000000547	0,0020910493827160444
2	2	ALCO7	90	0,4559999999999943	0,002160493827160489
3	3	CGV2 GR	90,5	0,35519999999999996	0,0020257826887661104
4	4	ALCO7	90	0,25600000000000547	4,552469135802478E-4
5	5	CGV2 GR	37,5	0,14000000000000284	0,002296296296296296
6	6	ALCO8A Q3	52,5	0,11600000000000284	0
7	7	ALCO7	90	0,25600000000000547	8,873456790123444E-4
8	8	ALCO8A Q3	90	0,25600000000000547	0,0010416666666666667
9	9	ALCO7	90	0,25600000000000547	7,175925925925923E-4
10	10	ALCO8 Q3	68	0,09120000000000285	7,455065359477117E-4
11	11	ALCO8A Q3	22	0,16479999999999928	0
12	12	ALCO7	90	0,4559999999999943	5,015432098765423E-4
13	13	ALCO8A Q3	90	0,25600000000000547	7,175925925925933E-4
14	14	ALCO7	90	0,4559999999999943	6,567283950617289E-4
15	15	ALCO8 Q3	90	0,25600000000000547	8,487654320987645E-4
16	16	ALCO7	90	0,25600000000000547	4,8611111111111111E-4
17	17	ALCO8 Q3	90	0,25600000000000547	0,0040817901234567 89
18	18	ALCO7	90	0,25600000000000547	0,0024768518518518555
19	19	ALCO8 Q3	80,5	0,07120000000000284	0,0027950310559006213
20	20	ALCO8A Q3	9,5	0,18479999999999993	0
21	21	ALCO7	90	0,4559999999999943	0,0022453703703703668
22	22	ALCO8A Q3	90	0,25600000000000547	4,2438271604938228E-4
23	23	ALCO8A Q3	90	0,4559999999999943	0,0031558641975308666
24	24	ALCO7	90	0,25600000000000547	0,0025

- ***DIM Reclamation:***



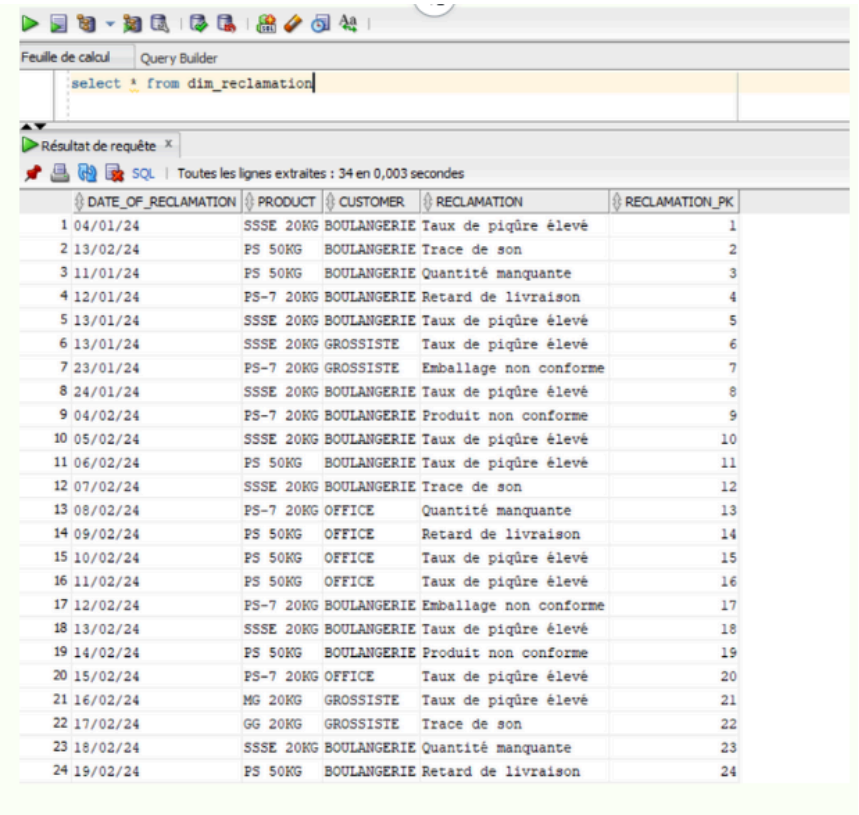
Once finished, with tmap we drag the relevant attributes from the file to the target table "dim_reclamation":





For <<Reclamation_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

After the execution of this job, here is our "dim_reclamation" table created in SQL Developer.



Feuille de calcul | Query Builder

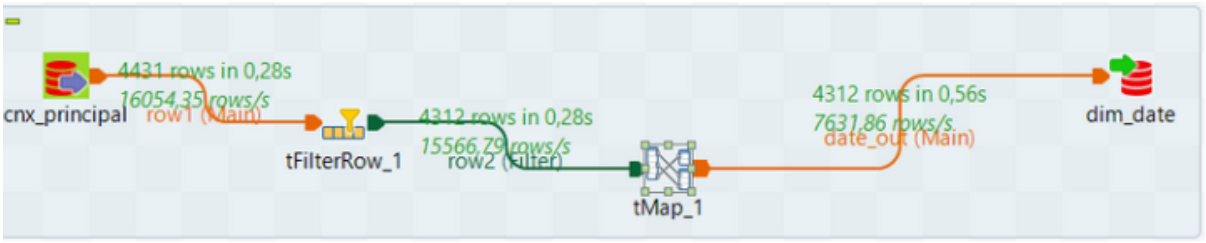
```
select * from dim_reclamation
```

Résultat de requête

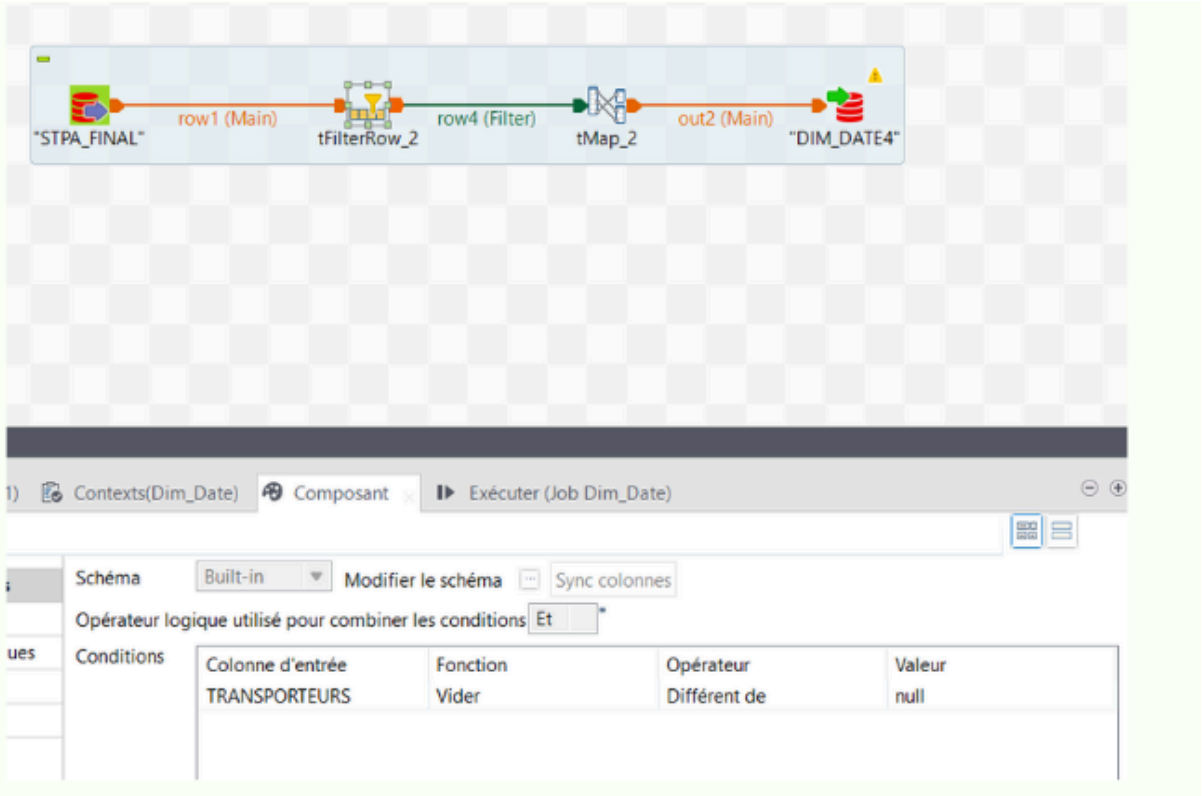
SQL | Toutes les lignes extraites : 34 en 0,003 secondes

	DATE_OF_RECLAMATION	PRODUCT	CUSTOMER	RECLAMATION	RECLAMATION_PK
1	04/01/24	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	1
2	13/02/24	PS 50KG	BOULANGERIE	Trace de son	2
3	11/01/24	PS 50KG	BOULANGERIE	Quantité manquante	3
4	12/01/24	PS-7 20KG	BOULANGERIE	Retard de livraison	4
5	13/01/24	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	5
6	13/01/24	SSSE 20KG	GROSSISTE	Taux de piqûre élevé	6
7	23/01/24	PS-7 20KG	GROSSISTE	Emballage non conforme	7
8	24/01/24	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	8
9	04/02/24	PS-7 20KG	BOULANGERIE	Produit non conforme	9
10	05/02/24	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	10
11	06/02/24	PS 50KG	BOULANGERIE	Taux de piqûre élevé	11
12	07/02/24	SSSE 20KG	BOULANGERIE	Trace de son	12
13	08/02/24	PS-7 20KG	OFFICE	Quantité manquante	13
14	09/02/24	PS 50KG	OFFICE	Retard de livraison	14
15	10/02/24	PS 50KG	OFFICE	Taux de piqûre élevé	15
16	11/02/24	PS 50KG	OFFICE	Taux de piqûre élevé	16
17	12/02/24	PS-7 20KG	BOULANGERIE	Emballage non conforme	17
18	13/02/24	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	18
19	14/02/24	PS 50KG	BOULANGERIE	Produit non conforme	19
20	15/02/24	PS-7 20KG	OFFICE	Taux de piqûre élevé	20
21	16/02/24	MG 20KG	GROSSISTE	Taux de piqûre élevé	21
22	17/02/24	GG 20KG	GROSSISTE	Trace de son	22
23	18/02/24	SSSE 20KG	BOULANGERIE	Quantité manquante	23
24	19/02/24	PS 50KG	BOULANGERIE	Retard de livraison	24

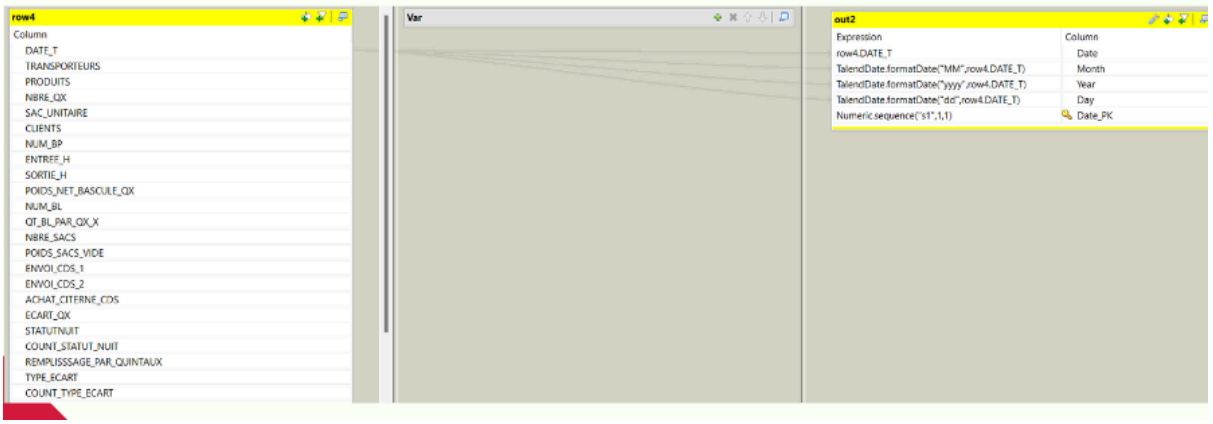
- ***DIM Date:***

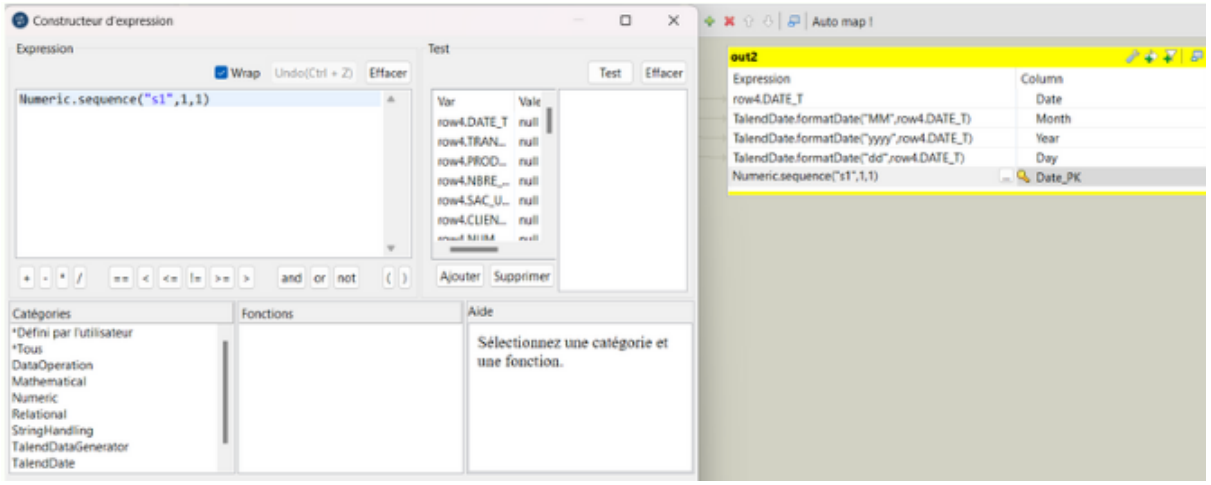


With TFilterRow, we removed the rows where the carriers are null as we do not need them.



Once finished, with tmap we drag the relevant attributes from the file to the target table "dim_date":





For <<Date_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

After the execution of this job, here is our "dim_date" table created in SQL Developer.

Feuille de calcul | Query Builder

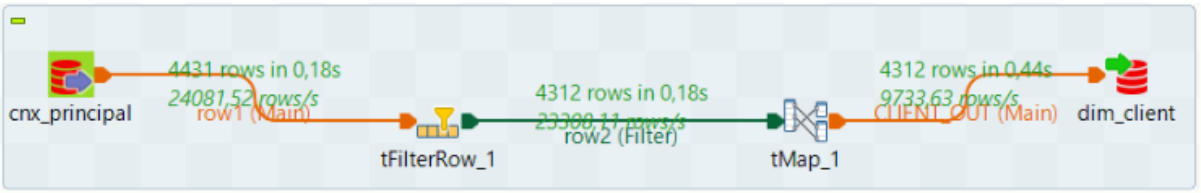
```
select * from dim_date4
```

Résultat de requête

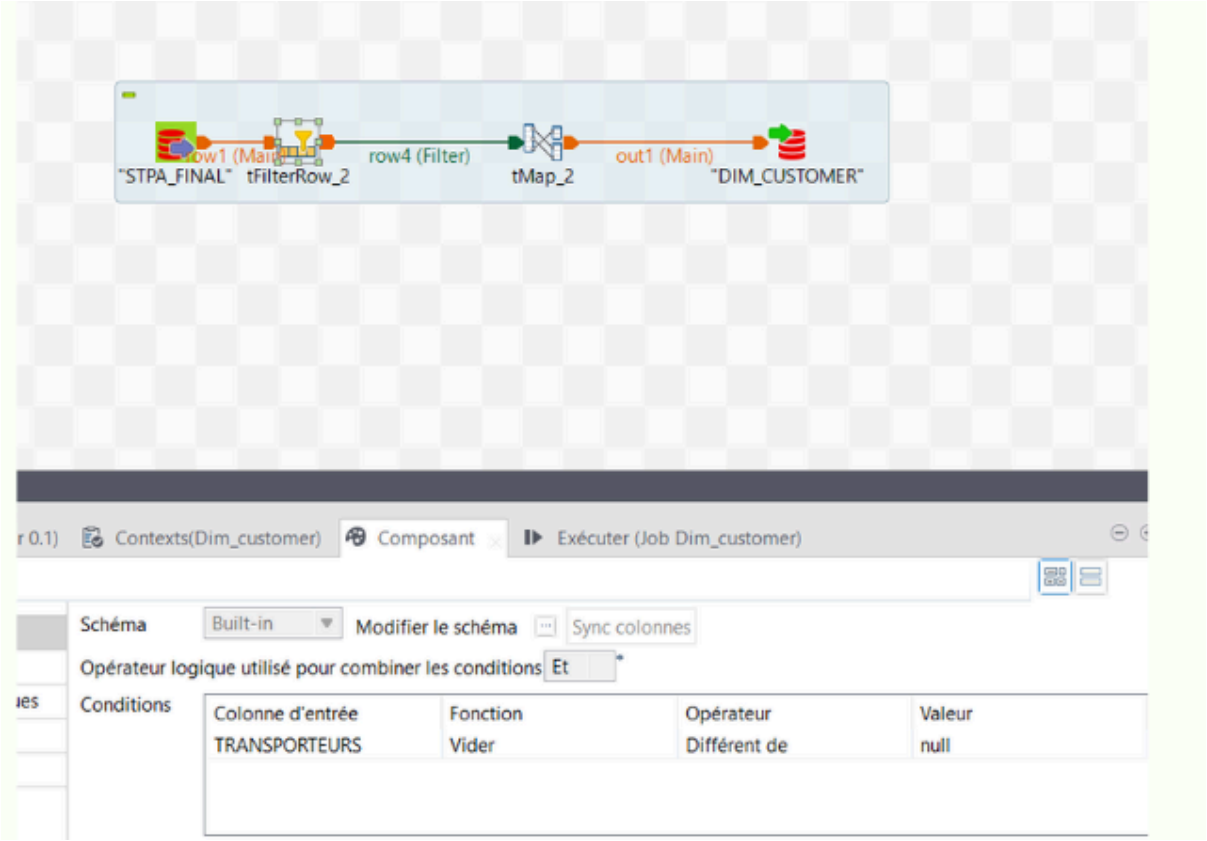
SQL | 50 lignes extraites en 0,003 secondes

	DATE_PK	DAY	MONTH	YEAR	Date
1	1	11	01	2024	11/01/24
2	2	11	01	2024	11/01/24
3	3	11	01	2024	11/01/24
4	4	11	01	2024	11/01/24
5	5	11	01	2024	11/01/24
6	6	11	01	2024	11/01/24
7	7	11	01	2024	11/01/24
8	8	11	01	2024	11/01/24
9	9	11	01	2024	11/01/24
10	10	11	01	2024	11/01/24
11	11	11	01	2024	11/01/24
12	12	11	01	2024	11/01/24
13	13	11	01	2024	11/01/24
14	14	11	01	2024	11/01/24
15	15	11	01	2024	11/01/24
16	16	11	01	2024	11/01/24

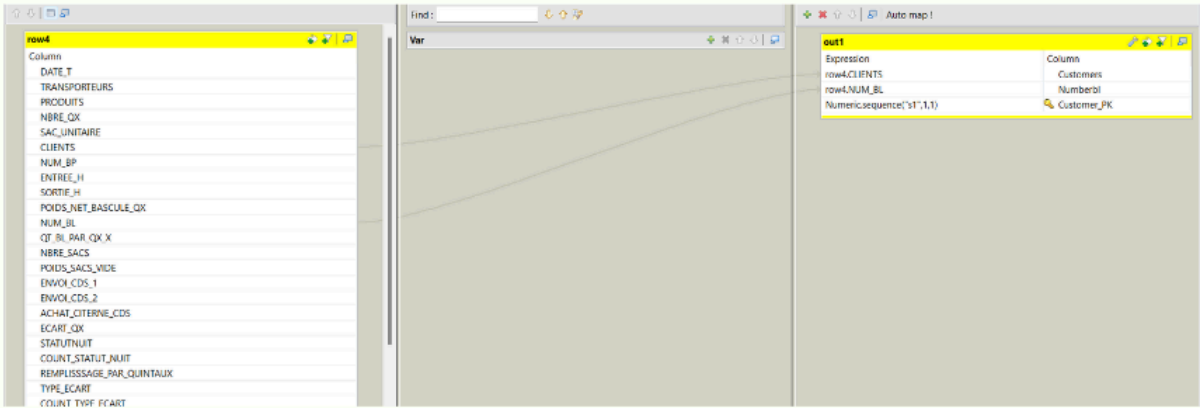
- ***DIM Customer:***

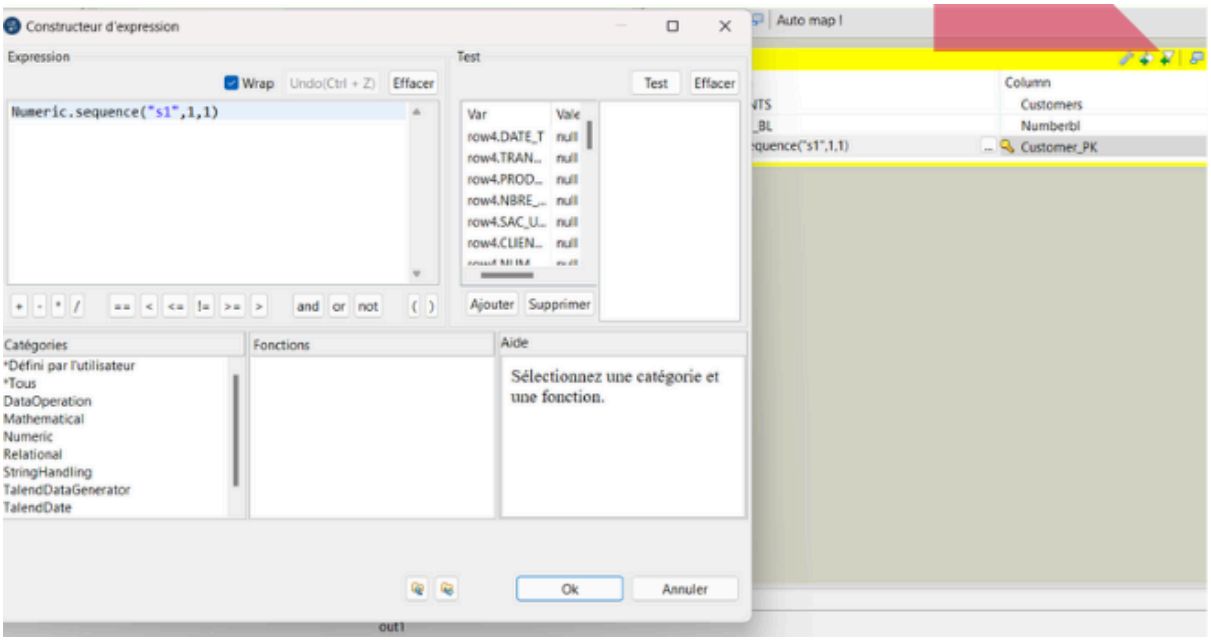


With TFilterRow, we removed the rows where the carriers are null as we do not need them.



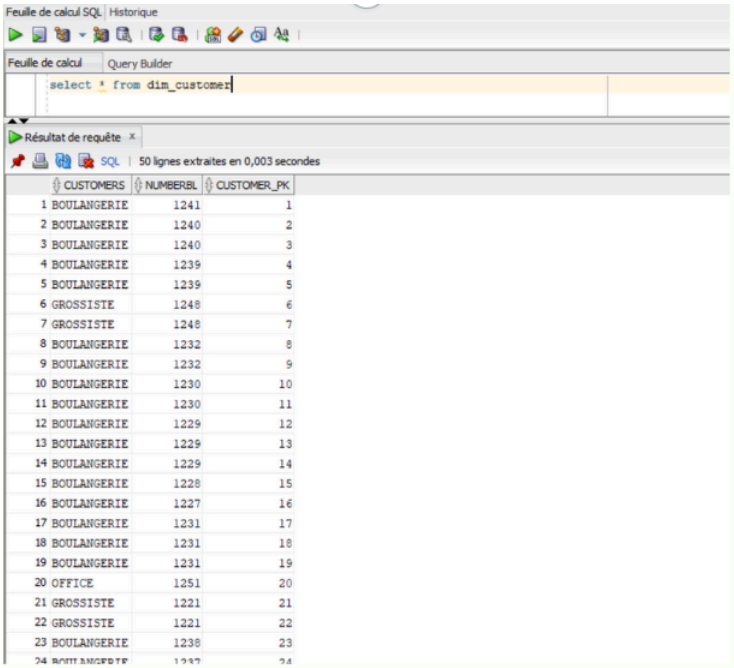
Once finished, with tmap we drag the relevant attributes from the file to the target table "Dim_Customer":





For <<Customer_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

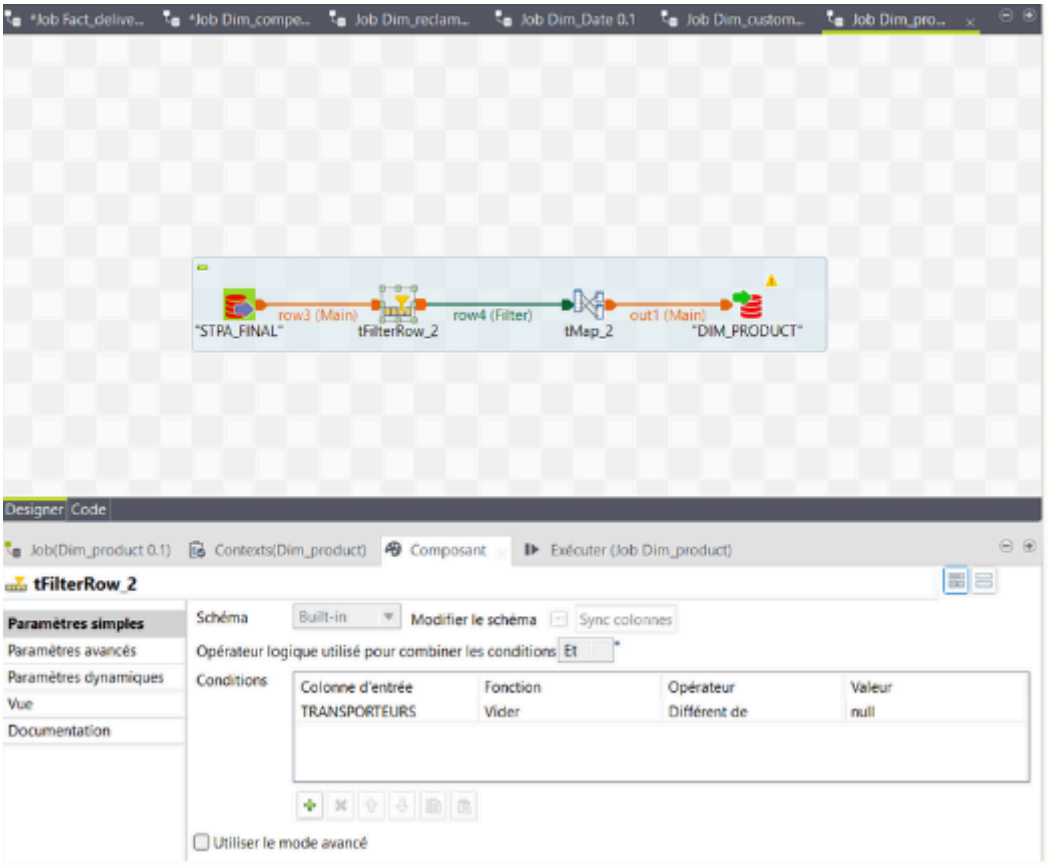
After the execution of this job, here is our "dim_customer" table created in SQL Developer.



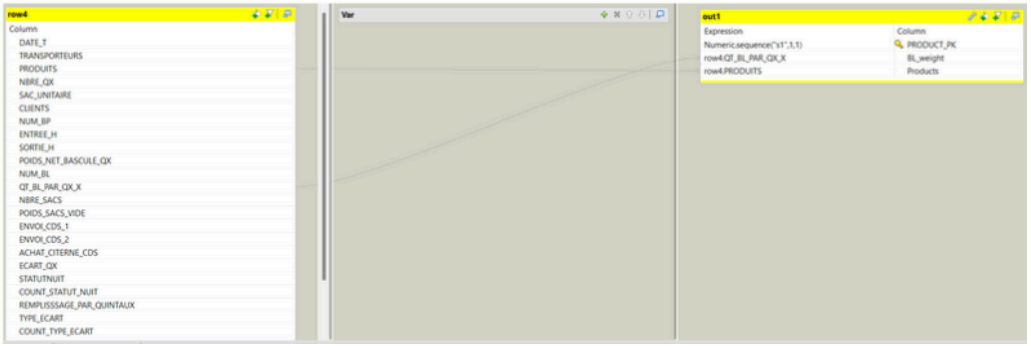
	CUSTOMERS	NUMBERBL	CUSTOMER_PK
1	BOULANGERIE	1241	1
2	BOULANGERIE	1240	2
3	BOULANGERIE	1240	3
4	BOULANGERIE	1239	4
5	BOULANGERIE	1239	5
6	GROSSISTE	1248	6
7	GROSSISTE	1248	7
8	BOULANGERIE	1232	8
9	BOULANGERIE	1232	9
10	BOULANGERIE	1230	10
11	BOULANGERIE	1230	11
12	BOULANGERIE	1229	12
13	BOULANGERIE	1229	13
14	BOULANGERIE	1229	14
15	BOULANGERIE	1228	15
16	BOULANGERIE	1227	16
17	BOULANGERIE	1231	17
18	BOULANGERIE	1231	18
19	BOULANGERIE	1231	19
20	OFFICE	1251	20
21	GROSSISTE	1221	21
22	GROSSISTE	1221	22
23	BOULANGERIE	1238	23

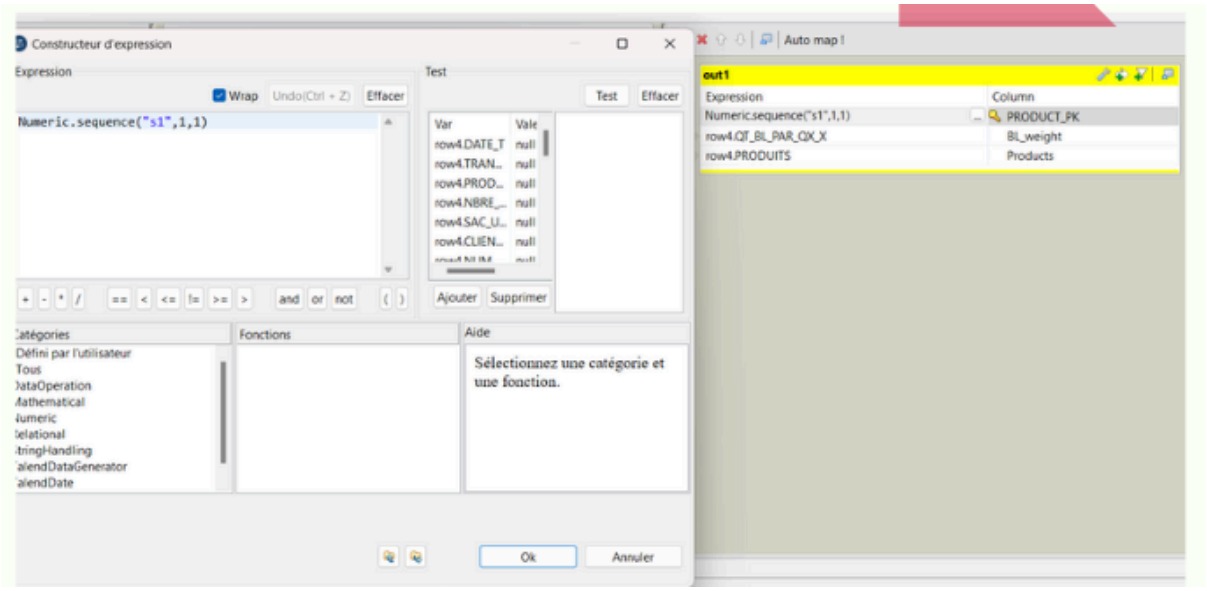
- ***DIM Product:***

With TFilterRow, we removed the rows where the carriers are null as we do not need them.



Once finished, with tmap we drag the relevant attributes from the file to the target table "Dim_Product":





For <<Product_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

After the execution of this job, here is our "Dim_Product" table created in SQL Developer.

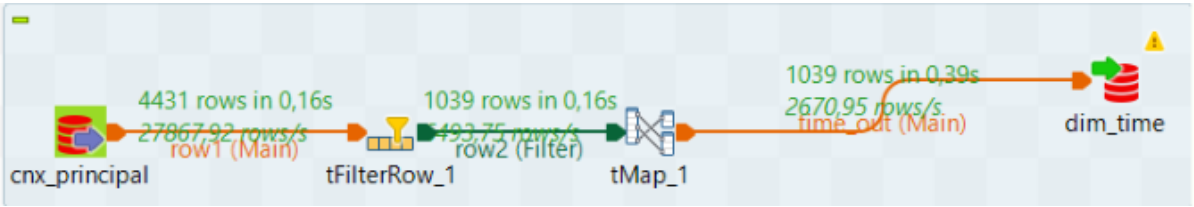
```
select * from dim_product
```

Résultat de requête

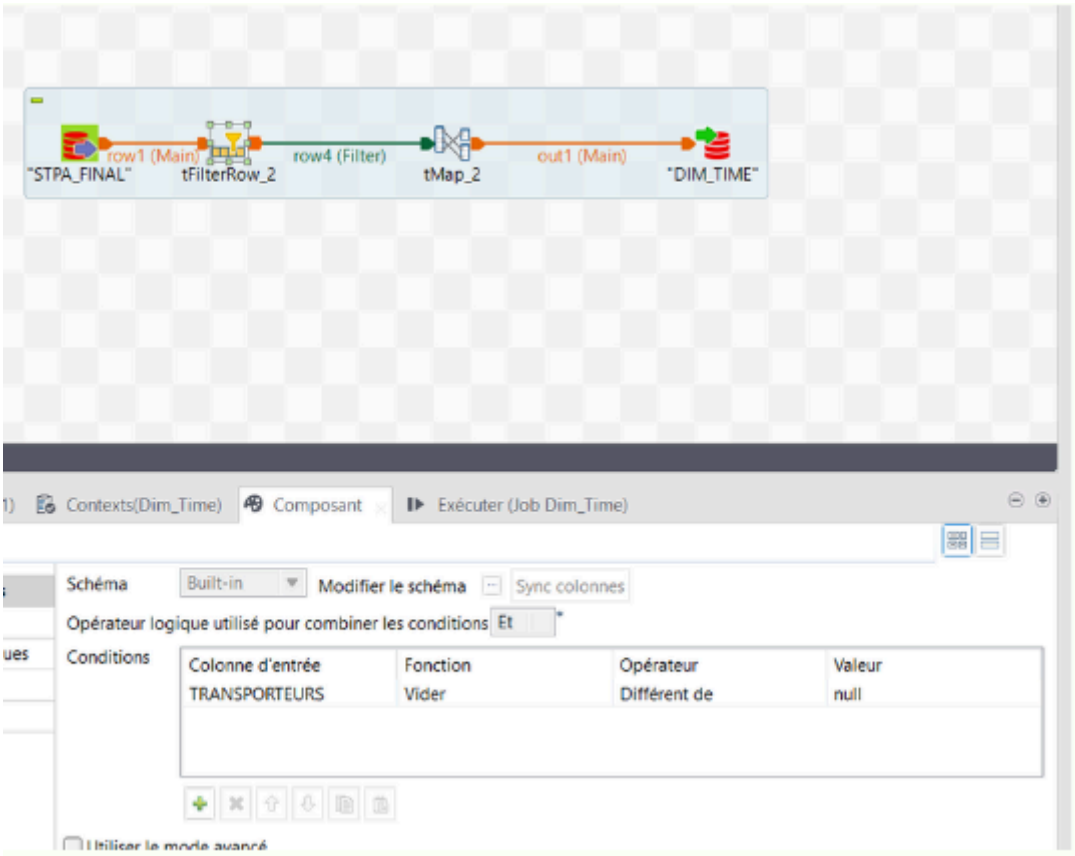
SQL | 50 lignes extraites en 0,004 secondes

	PRODUCT_PK	PRODUCTS	BL_WEIGHT
1	1	PS 50KG	20
2	2	PS 50KG	10
3	3	PS-7 20KG	3
4	4	PS 50KG	20
5	5	PS-7 20KG	2
6	6	PS-7 20KG	5
7	7	SSSE 20KG	5
8	8	PS 50KG	20
9	9	SSSE 20KG	5
10	10	PS 50KG	30
11	11	SSSE 20KG	5
12	12	PS 50KG	20
13	13	PS-7 20KG	3
14	14	SSSE 20KG	10
15	15	SSSE 20KG	1
16	16	PS 50KG	40
17	17	PS 50KG	10
18	18	PS-7 20KG	2
19	19	SSSE 20KG	10
20	20	SON 50 KG	300
21	21	PS-7 20KG	30
22	22	SSSE 20KG	20
23	23	PS 50KG	20
24	24	PS-7 20KG	10

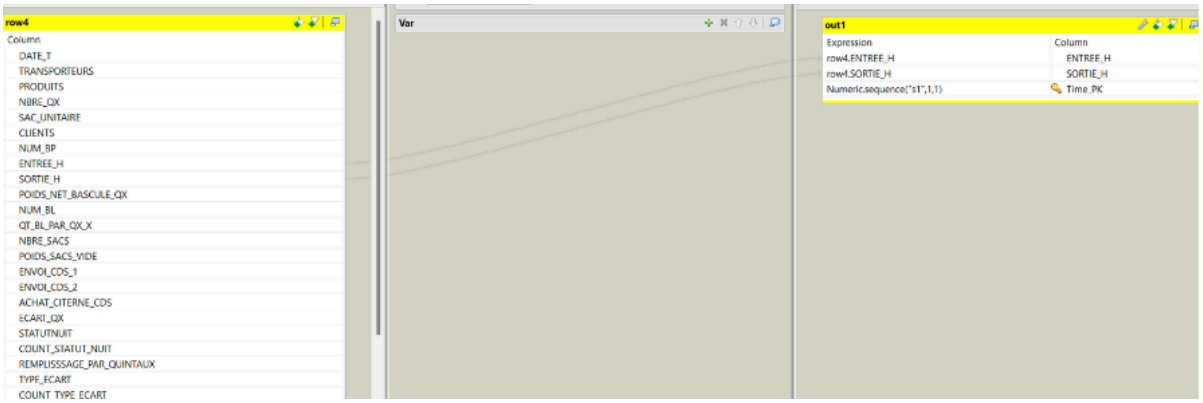
- ***DIM Time :***

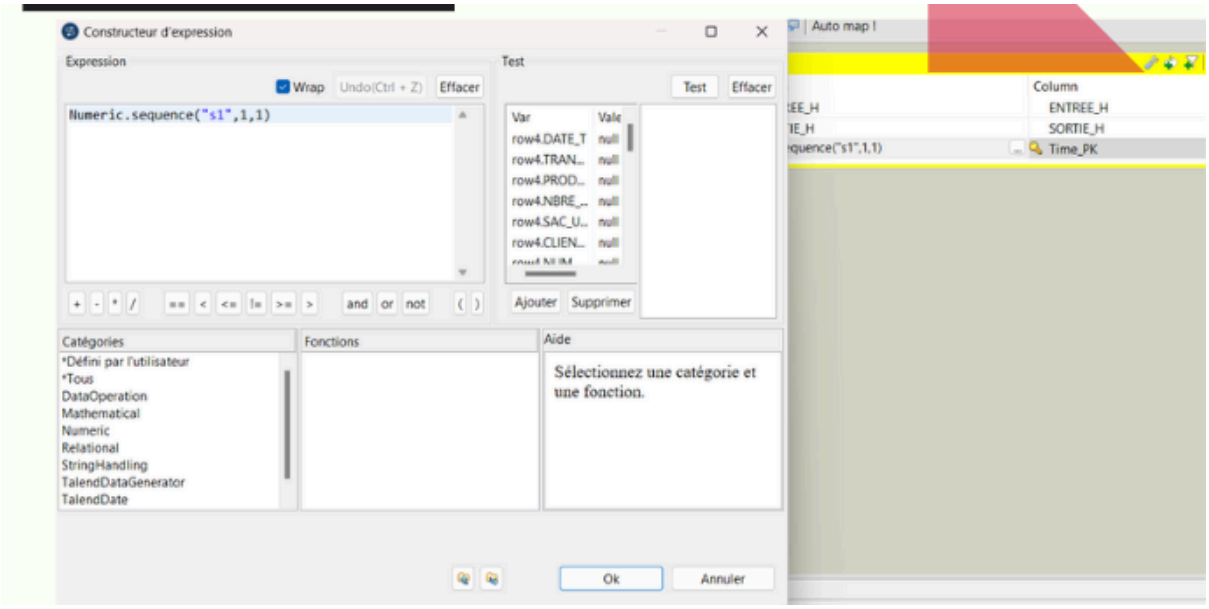


With TFilterRow, we removed the rows where the carriers are null as we do not need them.



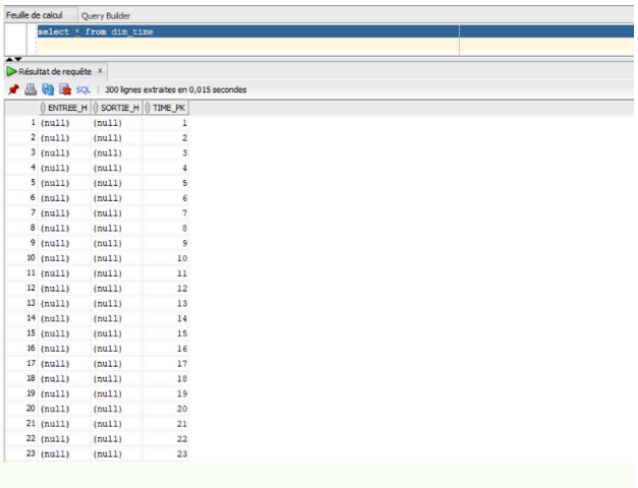
Once finished, with tmap we drag the relevant attributes from the file to the target table "Dim_Time":



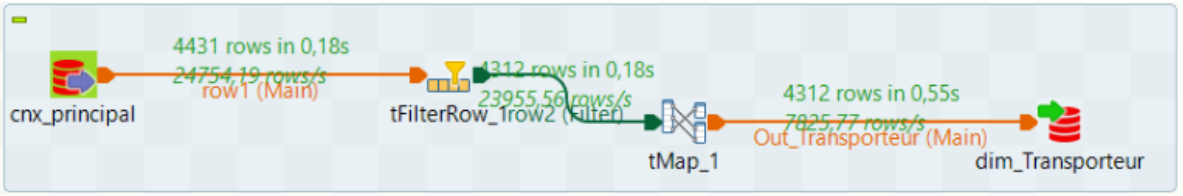


For <<Time_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

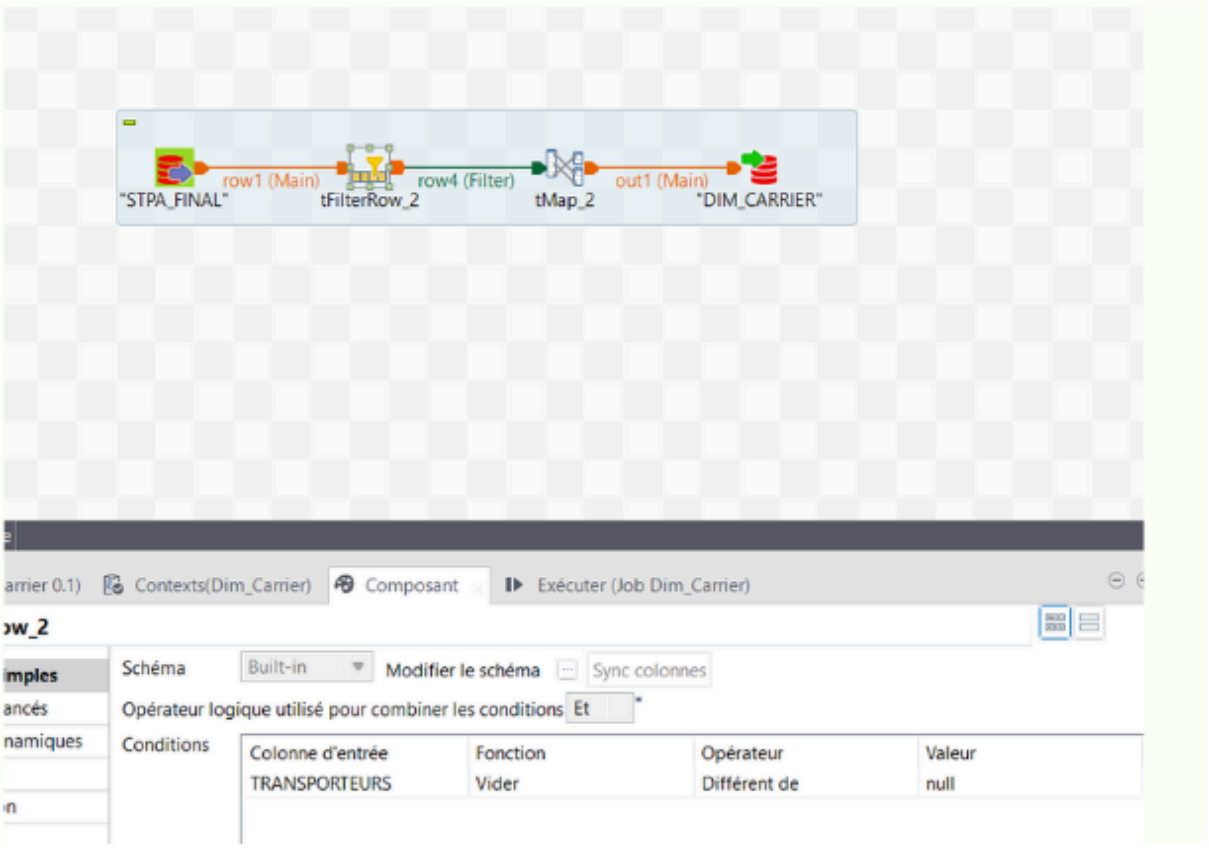
After the execution of this job, here is our "Dim_Time" table created in SQL Developer.



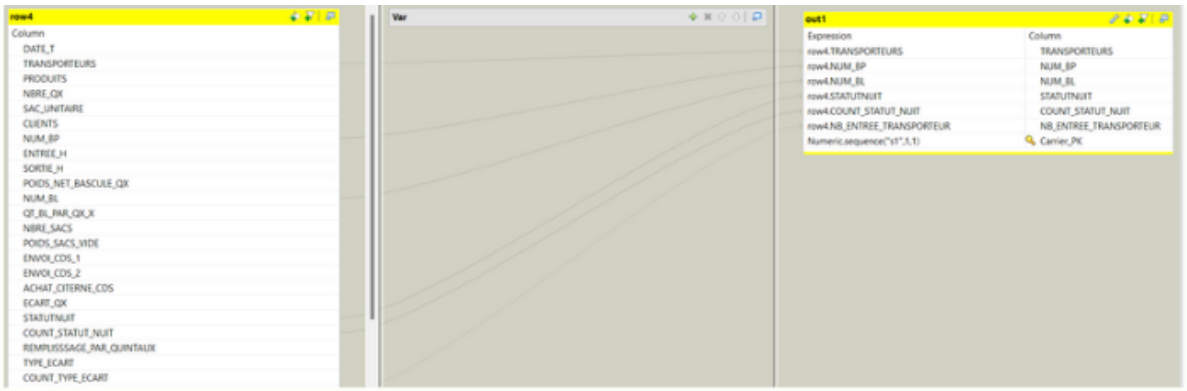
- ***DIM Carrier :***

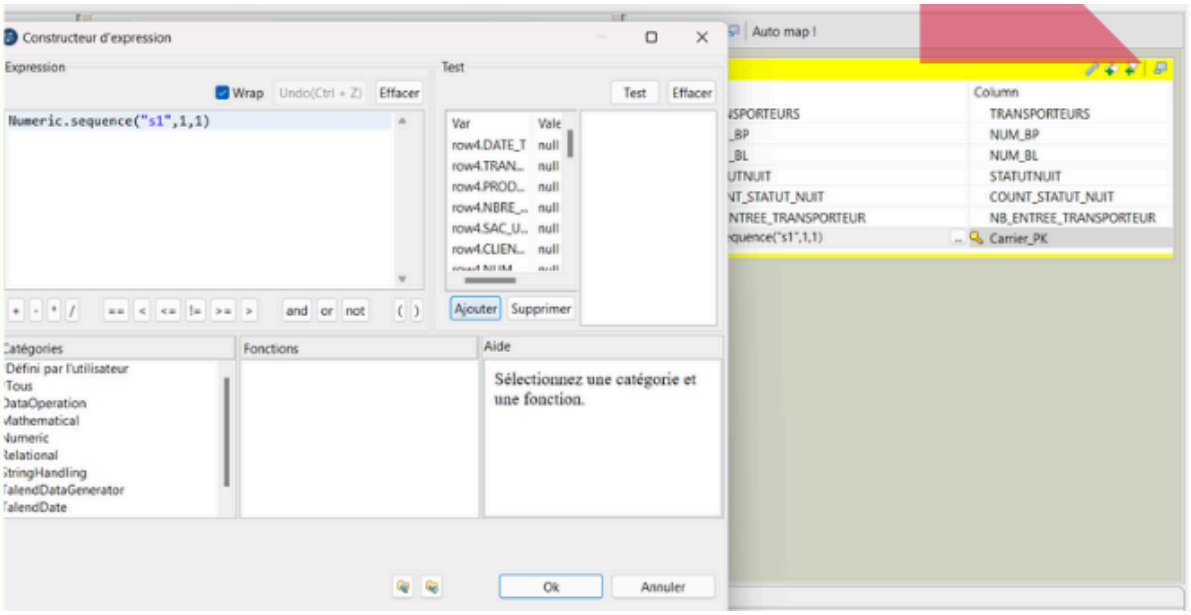


With TFilterRow, we removed the rows where the carriers are null as we do not need them.



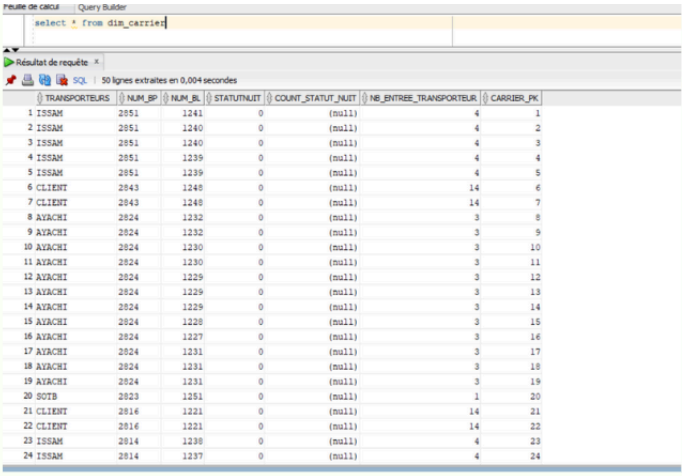
Once finished, with tmap we drag the relevant attributes from the file to the target table "Dim_Carrier":





For <<Carrier_PK >> we used the numeric function “Numeric.sequence(“s1”,1,1)” to have automatic incrementing.

After the execution of this job, here is our "Dim_Carrier" table created in SQL Developer.

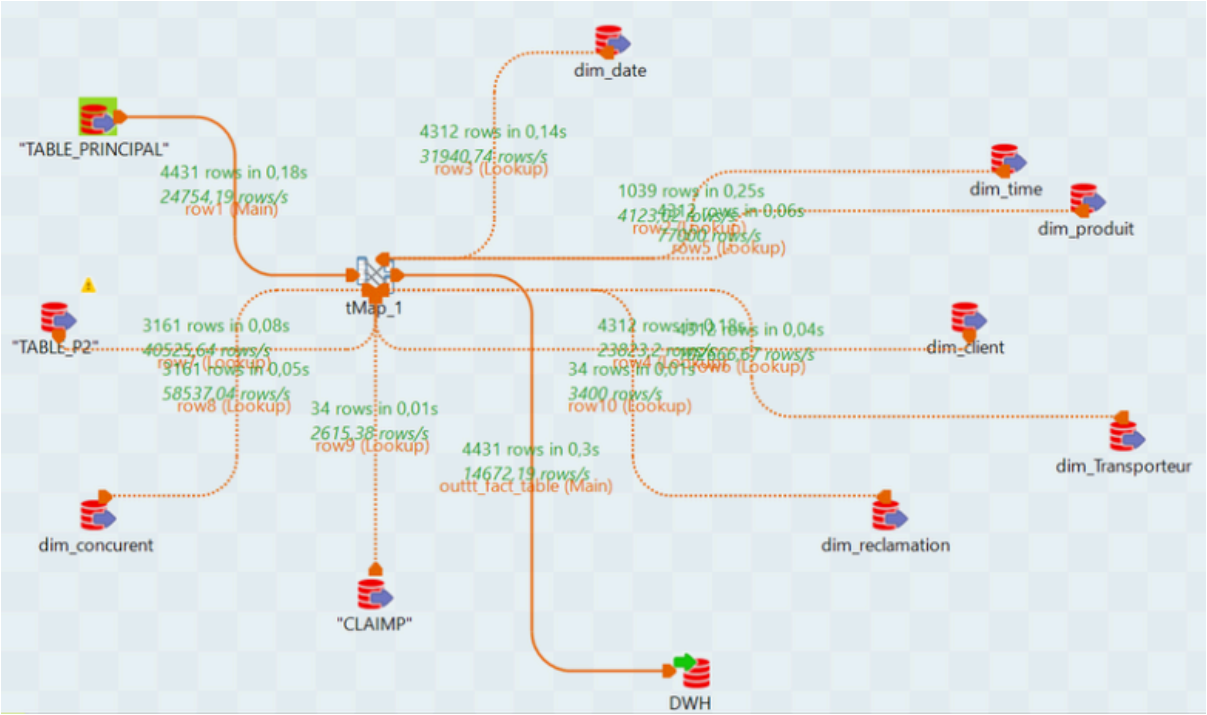


	TRANSPORTEURS	NUM_BP	NUM_BL	STATUTNUIT	COUNT_STATUT_NUIT	NB_ENTREE_TRANSPORTEUR	CARRIER_PK
1	ISSAM	2851	1241	0	(null)	4	1
2	ISSAM	2851	1240	0	(null)	4	2
3	ISSAM	2851	1240	0	(null)	4	3
4	ISSAM	2851	1239	0	(null)	4	4
5	ISSAM	2851	1239	0	(null)	4	5
6	CLIENT	2843	1248	0	(null)	14	6
7	CLIENT	2843	1248	0	(null)	14	7
8	AYACHI	2824	1232	0	(null)	3	8
9	AYACHI	2824	1232	0	(null)	3	9
10	AYACHI	2824	1230	0	(null)	3	10
11	AYACHI	2824	1230	0	(null)	3	11
12	AYACHI	2824	1229	0	(null)	3	12
13	AYACHI	2824	1229	0	(null)	3	13
14	AYACHI	2824	1229	0	(null)	3	14
15	AYACHI	2824	1228	0	(null)	3	15
16	AYACHI	2824	1227	0	(null)	3	16
17	AYACHI	2824	1231	0	(null)	3	17
18	AYACHI	2824	1231	0	(null)	3	18
19	AYACHI	2824	1231	0	(null)	3	19
20	SOTB	2823	1251	0	(null)	1	20
21	CLIENT	2816	1221	0	(null)	14	21
22	CLIENT	2816	1221	0	(null)	14	22
23	ISSAM	2814	1238	0	(null)	4	23
24	ISSAM	2814	1237	0	(null)	4	24

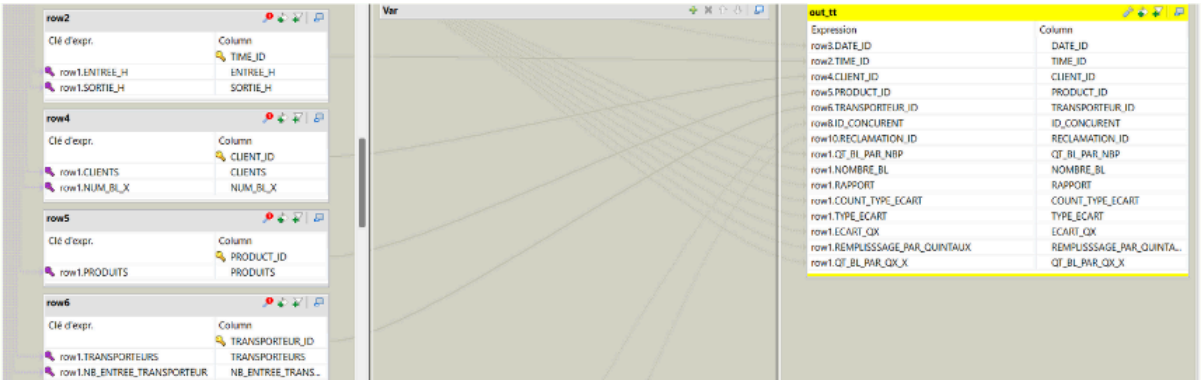
Step 3 : Feeding Fact table

- ***Fact_Delivery:***

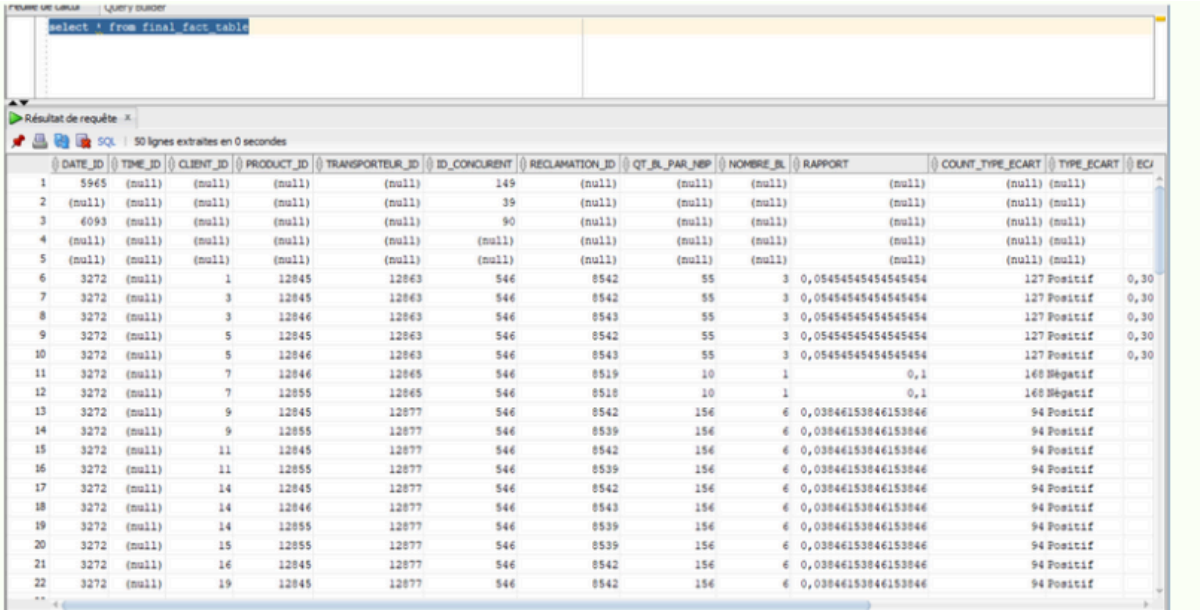
We fill the fact table by integrating relevant data from a variety of sources, creating a consolidated database that will be used for future analyses.



We created joins between the different dimensions using the TMap component.

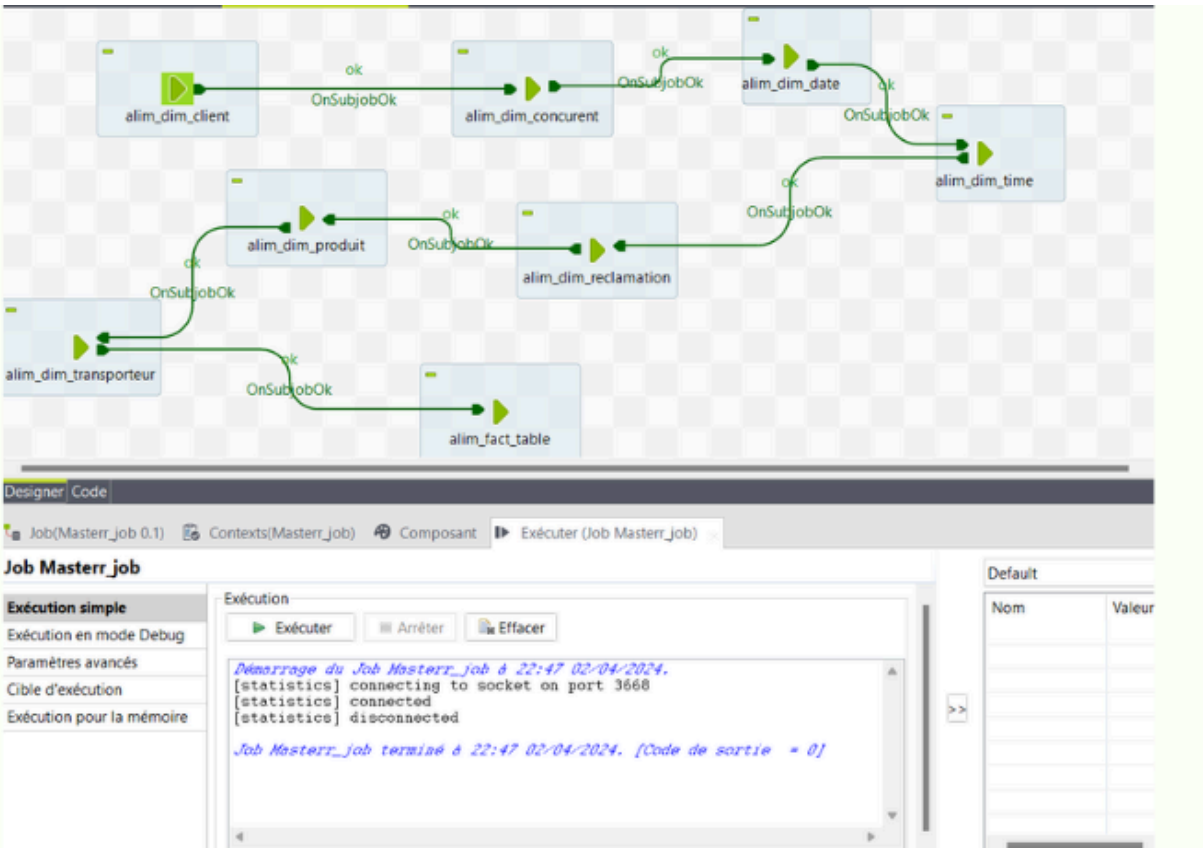


In this step, we're going to load our target table which is the table made with the different dimensions already defined earlier:

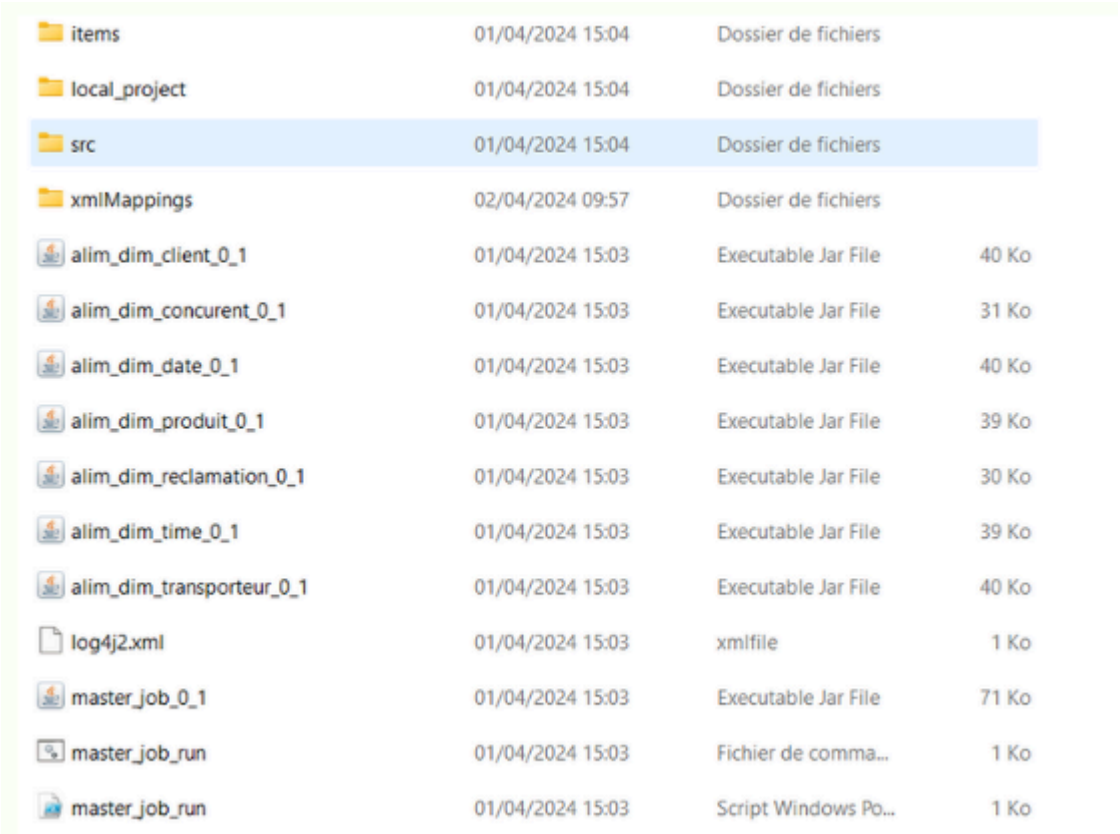


Step 4 : Export Database Connection as Context

The process involves creating a master job to oversee and execute individual tasks sequentially. This master job integrates with the main database connection and source data paths.



If you erase the content of the table and then execute it again, the table will be filled after executing the master_job.



items	01/04/2024 15:04	Dossier de fichiers	
local_project	01/04/2024 15:04	Dossier de fichiers	
src	01/04/2024 15:04	Dossier de fichiers	
xmlMappings	02/04/2024 09:57	Dossier de fichiers	
alim_dim_client_0_1	01/04/2024 15:03	Executable Jar File	40 Ko
alim_dim_concurent_0_1	01/04/2024 15:03	Executable Jar File	31 Ko
alim_dim_date_0_1	01/04/2024 15:03	Executable Jar File	40 Ko
alim_dim_produit_0_1	01/04/2024 15:03	Executable Jar File	39 Ko
alim_dim_reclamation_0_1	01/04/2024 15:03	Executable Jar File	30 Ko
alim_dim_time_0_1	01/04/2024 15:03	Executable Jar File	39 Ko
alim_dim_transporteur_0_1	01/04/2024 15:03	Executable Jar File	40 Ko
log4j2.xml	01/04/2024 15:03	xmlfile	1 Ko
master_job_0_1	01/04/2024 15:03	Executable Jar File	71 Ko
master_job_run	01/04/2024 15:03	Fichier de comma...	1 Ko
master_job_run	01/04/2024 15:03	Script Windows Po...	1 Ko

3. *Power BI*

3.1 ALCO

<u>Step 1 : Importing the Data Warehouse :</u>

For the realization of the Dashboard, we first start by importing the data from the database, the data we have stored and prepared at the level of our DW will be used to create clear dashboards.

Step 2 : Creation of KPIs :

This is because KPIs in power BI are essential metrics that track and evaluate the performance of a specific company or process.

Cp :

```
CP_Humidity =
VAR USL = 15
VAR LSL = 13
VAR Ecart_Type_Humidite = 'public FAIT_VENTE'[ecart_humudity]

RETURN
    (USL - LSL) / (6 * Ecart_Type_Humidite)
```

CPk :

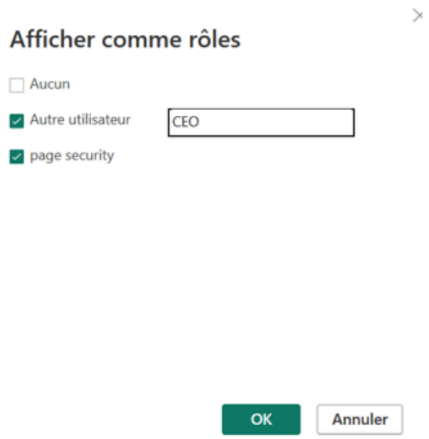
```
CPK_HUmidity = VAR USL = 15
VAR LSL = 13
VAR Moyenne_Humidite = 'public FAIT_VENTE'[moyenne_Humidity]
VAR Ecart_Type_Humidite = 'public FAIT_VENTE'[ecart_humudity]

RETURN
    MIN(
            (USL - Moyenne_Humidite) / (3 * Ecart_Type_Humidite),
            (Moyenne_Humidite - LSL) / (3 * Ecart_Type_Humidite)
    )
```

AVG :

```
Avg_Price_Diver_Grain_per_Year =
CALCULATE(AVERAGE('public Dim_Price_MP'[Price]),FILTER('public Dim_Price_MP','public
Dim_Price_MP'[MP] = "Diver Grain"))
```

```
Avg_Price_Son_ble_per_Year =
CALCULATE(AVERAGE('public Dim_Price_MP'[Price]),FILTER('public Dim_Price_MP','public
Dim_Price_MP'[MP] = "Son de blé"))
```

Step 3 : Creating dashboards:



- ***Step Dashboard Security with RLS***

Our dashboard homepage leverages Row-Level Security (RLS) to restrict access to data. This ensures only authorized users see relevant information.

- Benefits of RLS
- Protects sensitive data
- Enhances compliance

This approach strengthens data security and helps meet regulatory requirements.

- ***Dashboard for Sales Manager:***

The dashboard provided to production managers includes a comprehensive set of components that effectively present key performance indicators (KPIs) and data visualizations, allowing sales managers to gain valuable insights into various aspects of their team's performance.

- ***KPI Components for Threshold Analysis:***

- CP (Capability Index): This metric indicates the process capability compared to the specification limits. It measures the process's ability to produce products within acceptable quality standards.
- CPK (Capability Index to Six Sigma): This metric represents the process capability in terms of Six Sigma units.

A CPK value greater than 1 indicates that the process is capable of producing products within the specification limits with a high degree of confidence.

Thresholds and Acceptance Zones: The dashboard incorporates thresholds and acceptance zones for each KPI, providing a clear visual representation of acceptable performance levels. This enables sales managers to quickly identify areas that require attention and corrective actions.

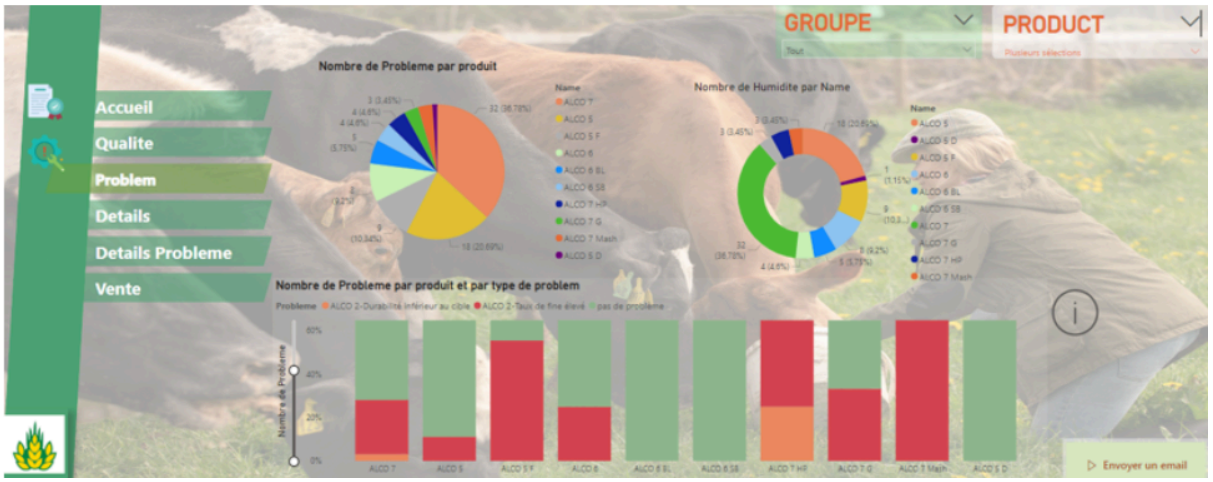
- ***Histogram Charts for Quality Visualization:***

Product Composition Quality: Histogram charts are employed to visualize the distribution of product composition metrics, such as protein, fat, and moisture content. This visual representation allows sales managers to assess the quality consistency of each product.

Product and Group Filters: The dashboard provides filters for products and groups, enabling sales managers to drill down into specific product categories and analyze composition quality on a granular level.

- ***CART for KPI Distribution:***

KPI-Driven CARTS: CARTS are utilized to display the geographical distribution of KPIs, such as sales volume and market share. This visualization helps sales managers identify regional trends and potential growth opportunities.



The provided dashboard effectively visualizes product-related issues, offering valuable insights into the distribution and types of problems encountered across different products.

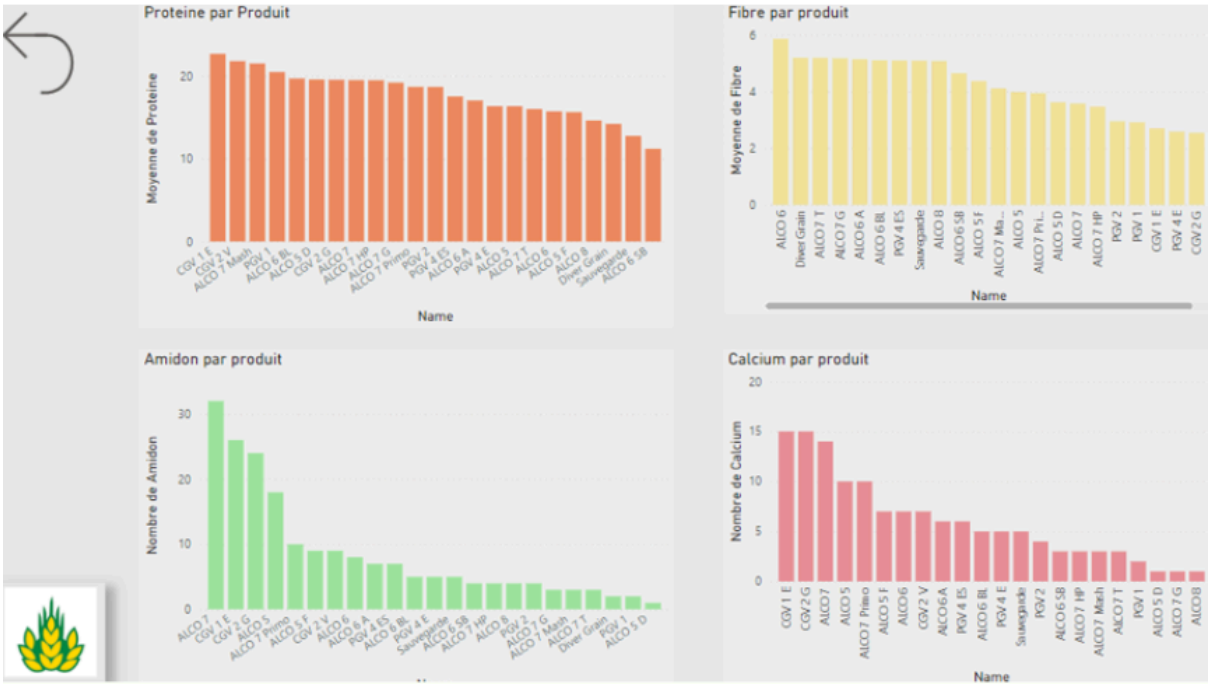
Pie Chart: Product Distribution of Issues

- Product-Wise Issue Count: This pie chart represents the number of issues reported for each product, providing a clear overview of the overall issue distribution.
- Issue Identification: By analyzing the pie chart, users can quickly identify products that are disproportionately affected by issues, allowing for targeted troubleshooting and improvement efforts.

Stacked Histogram: Issue Types by Product

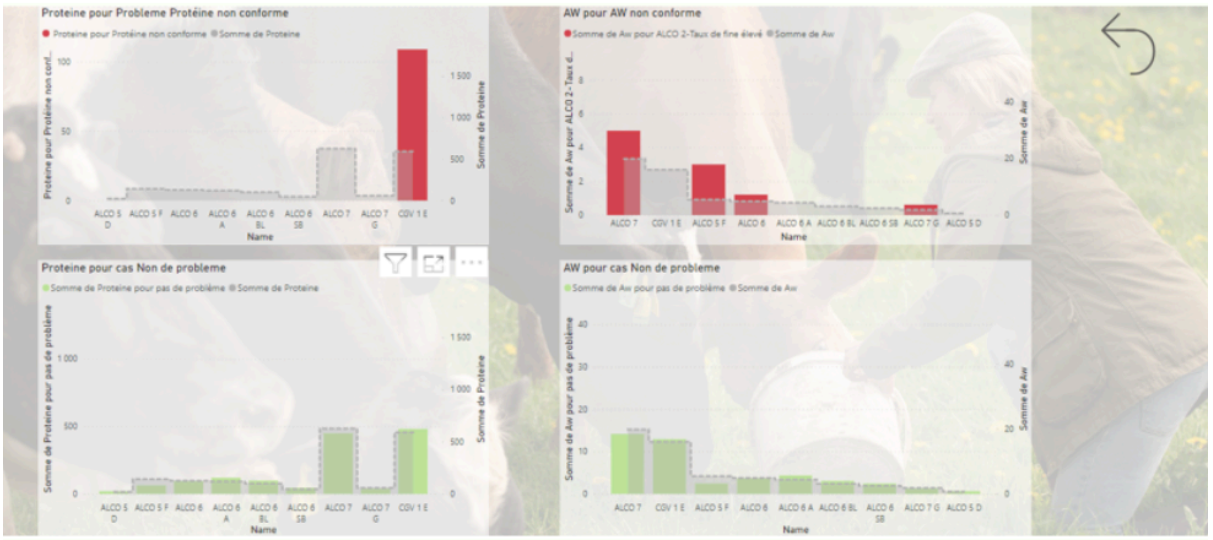
- Product-Specific Issue Breakdown: This stacked histogram categorizes issues based on their type and presents the number of each type of issue for each product.

- Issue Trend Analysis: The stacked histogram enables users to identify recurring issue types for specific products, facilitating trend analysis and potential root cause identification.



Stacked Histogram: Product Composition Breakdown

- Component-Wise Distribution: For each product, a stacked histogram is presented, illustrating the relative proportions of protein, starch, and calcium within that product.
- Product Comparison: The histograms are arranged horizontally, allowing users to easily compare the composition of different products side-by-side.
- Product Selection: Users can select specific products to focus on the composition of their choice.



1. Grouped Bar Chart: Protein Issue Distribution

- Protein Issue Categories: The chart categorizes protein issues into specific groups, such as low protein content, protein denaturation, and protein contamination.
- Issue Severity: The bars are color-coded, with red representing severe issues and green indicating minor or non-existent issues.
- Product Comparison: The chart allows users to compare the distribution of protein issues across different products.

2. Stacked Histogram: Protein Issue Breakdown by Product

- Product-Specific Issue Analysis: For each product, a stacked histogram is presented, illustrating the relative proportions of different protein issue categories within that product.
- Issue Trend Identification: The stacked histogram enables users to identify recurring protein issue types for specific products, facilitating trend analysis and potential root cause identification.

AW Component Analysis

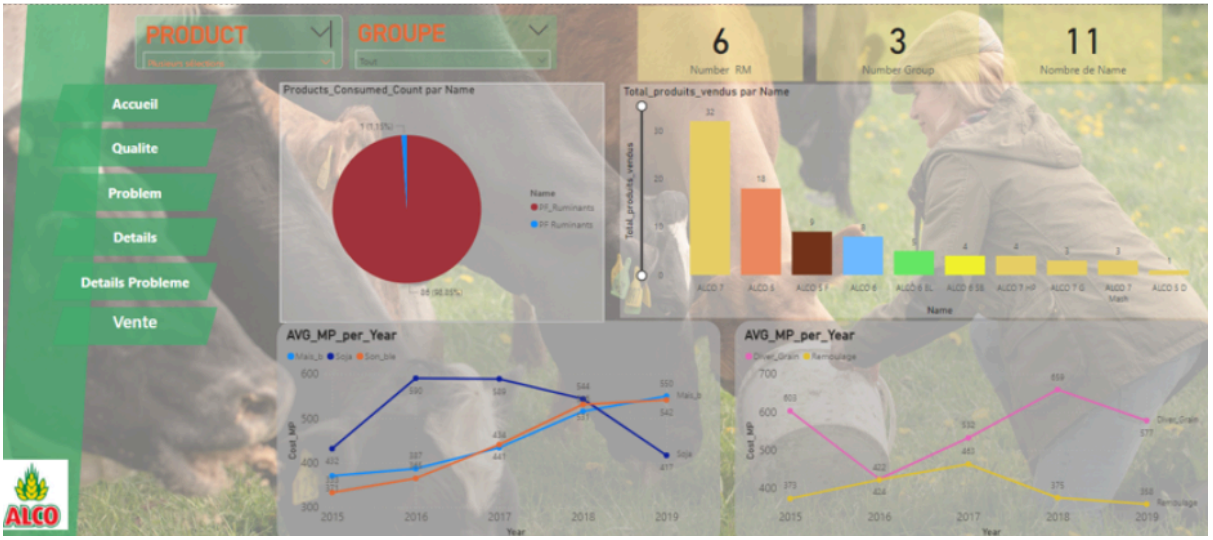
1. Grouped Bar Chart: AW Component Issue Distribution

- AW Component Issue Categories: The chart categorizes AW component issues into specific groups, such as AW component quality, AW component consistency, and AW component integration.
- Issue Severity: The bars are color-coded, with red representing severe issues and green indicating minor or non-existent issues.
- Product Comparison: The chart allows users to compare the distribution of AW component issues across different products.

2. Stacked Histogram: AW Component Issue Breakdown by Product

- Product-Specific Issue Analysis: For each product, a stacked histogram is presented, illustrating the relative proportions of different AW component issue categories within that product.
- Issue Trend Identification: The stacked histogram enables users to identify recurring AW component issue types for specific products, facilitating trend analysis and potential root cause identification.

- Sales Manager Dashboard:



The provided dashboard page effectively visualizes product consumption patterns, cost trends, and top product groups, offering valuable insights into sales performance, cost optimization, and product strategy.

Pie Chart: Top Product Group Consumption

- Product Group Distribution: This pie chart represents the percentage of consumption for each product group, providing a clear overview of the overall consumption distribution.
- Group Identification: By analyzing the pie chart, users can quickly identify the product groups that account for the majority of consumption, allowing for targeted marketing and sales efforts.

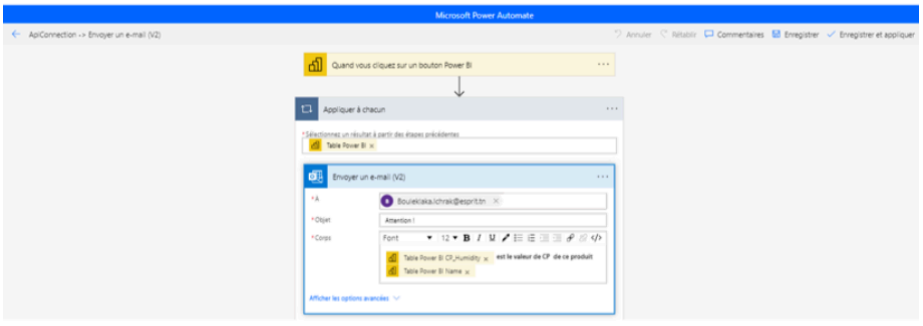
Stacked Histogram: Top Consumed Products by Year

- Product-Wise Consumption Trends: This stacked histogram categorizes products based on their consumption and presents the consumption of each product for each year.
- Consumption Trend Analysis: The stacked histogram enables users to identify trends in product consumption over time, allowing for proactive inventory management and product development decisions.

Line Chart: Cost Variation by Year

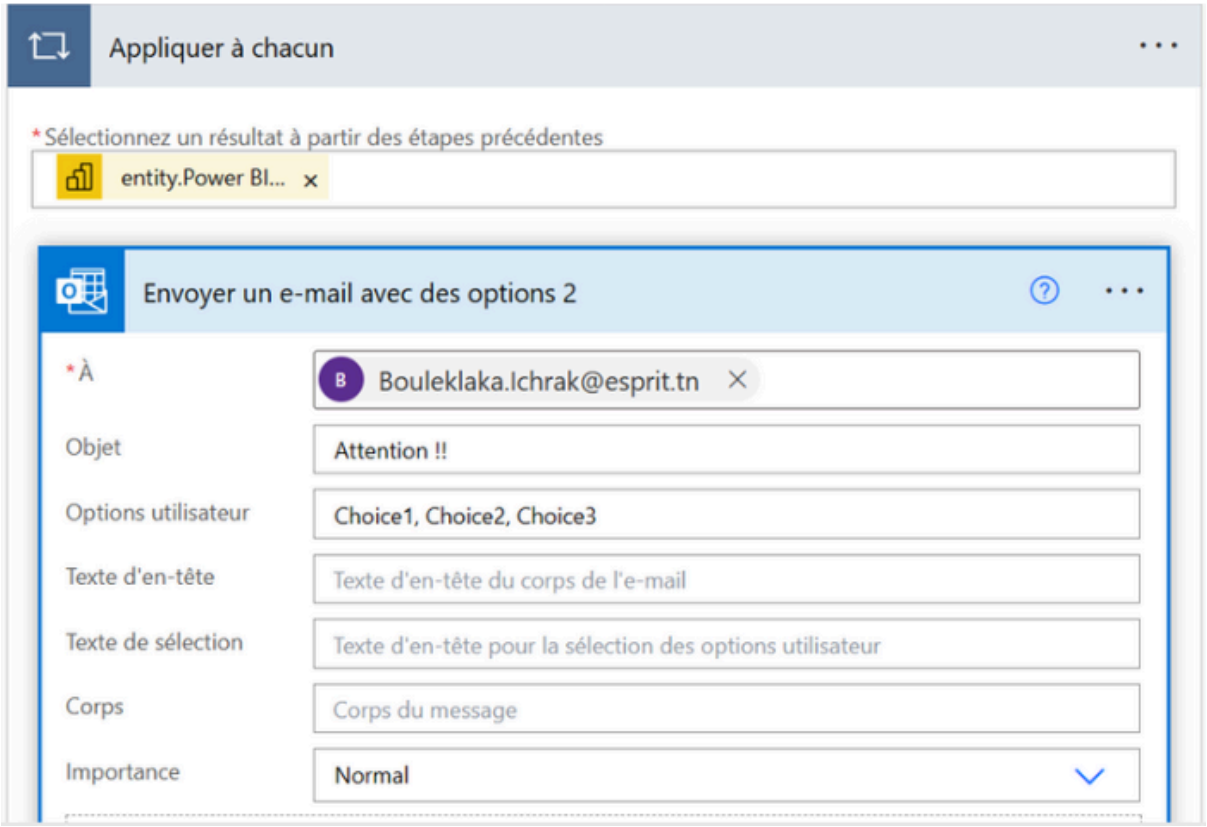
- Cost Trend Analysis: This line chart depicts the variation in production costs over time, allowing users to identify cost-saving opportunities and assess the impact of cost fluctuations on profitability.
- Year-on-Year Comparison: The line chart facilitates year-on-year comparisons of production costs, enabling users to track cost trends and make informed decisions

Step 4 : Power Automate



Automated CP/CPK Alerts with Power Automate

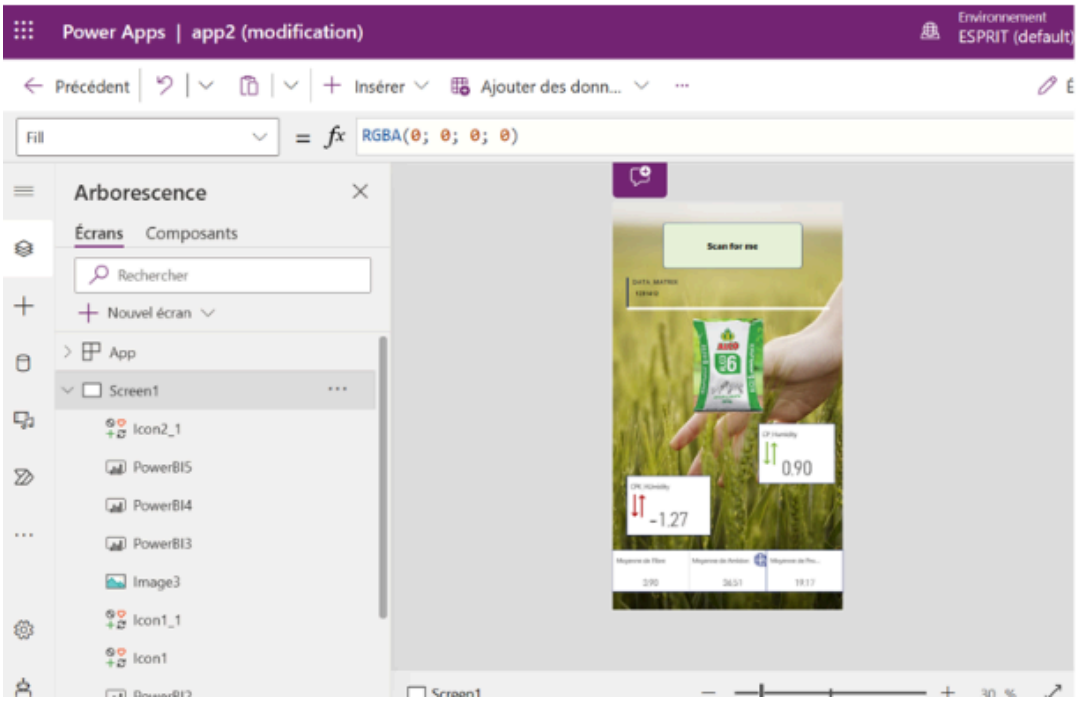
This report details the creation of an automated email alert system using Power Automate. The system triggers emails whenever Capability Index (CP) or Capability Index to Six Sigma (CPK) values fall outside acceptable ranges, ensuring timely notification for process quality maintenance.



- Automated Data Change Alerts with Power Automate

This report details the development of an automated email notification system using Power Automate. The system triggers emails whenever significant changes occur in your data, ensuring relevant personnel are promptly informed for improved data management and decision-making.

Step 5 : Power Apps



QR Code-Powered Product Analysis with Power Apps and Power BI

This report presents a QR code-based product scanning application that utilizes Power Apps and Power BI to provide real-time CP, CPK, and component average insights, enhancing product quality monitoring and decision-making.

Key Features:

- Streamlined QR code scanning for data collection
- Interactive Power BI dashboards for data visualization
- Real-time product quality insights
- Mobile accessibility for on-the-go analysis

Benefits:

- Improved data accuracy and efficiency
- Enhanced product quality monitoring
- Proactive identification of potential quality issues

- Informed decision-making for product improvement

3.2 STPA

Step 1 : Importing the Data Warehouse :

For the realization of the Dashboad, we first start by importing the data from the database, the data we have stored and prepared at the level of our DW will be used to create clear dashboards.

Step 2 : Creation of KPIs:

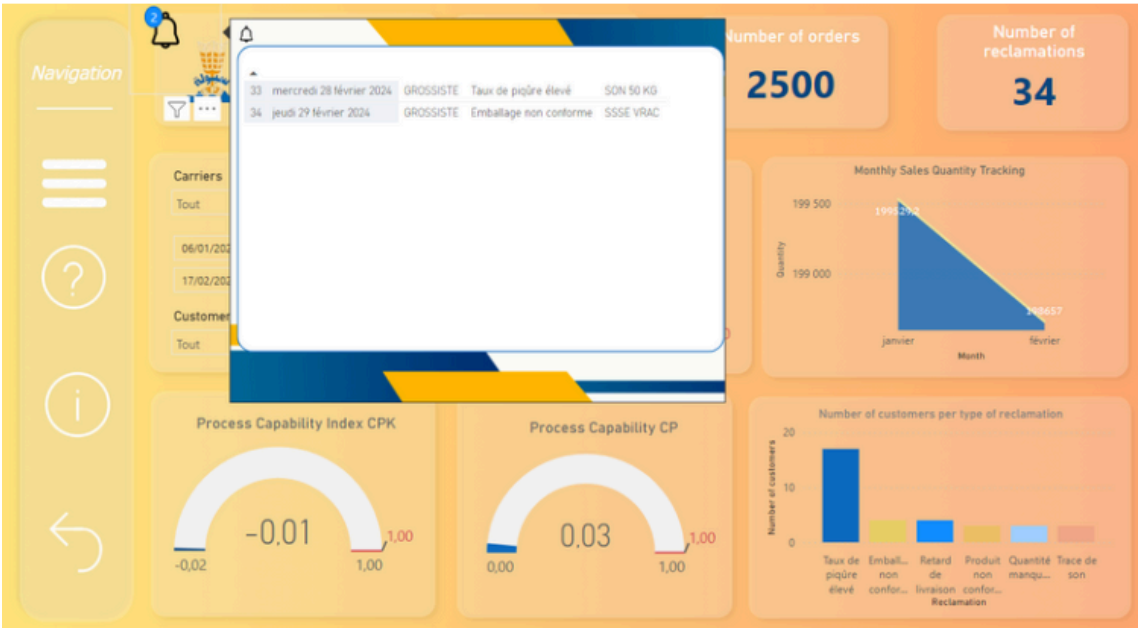
This is because KPIs in power BI are essential metrics that track and evaluate the performance of a specific company or process.

```
CPK =

VAR MeanColumn = AVERAGE(fact_table[ECART_QX])

VAR StdDevColumn = STDEV.P(fact_table[ECART_QX])

RETURN

  MIN(

    ((3 - MeanColumn) / (3 * StdDevColumn)),

    ((MeanColumn - 1) / (3 * StdDevColumn)))
```

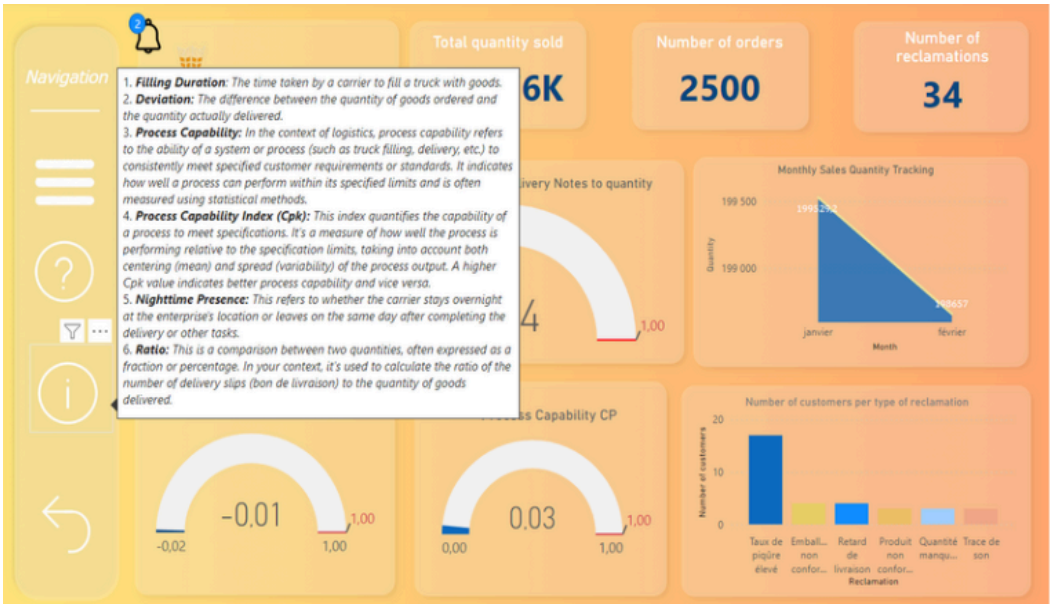
CP = (3 - 1) / (6 * STDEVX.P('fact_table', 'fact_table'[ECART_QX]))

Step 3 : Creating dashboards:

- ***Full Insights***



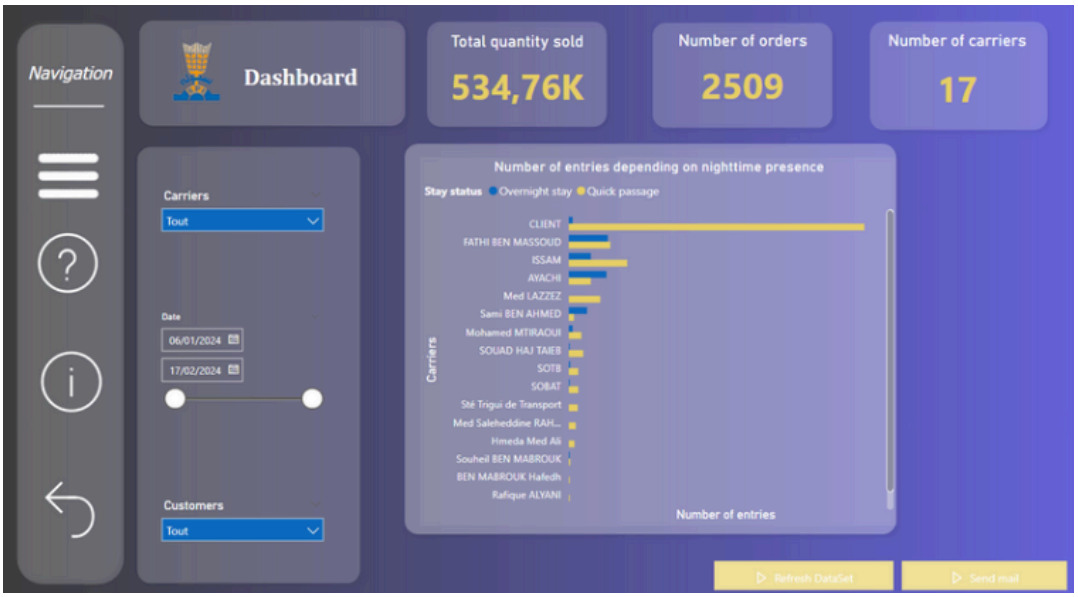

```
1  Notification Highlight =
2  IF(
3      OR(
4          SELECTEDVALUE('Feuil1 (3)'[Date de réclamation]) = DATE(2024, 2, 28),
5          SELECTEDVALUE('Feuil1 (3)'[Date de réclamation]) = DATE(2024, 2, 29)
6      ),
7      "rgb(240, 243, 247)"
8  )
9
```



The "Full Insights" dashboard on Sboula offers a comprehensive overview of key metrics and performance indicators. Positioned on the left is a convenient pop-up menu providing easy access to various functions and filters. The dashboard prominently displays essential metrics such as CP, CPK, and the delivery notes to quantity ratio, all within acceptable norms.

Among the highlighted insights are the total quantity sold, number of orders processed, and instances of complaints. A monthly sales trend curve and a histogram detailing complaint types per customer are also featured, aiding in trend analysis and issue identification. Additionally, by clicking on the small "i" icon, users can access supplementary information to gain deeper insights into specific data points or metrics, enhancing decision-making capabilities. In the top left corner, a notification panel dynamically showcases the two most recent complaints, ensuring immediate attention to emerging concerns.

- ***Carriers Dashboard***



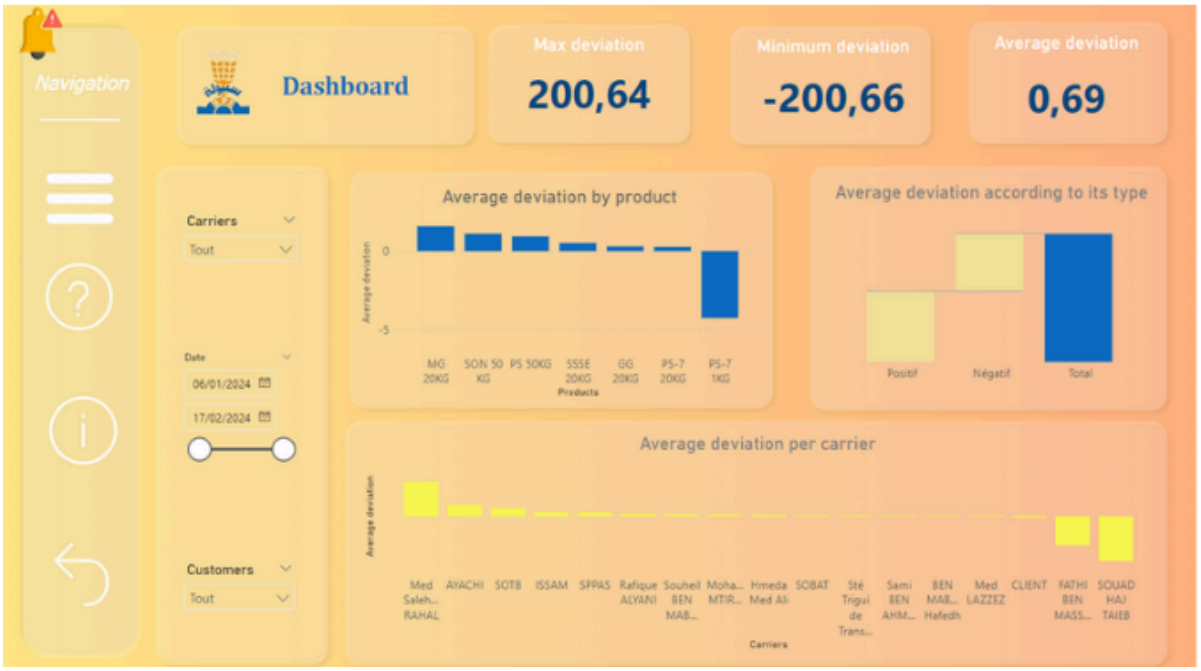
The "Carriers" dashboard on Sboula offers a comprehensive overview of key metrics and insights related to transportation logistics. Positioned conveniently on the left-hand side is a user-friendly pop-up menu, providing easy access to various functions and filters, enhancing the user experience. This dashboard showcases essential indicators such as total quantity sold, number of orders processed, and the count of transporters engaged. A standout feature of this dashboard is the grouped bar histogram, which effectively illustrates the distribution of entries for each transporter based on residency status. This visualization enables users to quickly assess the performance and distribution of transporters across different residency statuses, facilitating strategic decision-making. With customizable filters, users can tailor their analysis to specific parameters, ensuring relevant insights are readily available.

- ***Customers Dashboard***

The "Customers" dashboard on Sboula presents a comprehensive view of customer-related metrics and insights essential for strategic decision-making. Positioned conveniently on the left-hand side is a user-friendly pop-up menu, providing seamless navigation to various functions and filters, enhancing user interaction. This dashboard prominently displays key indicators such as total quantity sold, number of orders processed, and the count of unique customers engaged. Noteworthy visualizations include a histogram detailing the quantity sold per product, offering insights into product popularity and sales trends. Additionally, a pie chart illustrates the quantity sold per customer, providing a clear understanding of customer preferences and contribution to overall sales. Another standout feature is the stacked bar histogram, which depicts the distribution of quantity sold per product for each individual customer, facilitating targeted marketing strategies and personalized customer engagement. With customizable filters, users can refine their analysis based on specific parameters,

ensuring actionable insights are readily available. The "Customers" dashboard on Sboula empowers users to optimize customer relationships, tailor marketing efforts, and drive business growth effectively.

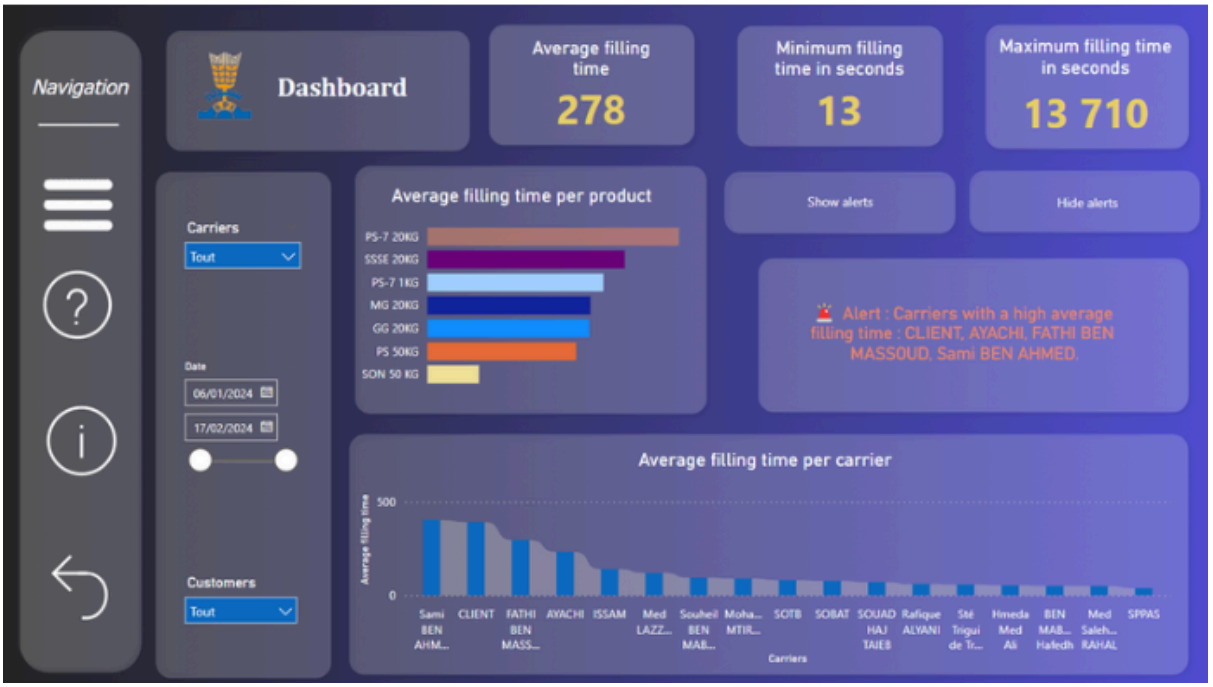
- ***Deviation Dashboard***




```
Alertes =
VAR SeuilEcart = 5
VAR TransporteursEnEcart =
    FILTER(
        VALUES(dim_transporteur[TRANSPORTEURS]),
        CALCULATE(AVERAGE(fact_table[ECART_QX])) > SeuilEcart || CALCULATE(AVERAGE(fact_table[ECART_QX])) < -SeuilEcart
    )
VAR TransporteursEnEcartNoms =
    CONCATENATEX(
        TransporteursEnEcart,
        dim_transporteur[TRANSPORTEURS],
        ", "
    )
VAR MessageAlerte =
    IF (
        COUNTROWS(TransporteursEnEcart) > 0,
        "🚨 Alerte : Transporteurs avec un écart moyen élevé : "  & TransporteursEnEcartNoms &"." ,
        BLANK()
    )
RETURN
    MessageAlerte
```

The "Deviation" dashboard on Sboula offers a comprehensive insight into the performance metrics crucial for quality control and optimization. Positioned conveniently on the left is a user-friendly pop-up menu, providing easy access to various functions and filters, enhancing user experience and navigation. This dashboard prominently displays key indicators such as maximum deviation, average deviation, and minimum deviation, allowing users to monitor performance against set standards effectively. One of the standout features is the histogram illustrating the average deviation per product, providing insights into product quality and consistency. Another histogram showcases the average deviation per transporter, enabling users to identify potential areas for improvement in transportation logistics. Additionally, a cascade chart visualizes the average deviation categorized by type (positive or negative), offering a comprehensive overview of deviation trends and patterns. Furthermore, an alert mechanism positioned at the top left corner dynamically displays the names of transporters with significantly high or low deviations. This alert system, powered by DAX, ensures timely identification of outliers and enables proactive intervention to address deviations effectively. With customizable filters, users can refine their analysis based on specific parameters, ensuring actionable insights are readily available. The "Deviation" dashboard on Sboula empowers users to optimize quality control processes, improve operational efficiency, and ultimately enhance overall performance.

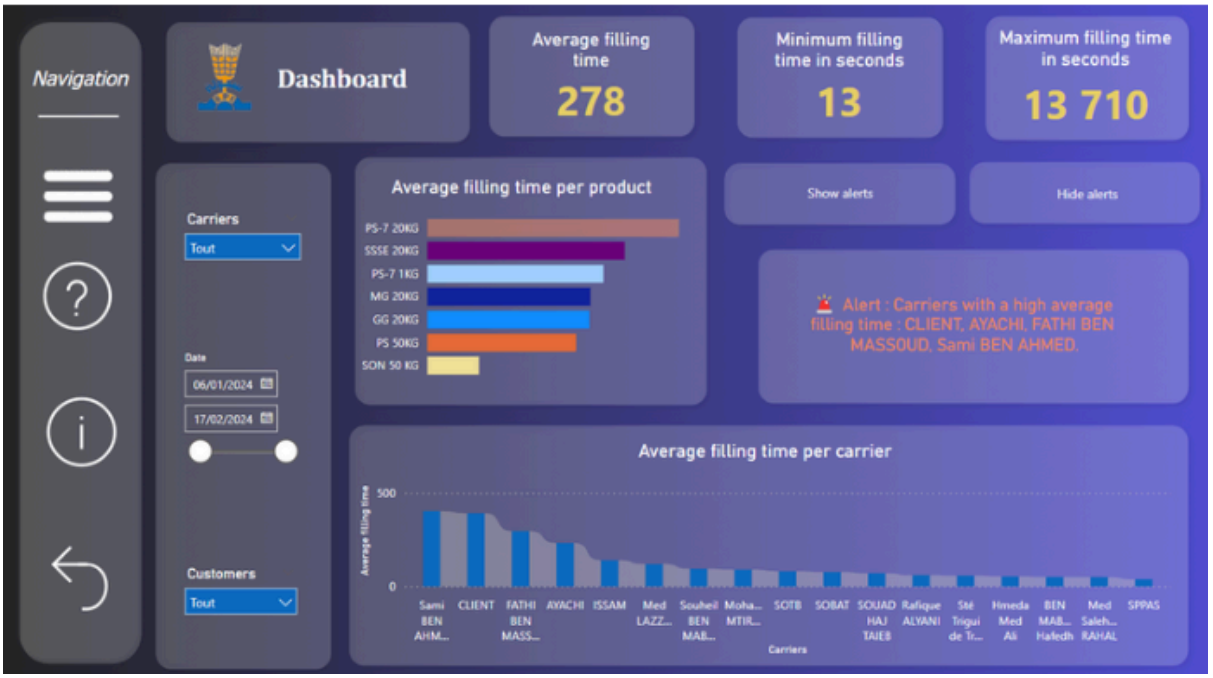
- ***Filling time Dashboard***



The "Deviation" dashboard on Sboula offers a comprehensive insight into the performance metrics crucial for quality control and optimization. Positioned conveniently on the left is a user-friendly pop-up menu, providing easy access to various functions and filters, enhancing user experience and navigation. This dashboard prominently displays key indicators such as maximum deviation, average deviation, and minimum deviation, allowing users to monitor performance against set standards effectively. One of the standout features is the histogram illustrating the average deviation per product, providing insights into product quality and consistency. Another histogram showcases the average deviation per transporter, enabling users to identify potential areas for improvement in transportation logistics. Additionally, a cascade chart visualizes the average deviation categorized by type (positive or negative), offering a comprehensive overview of deviation trends and patterns. Furthermore, an alert mechanism positioned at the top left corner dynamically displays the names of transporters with significantly high or low deviations. This alert system, powered by DAX, ensures timely identification of outliers and enables proactive intervention to address deviations effectively.

With customizable filters, users can refine their analysis based on specific parameters, ensuring actionable insights are readily available. The "Deviation" dashboard on Sboula empowers users to optimize quality control processes, improve operational efficiency, and ultimately enhance overall performance.

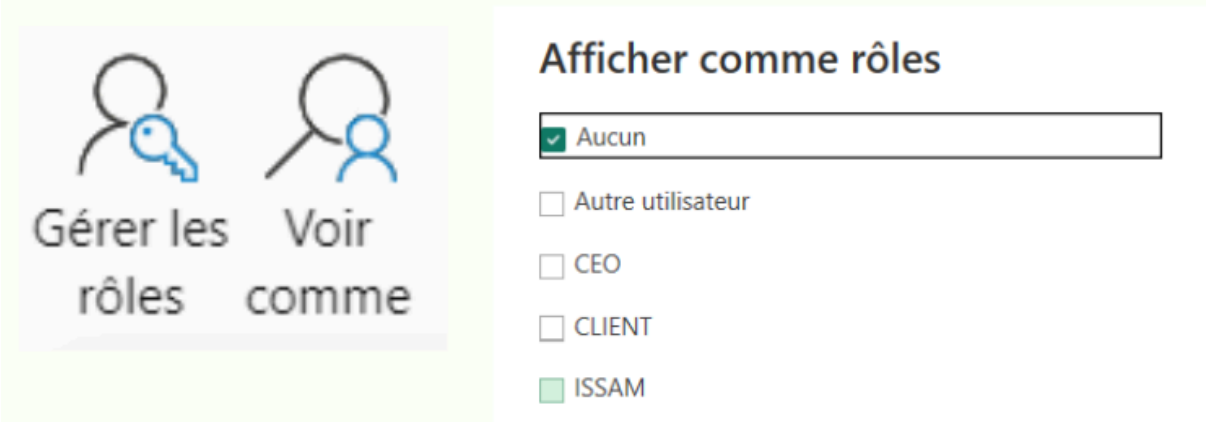
- ***Filling time Dashboard***



```
Alertees =
VAR SeuilEcart = 180
VAR TransporteursEnEcart =
    FILTER(
        VALUES(dim_transporteur[TRANSPORTEURS]),
        CALCULATE(AVERAGE(fact_table[REMPLISSSAGE_PAR_QUINTAUX])) > SeuilEcart || CALCULATE(AVERAGE(fact_table[REMPLISSSAGE_PAR_QUINTAUX])) <
        -SeuilEcart
    )
VAR TransporteursEnEcartNoms =
    CONCATENATEX(
        TransporteursEnEcart,
        dim_transporteur[TRANSPORTEURS],
        ", "
    )
VAR MessageAlerte =
    IF (
        COUNTROWS(TransporteursEnEcart) > 0,
        "⚠ Alert : Carriers with a high average filling time : "  & TransporteursEnEcartNoms &"." ,
        BLANK()
    )
RETURN
    MessageAlerte
```

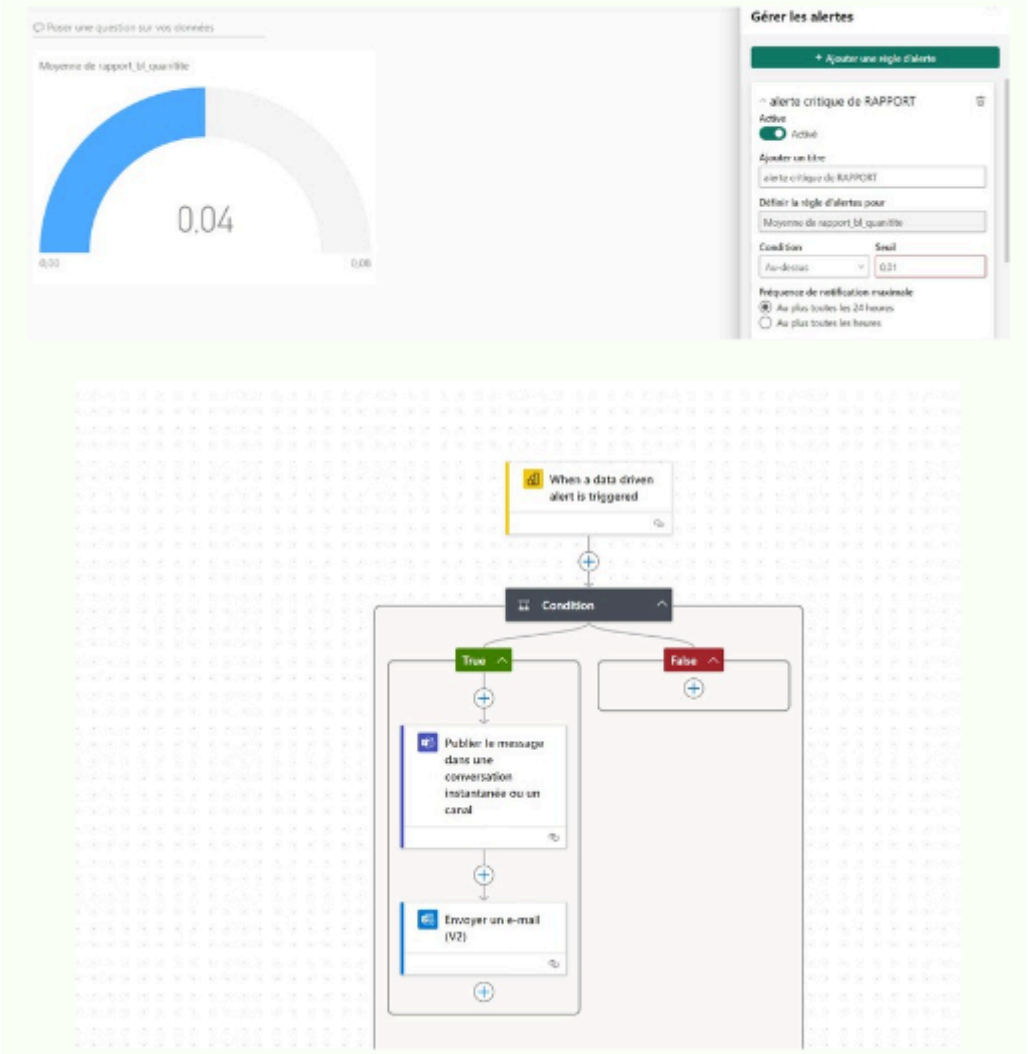
The "Filling Time" dashboard on Sboula provides comprehensive insights into the efficiency of filling processes crucial for operational optimization. Positioned conveniently on the left is a user-friendly pop-up menu, offering seamless access to various functions and filters, enhancing user navigation and interaction. This dashboard prominently displays key indicators such as maximum filling time, average filling time, and minimum filling time, enabling users to monitor performance against set benchmarks effectively. One of the prominent features is the histogram illustrating filling time per product, providing valuable insights into product-specific filling efficiency and identifying areas for improvement. Another histogram showcases the average filling time per transporter, facilitating the evaluation of transporter performance and potential optimization of distribution processes. Moreover, an alert mechanism positioned at the top left corner dynamically displays the names of transporters with significantly high filling times. This alert system, powered by DAX, ensures timely identification of bottlenecks and enables proactive intervention to address inefficiencies effectively. With customizable filters, users can refine their analysis based on specific parameters, ensuring actionable insights are readily available. The "Filling Time" dashboard on Sboula empowers users to optimize filling processes, improve operational efficiency, and enhance overall performance.

Row-Level Security Implementation in Power BI



Row-Level Security (RLS) is a crucial feature in Power BI that ensures secure and controlled access to data based on user roles. By utilizing RLS effectively, we have tailored access permissions to meet specific user needs while safeguarding sensitive information. In our implementation, the CEO role has been granted unrestricted access to all data across all dashboards. This allows for a comprehensive overview of organizational performance and facilitates strategic decision-making. Conversely, users with roles such as "Transporter," exemplified by Issam, have restricted access limited to data relevant to their responsibilities. This ensures that Issam, for instance, can only access data pertinent to transportation activities, safeguarding confidentiality and maintaining data integrity. By adopting RLS, we have established a robust data security framework that balances the need for accessibility with the imperative of data protection. This approach not only enhances operational efficiency by providing users with tailored insights but also mitigates the risk of unauthorized data access, thereby upholding compliance standards and fostering trust in our data management practices

Step 4 : Power Automate



In our project, we leveraged Power Automate to enhance our monitoring and alerting systems. Specifically, we utilized Power Automate to create automated alerts triggered when predefined thresholds were exceeded in our reports. By integrating Power Automate with our reporting tools, we were able to set up notifications that instantly alerted stakeholders when key metrics surpassed certain predefined limits. This proactive approach to monitoring allowed us to swiftly address issues and take corrective actions, ensuring timely responses to critical events and minimizing disruptions to our project workflow. Through the strategic

implementation of Power Automate, we effectively enhanced our reporting capabilities and maintained a proactive stance in managing project performance.

4. Machine Learning

4.1 STPA

- **Regression**

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import cross_val_score
```

```
X =filtered_data.drop(columns=['Rapport'])  # Features
y = filtered_data['Rapport']  # Target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=40)

model = LinearRegression()

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f"Root Mean Squared Error: {rmse}")
```

```
Root Mean Squared Error: 0.025909659870338657
```

```python
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import GradientBoostingRegressor

gb_regressor = GradientBoostingRegressor()

param_grid = {
    'n_estimators': [50, 100, 200],
    'learning_rate': [0.01, 0.1, 0.5],
    'max_depth': [3, 4, 5]
}

grid_search = GridSearchCV(gb_regressor, param_grid, cv=5, scoring='neg_mean_squared_error')

grid_search.fit(X_train, y_train)

best_params = grid_search.best_params_
best_model = grid_search.best_estimator_

y_pred = best_model.predict(X_test)

rmse = np.sqrt(mse)
print(f"Mean Squared Error: {mse}")

# Print the best hyperparameters
print("Best Hyperparameters:", best_params)
```

```
Mean Squared Error: 0.0006713104745966374
Best Hyperparameters: {'learning_rate': 0.5, 'max_depth': 3, 'n_estimators': 200}
```

We used two models, LinearRegression() and GradientBoostingRegressor(), to predict the value of a ratio for different given parameters, determining if it is greater than 1 or less than 1

- The LinearRegression model yielded a high Root Mean Squared Error (RMSE), indicating it is not a good model. Therefore, we sought a more effective model:
- GradientBoostingRegressor() which gave a mean squared error of 0.00067 with the optimal parameters {'learning_rate': 0.5, 'max_depth': 3, 'n_estimators': 200}, making it a good model.

- **Classification**

Classification

Type d'écart

```
[ ] filtered_data["Type_Ecart"].value_counts()
```

```
Type_Ecart
1    840
0    199
Name: count, dtype: int64
```

```
[ ] filtered_data.dtypes
```

```
Transporteurs                int64
Clients                      int64
Qt BL/qx_x                 float64
Ecart/Qx                   float64
Type_Ecart                   int64
Count_Type_Ecart           float64
NB Entree Transporteur     float64
Qt BL/NBP                  float64
Nombre_BL                  float64
Rapport                    float64
remplisssage_par_quintaux  float64
dtype: object
```

```
] filtered_data_updated = filtered_data.drop(["Count_Type_Ecart","Ecart/Qx"], axis = 1)
```

```
filtered_data_updated
```

	Transporteurs	Clients	Qt BL/qx_x	Type_Ecart	NB Entree Transporteur	Qt BL/NBP	Nombre_BL	Rapport	remplisssage_par_quintaux
117	11	2	50.0	1	1.0	300.0	3.0	0.010000	75.0
123	5	0	15.0	1	4.0	210.0	11.0	0.052381	50.0
144	3	0	10.0	1	4.0	125.0	7.0	0.056000	106.0
156	3	0	40.0	1	4.0	106.0	3.0	0.028302	121.0
164	0	0	25.0	1	3.0	175.0	8.0	0.045714	66.0
...	...	...	...	...	...	...	...	...	...
4422	10	2	50.0	1	1.0	300.0	1.0	0.003333	54.0
4424	14	3	120.0	1	1.0	120.0	1.0	0.008333	500.0
4425	0	3	150.0	1	2.0	150.0	1.0	0.006667	22.0
4426	2	1	100.0	1	2.0	250.0	1.0	0.004000	238.0
4429	2	1	200.0	0	2.0	200.0	1.0	0.005000	327.0

1039 rows × 9 columns

```
[ ] filtered_data_updated.columns

    Index(['Transporteurs', 'Clients', 'Qt BL/qx_x', 'Type_Ecart',
           'NB Entree Transporteur', 'Qt BL/NBP', 'Nombre_BL', 'Rapport',
           'remplisssage_par_quintaux'],
          dtype='object')

[ ] filtered_data_updated = filtered_data_updated[['Transporteurs', 'Clients', 'Qt BL/qx_x',
           'NB Entree Transporteur', 'Qt BL/NBP', 'Nombre_BL', 'Rapport',
           'remplisssage_par_quintaux','Type_Ecart']]

[ ] X = filtered_data_updated.drop('Type_Ecart', axis=1)
    y = filtered_data_updated['Type_Ecart']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

We used two classification models, GradientBoostingClassifier and RandomForestClassifier, to classify the type of deviation as positive or negative.

- The first model achieved an accuracy of 0.8413, indicating it is a good model.
- The second model, RandomForestClassifier, achieved an accuracy of 0.8317, which also indicates it is a good model.

To test the efficiency of these models for different probability thresholds, we used an ROC curve to model the performance for various thresholds. We found a good AUC (Area Under the Curve) of 0.85, which is close to 1, indicating that the model is performant. Based on the ROC curve, we optimized the model by identifying the best threshold.

```
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)
```

```
clf = RandomForestClassifier(n_estimators=100, random_state=40)
clf.fit(X_train_resampled, y_train_resampled)


y_pred = clf.predict(X_test)
```

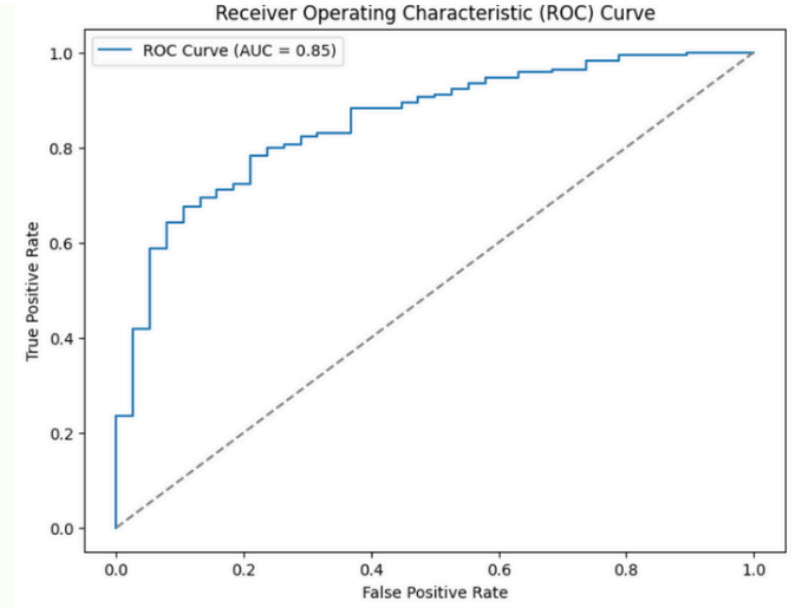
```
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 0.8413461538461539
```

```
clf = GradientBoostingClassifier(random_state=40)

param_grid = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.01, 0.1, 0.5],
    'max_depth': [3, 5, 7]
}

cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
grid_search = GridSearchCV(estimator=clf, param_grid=param_grid, scoring='accuracy', cv=cv, verbose=1, n_jobs=-1)

grid_search.fit(X_train_resampled, y_train_resampled)

best_clf = grid_search.best_estimator_
y_pred = best_clf.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")
```

```
Fitting 5 folds for each of 27 candidates, totalling 135 fits
Accuracy: 0.8317307692307693
```

```python
y_pred_proba = best_clf.predict_proba(X_test)[:, 1]

auc_score = roc_auc_score(y_test, y_pred_proba)
print(f"AUC: {auc_score}")

fpr, tpr, thresholds = roc_curve(y_test, y_pred_proba)

# Plot ROC curve
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, label=f'ROC Curve (AUC = {auc_score:.2f})')
plt.plot([0, 1], [0, 1], linestyle='--', color='gray')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend()
plt.show()
```



- **Filling time**

Remplissage

```
[ ] filtered_data.columns

    Index(['Transporteurs', 'Clients', 'Qt BL/qx_x', 'Ecart/Qx', 'Type_Ecart',
           'Count_Type_Ecart', 'NB Entree Transporteur', 'Qt BL/NBP', 'Nombre_BL',
           'Rapport', 'remplisssage_par_quintaux'],
          dtype='object')

[ ] filtered_data['remplissage_category'] = np.where(filtered_data['remplisssage_par_quintaux'] <= 180, 1,
                                            np.where(filtered_data['remplisssage_par_quintaux'] <= 300, 0, 2))
```

Double-cliquez (ou appuyez sur Entrée) pour modifier

```
[ ] filtered_data['remplissage_category'].value_counts()

    remplissage_category
    1    492
    2    369
    0    178
    Name: count, dtype: int64

[ ] X = filtered_data.drop(['remplissage_category','remplisssage_par_quintaux'], axis=1)
    y = filtered_data['remplissage_category']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
Epoch 1/10
21/21 [==============================] - 1s 14ms/step - loss: 44.9444 - accuracy: 0.4217 - val_loss: 7.5059 - val_accuracy: 0.6946
Epoch 2/10
21/21 [==============================] - 0s 4ms/step - loss: 18.3987 - accuracy: 0.5633 - val_loss: 3.5646 - val_accuracy: 0.7365
Epoch 3/10
21/21 [==============================] - 0s 4ms/step - loss: 10.4099 - accuracy: 0.6521 - val_loss: 3.6927 - val_accuracy: 0.7904
Epoch 4/10
21/21 [==============================] - 0s 4ms/step - loss: 8.6486 - accuracy: 0.6717 - val_loss: 3.1380 - val_accuracy: 0.8443
Epoch 5/10
21/21 [==============================] - 0s 3ms/step - loss: 7.8806 - accuracy: 0.6913 - val_loss: 3.5572 - val_accuracy: 0.8443
Epoch 6/10
21/21 [==============================] - 0s 3ms/step - loss: 7.0105 - accuracy: 0.7184 - val_loss: 3.6774 - val_accuracy: 0.8323
Epoch 7/10
21/21 [==============================] - 0s 3ms/step - loss: 5.3343 - accuracy: 0.7575 - val_loss: 3.0482 - val_accuracy: 0.8443
Epoch 8/10
21/21 [==============================] - 0s 3ms/step - loss: 5.3840 - accuracy: 0.7199 - val_loss: 2.6422 - val_accuracy: 0.8503
Epoch 9/10
21/21 [==============================] - 0s 4ms/step - loss: 4.2189 - accuracy: 0.7560 - val_loss: 2.3328 - val_accuracy: 0.8743
Epoch 10/10
21/21 [==============================] - 0s 4ms/step - loss: 4.0392 - accuracy: 0.7319 - val_loss: 2.4528 - val_accuracy: 0.8683
7/7 [==============================] - 0s 2ms/step
Accuracy: 0.8462
              precision    recall  f1-score   support

           0       0.75      0.57      0.65        37
           1       0.91      0.84      0.88       101
           2       0.80      1.00      0.89        70

    accuracy                           0.85       208
   macro avg       0.82      0.80      0.80       208
weighted avg       0.85      0.85      0.84       208
```

We used two models to predict the filling value:

- KNeighborsClassifier() which achieved an accuracy of 0.6346, indicating it is not a very performant model.
- A more advanced neural network model, Sequential, which achieved an accuracy of 0.8462, indicating it is a very performant model.

Here is an example of testing this model: We input a set of parameters, and it provided an optimal result, indicating that the filling time is between 3 and 5 minutes.

```
X_test.iloc[0]
```

```
Transporteurs                3.000000
Clients                      0.000000
Qt BL/qx_x                  60.000000
Ecart/Qx                     0.382200
Type_Ecart                   1.000000
Count_Type_Ecart           127.000000
NB Entree Transporteur       4.000000
Qt BL/NBP                  109.000000
Nombre_BL                    4.000000
Rapport                      0.036697
remplisssage_par_quintaux   39.000000
Name: 1407, dtype: float64
```

```
[ ] X_input = X_test.iloc[0].values.reshape(1, -1)
```

```
[ ]
    y_pred_prob_exemple = model.predict(X_input)
    y_pred_exemple = np.argmax(y_pred_prob_exemple, axis=-1)

    1/1 [==============================] - 0s 114ms/step
```

```
[ ] print(y_pred_exemple)

    [1]
```

- Sentiment Analysis

For sentiment analysis, we opted to use complaint data to determine if the client is satisfied or not. We used two models:

- MultinomialNB() which achieved an accuracy of 0.939, indicating it is a very performant model. We tested this model with the examples new_comments = ["Perfect!", "Delivery delay."], and it gave the results [1, -1], indicating that 1 means the client is satisfied and -1 means the client is not satisfied.

```
] if y_pred_exemple == 1:
      print("Optimal")
  elif y_pred_exemple == 0:
      print("Remplissage lent")
  elif y_pred_exemple == 2:
      print("Remplissage très lent avec Python")
  else:
      print("Valeur de ypred non reconnue")
```

```
Optimal
```

Sentiment Analysis

```
!pip install pandas transformers
```

```
Requirement already satisfied: pandas in /usr/local/lib/python3.10/dist-packages (2.0.3)
Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.40.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.10/dist-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2023.4)
Requirement already satisfied: tzdata>=2022.1 in /usr/local/lib/python3.10/dist-packages (from pandas) (2024.1)
Requirement already satisfied: numpy>=1.21.0 in /usr/local/lib/python3.10/dist-packages (from pandas) (1.25.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.13.4)
Requirement already satisfied: huggingface-hub<1.0,>=0.19.3 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.20.3)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (24.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.1)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2023.12.25)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.31.0)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.19.1)
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.4.3)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.2)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.19.3->transformers) (2023.6.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.19.3->transformers) (4.11.0)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.10/dist-packages (from python-dateutil>=2.8.2->pandas) (1.16.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2024.2.2)
```

```
[ ] data_reclamation= pd.read_excel(r"/content/reclamation2.xlsx")
```

```
data_reclamation= pd.read_excel(r"/content/reclamation2.xlsx")
```

```
data_reclamation
```

	Date de réclamation	Produit	Client	Commentaire	Sentiment
0	2024-01-04	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	-1
1	2024-02-13	PS 50KG	BOULANGERIE	Trace de son	-1
2	2024-01-11	PS 50KG	BOULANGERIE	Quantité manquante	-1
3	2024-01-12	PS-7 20KG	BOULANGERIE	Retard de livraison	-1
4	2024-01-13	SSSE 20KG	BOULANGERIE	Taux de piqûre élevé	-1
...	...	...	...	...	...
158	2024-08-24	SSSE 20KG	BOULANGERIE	Emballage non conforme	-1
159	2024-08-25	PS-7 20KG	BOULANGERIE	Taux de piqûre élevé	-1
160	2024-08-26	PS-7 20KG	BOULANGERIE	Produit non conforme	-1
161	2024-08-27	PS 50KG	BOULANGERIE	Taux de piqûre élevé	-1
162	2024-08-28	PS-7 20KG	BOULANGERIE	Retard de livraison	-1

163 rows × 5 columns

```
data_reclamation = data_reclamation.sample(frac=1, random_state=42).reset_index(drop=True)
```

```
data_reclamation
```

	Date de réclamation	Produit	Client	Commentaire	Sentiment
0	2024-08-01	PS-7 20KG	BOULANGERIE	Le colis était correctement emballé	0
1	2024-07-12	PS-7 20KG	BOULANGERIE	Le colis était correctement emballé	0
2	2024-07-28	SON 50 KG	GROSSISTE	Le transporteur a remis le colis comme prévu	0
3	2024-05-13	SSSE 20KG	BOULANGERIE	Livraison rapide et sans souci	1
4	2024-06-22	SSSE 20KG	BOULANGERIE	Le colis était correctement emballé	0
...	...	...	...	...	...
158	2024-05-29	PS-7 20KG	BOULANGERIE	Livraison impeccable, merci	1
159	2024-07-03	PS 50KG	BOULANGERIE	La livraison était standard, sans problème par...	0
160	2024-02-10	PS 50KG	OFFICE	Taux de piqûre élevé	-1
161	2024-06-19	SSSE 20KG	BOULANGERIE	La livraison était ordinaire, sans surprise	0
162	2024-06-29	PS 50KG	OFFICE	La livraison était ordinaire, sans surprise	0

163 rows × 5 columns

```
vectorizer = TfidfVectorizer(max_features=1000)
X = vectorizer.fit_transform(data_reclamation['Commentaire'])
```

```
X_train, X_test, y_train, y_test = train_test_split(X, data_reclamation['Sentiment'], test_size=0.2, random_state=40)
```

```
from sklearn.naive_bayes import MultinomialNB

model = MultinomialNB()
model.fit(X_train, y_train)
```

```
▾ MultinomialNB
MultinomialNB()
```

```
accuracy = model.score(X_test, y_test)
print(f"Accuracy: {accuracy}")
```

```
Accuracy: 0.9393939393939394
```

```
new_comments = ["Parfait!", "Retard de livraison."]
new_comments_transformed = vectorizer.transform(new_comments)
predictions = model.predict(new_comments_transformed)
print(predictions)
```

```
[ 1 -1]
```

Transformer : Bert

```
[ ] label_map = {-1: 0, 0: 1, 1: 2}
    data_reclamation['encoded_sentiment'] = data_reclamation['Sentiment'].map(label_map)

    # Verify label mapping
    print(data_reclamation['encoded_sentiment'].value_counts())
```

```
encoded_sentiment
0    64
1    63
2    36
Name: count, dtype: int64
```

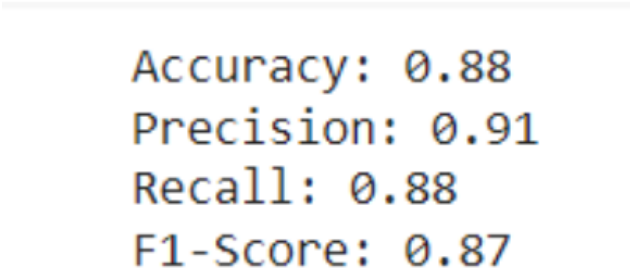
```
[ ] import torch
    from transformers import BertTokenizer, BertForSequenceClassification, AdamW, get_scheduler
    from sklearn.model_selection import train_test_split
    from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
    from torch.utils.data import DataLoader, TensorDataset

    model_name = "bert-base-uncased"
    tokenizer = BertTokenizer.from_pretrained(model_name)
    model = BertForSequenceClassification.from_pretrained(model_name, num_labels=3)
```

```
encoded_data = tokenizer(list(data_reclamation['Commentaire']), padding=True, truncation=True, max_length=128, return_tensors='pt')
labels = torch.tensor(data_reclamation['encoded_sentiment'].values)
```

```
train_inputs, val_inputs, train_labels, val_labels = train_test_split(encoded_data.input_ids, labels, test_size=0.2, random_state=42)
train_masks, val_masks = train_test_split(encoded_data.attention_mask, test_size=0.2, random_state=42)

train_dataset = TensorDataset(train_inputs, train_masks, train_labels)
val_dataset = TensorDataset(val_inputs, val_masks, val_labels)

batch_size = 16
train_loader = DataLoader(train_dataset, batch_size=batch_size, shuffle=True)
val_loader = DataLoader(val_dataset, batch_size=batch_size)
```

```
num_epochs = 3
learning_rate = 2e-5

optimizer = AdamW(model.parameters(), lr=learning_rate)
scheduler = get_scheduler("linear", optimizer, num_warmup_steps=0, num_training_steps=num_epochs * len(train_loader))
```

```
[ ] model.train()
    for epoch in range(num_epochs):
        for batch in train_loader:
            inputs, masks, labels = batch
            optimizer.zero_grad()
            outputs = model(inputs, attention_mask=masks, labels=labels)
            loss = outputs.loss
            loss.backward()
            optimizer.step()
            scheduler.step()

    model.eval()
    predictions = []
    true_labels = []
    with torch.no_grad():
        for batch in val_loader:
            inputs, masks, labels = batch
            outputs = model(inputs, attention_mask=masks)
            logits = outputs.logits
            batch_predictions = torch.argmax(logits, dim=1)
            predictions.extend(batch_predictions.cpu().numpy())
            true_labels.extend(labels.cpu().numpy())

    accuracy = accuracy_score(true_labels, predictions)
    precision = precision_score(true_labels, predictions, average='weighted')
    recall = recall_score(true_labels, predictions, average='weighted')
    f1 = f1_score(true_labels, predictions, average='weighted')

    print(f"Accuracy: {accuracy:.2f}")
    print(f"Precision: {precision:.2f}")
    print(f"Recall: {recall:.2f}")
```



- The advanced BERT model, which gave the following results:

Accuracy: 0.88 Precision: 0.91 Recall: 0.88 F1-Score: 0.87

This indicates that it is a good model. We tested this model with the examples new_comments = ["Product not as described", "The delivery time was reasonable"], and it

predicted tensor([0, 1]), meaning 0 for the client being neither surprised nor dissatisfied, and 1 for the client being satisfied.

4.2 ALCO

```
import psycopg2
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
```

```
# Établir la connexion à la base de données
conn = psycopg2.connect(
    dbname="DWH",
    user="postgres",
    password="sayda",
    host="localhost",
    port="5432"
)
```

```
# Créer un curseur pour exécuter des requêtes SQL
cur = conn.cursor()
```

```
# Définir la requête SQL
sql = """
SELECT date_pk, group_pk, product_pk, mp_pk, "Humidite", "Aw", "Proteine", "Amidon", "Fibre", "Calcium", price
FROM public."Fait";
"""
```

We have established the connection with the database and performed this query to retrieve the 'fait' table.

```
# Lire les données dans un DataFrame Pandas
df = pd.read_sql_query(sql, conn)

# Convertir le DataFrame en une chaîne de caractères formatée comme un tableau
table_string = df.to_string(index=False)

# Afficher la chaîne de caractères
print(table_string)
```

```
C:\Users\user\anaconda3\lib\site-packages\pandas\io\sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engin
e/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQ
LAlchemy
  warnings.warn(
```

date_pk	group_pk	product_pk	mp_pk	Humidite	Aw	Proteine	Amidon	Fibre	Calcium	price
1	997	1	0	11,76	0,651	19,4	35,38	3,83	None	0
1	997	2	0	11,59	0,624	14,22	37,83	5,47	None	0
1	997	3	0	12,23	0,635	16,33	34,28	5,06	None	0
1	0	0	336	12,37	None	47,1	None	3,45	None	5
1	0	0	337	12,5	None	14,48	33,28	7,16	None	8000
1	0	0	338	13,04	None	7,21	66,38	1,95	None	12000
1	0	0	339	13,82	None	11,44	61,27	1,33	None	6450
1	0	0	336	12,38	None	46,48	None	3,56	None	5
1	0	0	336	12,29	None	46,66	None	3,9	None	5
10	1006	1	0	11,26	0,612	19,27	35,01	4	None	0
10	1006	1	0	11,23	0,615	19,05	36,63	3,86	None	0
10	1006	2	0	11,13	0,594	14,28	38,57	5,18	None	0
10	1006	3	0	11,7	0,61	16,7	33,41	5,31	None	0
10	1006	14	0	11,23	0,592	16,22	39,69	4,19	None	0

We have retrieved the data from the "Fait" table.

```python
# Regrouper par 'name' et calculer les moyennes
df_grouped = df.groupby('name').mean().reset_index()

# Sélectionner uniquement les colonnes 'name', 'Proteine', 'Amidon', et 'Calcium'
df_result = df_grouped[['name', 'Proteine', 'Amidon', 'Calcium']]

# Afficher le DataFrame résultant avec les noms des produits et leurs moyennes
print(df_result)
```

	name	Proteine	Amidon	Calcium
0	ALCO 5	16.395556	39.692222	0.967778
1	ALCO 5 F	15.647778	40.290000	0.955556
2	ALCO 6	15.714286	35.232857	0.918571
3	ALCO 6 A	17.278000	33.408000	0.992000
4	ALCO 6 BL	19.710000	30.014000	1.122000
5	ALCO 6 SB	11.210000	47.432500	0.665000
6	ALCO 7	19.546500	36.003000	0.960500
7	ALCO 7 HP	19.376667	35.940000	1.086667
8	ALCO 7 Mash	25.830000	27.765000	1.540000
9	ALCO 7 Primo	18.704000	36.757000	0.943000
10	ALCO 7 T	16.120000	35.960000	0.845000
11	CGV 1 E	22.581739	37.371304	0.795217
12	CGV 2 G	19.544737	42.631053	0.738421
13	CGV 2 V	21.797143	41.274286	0.761429
14	PGV 1	20.495000	38.630000	1.850000
15	PGV 2	18.697500	41.505000	1.945000
16	PGV 4 E	16.125000	39.312500	4.572500
17	PGV 4 ES	17.684000	37.312000	4.408000
18	Sauvegarde	12.717500	42.262500	0.755000

In this figure, we have performed a query between the "Fait" table and the "Product" table to display the products without duplication, along with their component values such as protein, starch, and calcium.

```
Produits pour le cluster 0 de protéine :
0       ALCO 5
1     ALCO 5 F
2       ALCO 6
5    ALCO 6 SB
10     ALCO 7 T
16      PGV 4 E
18   Sauvegarde
Name: name, dtype: object

Produits pour le cluster 1 de protéine :
3       ALCO 6 A
4      ALCO 6 BL
6        ALCO 7
7      ALCO 7 HP
9    ALCO 7 Primo
12       CGV 2 G
14         PGV 1
15         PGV 2
17      PGV 4 ES
Name: name, dtype: object

Produits pour le cluster 2 de protéine :
8    ALCO 7 Mash
11       CGV 1 E
13       CGV 2 V
Name: name, dtype: object
```

```
Produits pour le cluster 0 d'amidon :
0       ALCO 5
1     ALCO 5 F
5    ALCO 6 SB
12       CGV 2 G
13       CGV 2 V
15         PGV 2
16      PGV 4 E
18   Sauvegarde
Name: name, dtype: object

Produits pour le cluster 1 d'amidon :
2         ALCO 6
3       ALCO 6 A
6         ALCO 7
7      ALCO 7 HP
9    ALCO 7 Primo
10     ALCO 7 T
11       CGV 1 E
14         PGV 1
17      PGV 4 ES
Name: name, dtype: object

Produits pour le cluster 2 d'amidon :
4      ALCO 6 BL
8    ALCO 7 Mash
Name: name, dtype: object
```

```
Produits pour le cluster 0 de calcium :
16     PGV 4 E
17    PGV 4 ES
Name: name, dtype: object

Produits pour le cluster 1 de calcium :
0       ALCO 5
1     ALCO 5 F
2       ALCO 6
3       ALCO 6 A
4      ALCO 6 BL
5      ALCO 6 SB
6        ALCO 7
7      ALCO 7 HP
9    ALCO 7 Primo
10     ALCO 7 T
11       CGV 1 E
12       CGV 2 G
13       CGV 2 V
18   Sauvegarde
Name: name, dtype: object

Produits pour le cluster 2 de calcium :
8    ALCO 7 Mash
14         PGV 1
15         PGV 2
Name: name, dtype: object
```

We have implemented a simple algorithm (KMeans model) and applied it to each component to create clusters for each component. The clusters are divided based on high, medium, and low protein values, as well as for starch and calcium.

```
# Sélectionner uniquement les colonnes 'Proteine', 'Amidon', et 'Calcium' pour l'analyse
X = df_grouped[['Proteine', 'Amidon', 'Calcium']]

# Normaliser les données
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Calculer la matrice de similarité avec la distance euclidienne
distances = pairwise_distances(X_scaled, metric='euclidean')

# Appliquer Spectral Clustering
n_clusters = 3  # Nombre de clusters
spectral = SpectralClustering(n_clusters=n_clusters, affinity='precomputed', random_state=42)
cluster_labels = spectral.fit_predict(distances)

# Ajouter les labels de cluster au DataFrame
df_grouped['Cluster_Spectral'] = cluster_labels

# Affichage des produits associés à chaque cluster pour chaque composant
for cluster in range(n_clusters):
    cluster_data = df_grouped[df_grouped['Cluster_Spectral'] == cluster]
    print(f"Produits pour le cluster {cluster} :")
    print(cluster_data['name'])
    print()
# Affichage du DataFrame avec les clusters
print(df_grouped[['name', 'Proteine', 'Amidon', 'Calcium', 'Cluster_Spectral']])
# Visualisation
fig, axes = plt.subplots(3, 1, figsize=(10, 15))

# Liste des colonnes à utiliser pour la visualisation
features = ['Proteine', 'Amidon', 'Calcium']
```

```
Produits pour le cluster 0 :
0          ALCO 5
1        ALCO 5 F
2          ALCO 6
3        ALCO 6 A
4       ALCO 6 BL
5       ALCO 6 SB
6          ALCO 7
8     ALCO 7 Mash
9    ALCO 7 Primo
11        CGV 1 E
12        CGV 2 G
13        CGV 2 V
14          PGV 1
15          PGV 2
16        PGV 4 E
17       PGV 4 ES
18     Sauvegarde
Name: name, dtype: object
```

```
Produits pour le cluster 1 :
10    ALCO 7 T
Name: name, dtype: object

Produits pour le cluster 2 :
7    ALCO 7 HP
Name: name, dtype: object

            name   Proteine     Amidon   Calcium  Cluster_Spectral
0         ALCO 5  16.395556  39.692222  0.967778                 0
1       ALCO 5 F  15.647778  40.290000  0.955556                 0
2         ALCO 6  15.714286  35.232857  0.918571                 0
3       ALCO 6 A  17.278000  33.408000  0.992000                 0
4      ALCO 6 BL  19.710000  30.014000  1.122000                 0
5      ALCO 6 SB  11.210000  47.432500  0.665000                 0
6         ALCO 7  19.546500  36.003000  0.960500                 0
7      ALCO 7 HP  19.376667  35.940000  1.086667                 2
8    ALCO 7 Mash  25.830000  27.765000  1.540000                 0
9   ALCO 7 Primo  18.704000  36.757000  0.943000                 0
10     ALCO 7 T  16.120000  35.960000  0.845000                 1
```

Based on the information provided, it seems that you have applied an advanced algorithm (spectral clustering) to the data. The algorithm has created three clusters. In cluster 0, the levels of the components vary. In cluster 1, the protein levels are lower compared to the other clusters. Lastly, in cluster 2, the levels of the components are higher.

```python
from sklearn.metrics import silhouette_score

# Calcul de l'indice de silhouette pour KMeans
silhouette_kmeans = silhouette_score(X_scaled, labels_proteine)
print("Silhouette Score pour KMeans:", silhouette_kmeans)

# Calcul de l'indice de silhouette pour Spectral Clustering
silhouette_spectral = silhouette_score(X_scaled, cluster_labels)
print("Silhouette Score pour Spectral Clustering:", silhouette_spectral)
```

```
Silhouette Score pour KMeans: 0.10044495058764027
Silhouette Score pour Spectral Clustering: -0.3895528218902047
```

We have compared the two algorithms, KMeans and Spectral Clustering, using the evaluation metric KMeans. The KMeans algorithm has a score of 0.10044, while the Spectral Clustering algorithm has a score of -0.389. Since the score for Spectral Clustering is negative, it suggests that the algorithm is poorly defined. Therefore, KMeans is considered better than Spectral Clustering.

```python
# Requête SQL pour charger les données de la table Fait
sql_mp = """
SELECT F.mp_pk, M.name AS nom_MP, F."Proteine", F."Amidon", F."Fibre"
FROM public."Fait" F
JOIN public."Dim_MP" M ON F.mp_pk = M.mp_pk
WHERE F."Proteine" IS NOT NULL AND F."Amidon" IS NOT NULL AND F."Fibre" IS NOT NULL;
"""
```

```python
    # Lire les données dans un DataFrame Pandas
    df_mp = pd.read_sql_query(sql_mp, conn)

    # Afficher les premières lignes pour vérification
    print(df_mp.head())
    # Remplacer les virgules par des points dans les colonnes 'Proteine', 'Amidon' et 'Fibre'
    df_mp['Proteine'] = df_mp['Proteine'].str.replace(',', '.').astype(float)
    df_mp['Amidon'] = df_mp['Amidon'].str.replace(',', '.').astype(float)
    df_mp['Fibre'] = df_mp['Fibre'].str.replace(',', '.').astype(float)
```

```
   mp_pk      nom_mp Proteine Amidon Fibre
0    337  Son de blé    14,48  33,28  7,16
1    338      Mais b     7,21  66,38  1,95
2    339   Remoulage    11,44  61,27  1,33
3    337  Son de blé    14,06  36,95  6,51
4    338      Mais b     7,38  66,56  1,84
```

In this figure, we have performed a query between the "Fait" table and the "MP" table to retrieve only the names of the raw materials along with their component values such as protein, starch, and fiber.

Algorithme Simple : Classification

```
# Sélectionner la ligne avec la plus grande valeur de Protéine
meilleure_mp_proteine = df_mp.loc[df_mp['Proteine'].idxmax()]

# Afficher la meilleure matière première en termes de protéine
print("La meilleure matière première en termes de protéine est :")
print(meilleure_mp_proteine)
#######
# Sélectionner la ligne avec la plus grande valeur d'Amidon
meilleure_mp_amidon = df_mp.loc[df_mp['Amidon'].idxmax()]

# Afficher la meilleure matière première en termes d'Amidon
print("La meilleure matière première en termes d'Amidon est :")
print(meilleure_mp_amidon)
#########
# Sélectionner la ligne avec la plus grande valeur de Fibre
meilleure_mp_fibre = df_mp.loc[df_mp['Fibre'].idxmax()]

# Afficher la meilleure matière première en termes de Fibre
print("La meilleure matière première en termes de Fibre est :")
print(meilleure_mp_fibre)
```

```
La meilleure matière première en termes de protéine est :
mp_pk            359
nom_mp       Son blé
Proteine        16.2
Amidon         32.42
Fibre           6.59
Name: 14, dtype: object
La meilleure matière première en termes d'Amidon est :
mp_pk            338
nom_mp       Mais b
Proteine         9.5
Amidon         67.75
Fibre           1.96
Name: 33, dtype: object
La meilleure matière première en termes de Fibre est :
mp_pk            359
nom_mp       Son blé
Proteine       15.77
Amidon         28.01
Fibre           9.24
Name: 39, dtype: object
```

We applied a simple classification algorithm to identify the best raw materials in terms of protein, starch, and fiber. This figure indicates that wheat is the best raw material in terms of protein, but is the best raw material in terms of starch, and wheat is the best raw material in terms of fiber.

Algorithme Avancé : AffinityPropagation

```
from sklearn.cluster import AffinityPropagation
import pandas as pd

# Création d'un DataFrame exemple pour illustrer le code
data = {
    'nom_mp': ['Son de blé', 'Mais b', 'Remoulage', 'Son de blé', 'Mais b', 'Son de blé'],
    'Proteine': [14.48, 7.21, 11.44, 14.06, 7.38, 15.11],
    'Amidon': [33.28, 66.38, 61.27, 36.95, 66.56, 35.85],
    'Fibre': [7.16, 1.95, 1.33, 6.51, 1.84, 6.72]
}
df_mp = pd.DataFrame(data)

# Sélection des caractéristiques pour l'Affinity Propagation
X_affinity = df_mp[['Proteine', 'Amidon', 'Fibre']]

# Création du modèle Affinity Propagation
aff_prop = AffinityPropagation(random_state=0)
clusters_affinity = aff_prop.fit_predict(X_affinity)

# Ajouter les étiquettes de cluster au DataFrame
df_mp['Cluster_Affinity'] = clusters_affinity

# Afficher les matières premières et leur cluster assigné par Affinity Propagation
print("Matières premières et leur cluster assigné par Affinity Propagation :")
print(df_mp[['nom_mp', 'Proteine', 'Amidon', 'Fibre', 'Cluster_Affinity']])
```

```
# Afficher les matières premières et leur cluster assigné par Affinity Propagation
print("Matières premières et leur cluster assigné par Affinity Propagation :")
print(df_mp[['nom_mp', 'Proteine', 'Amidon', 'Fibre', 'Cluster_Affinity']])

# Trouver les matières premières les plus représentatives de chaque cluster Affinity Propagation
cluster_centers_indices = aff_prop.cluster_centers_indices_
mean_values_affinity = df_mp.iloc[cluster_centers_indices].drop(columns=['Cluster_Affinity'])

if len(cluster_centers_indices) > 0:
    # Identifier la meilleure matière première en termes de Protéine
    meilleure_prot_exemplar = mean_values_affinity.sort_values(by='Proteine', ascending=False).iloc[0]
    print("\nLa meilleure matière première en termes de Protéine est :")
    print(meilleure_prot_exemplar)

    # Identifier la meilleure matière première en termes d'Amidon
    meilleure_amidon_exemplar = mean_values_affinity.sort_values(by='Amidon', ascending=False).iloc[0]
    print("\nLa meilleure matière première en termes d'Amidon est :")
    print(meilleure_amidon_exemplar)

    # Identifier la meilleure matière première en termes de Fibre
    meilleure_fibre_exemplar = mean_values_affinity.sort_values(by='Fibre', ascending=False).iloc[0]
    print("\nLa meilleure matière première en termes de Fibre est :")
    print(meilleure_fibre_exemplar)
else:
    print("Aucun centre de cluster n'a été identifié par Affinity Propagation.")
```

```
Matières premières et leur cluster assigné par Affinity Propagation :
        nom_mp  Proteine  Amidon  Fibre  Cluster_Affinity
0  Son de blé      14.48   33.28   7.16                 1
1       Mais b      7.21   66.38   1.95                 0
2   Remoulage      11.44   61.27   1.33                 0
3  Son de blé      14.06   36.95   6.51                 1
4       Mais b      7.38   66.56   1.84                 0
5  Son de blé      15.11   35.85   6.72                 1

La meilleure matière première en termes de Protéine est :
nom_mp       Son de blé
Proteine          15.11
Amidon            35.85
Fibre              6.72
Name: 5, dtype: object

La meilleure matière première en termes d'Amidon est :
nom_mp       Mais b
Proteine       7.21
Amidon        66.38
Fibre          1.95
Name: 1, dtype: object

La meilleure matière première en termes de Fibre est :
nom_mp       Son de blé
Proteine          15.11
Amidon            35.85
Fibre              6.72
Name: 5, dtype: object
```

When applying the advanced algorithm, Affinity Propagation, it yielded the same results as the classification algorithm but with the addition of clusters. In other words, Affinity Propagation grouped the raw materials into clusters based on their protein, starch, and fiber content. The clusters likely represent different groups of raw materials with similar nutritional profiles.

When evaluating the performance of the two algorithms, we observed that Affinity Propagation outperforms classification as it approaches a value close to 1. This suggests that Affinity Propagation is a more effective algorithm than classification in the given context.

```
Silhouette Score pour KMeans: 0.7487114101041327
Silhouette Score pour Affinity Propagation: 0.8789545397772517
```

When evaluating the performance of the two algorithms, we observed that Affinity Propagation outperforms classification as it approaches a value close to 1. This suggests that Affinity Propagation is a more effective algorithm than classification in the given context.

5. *Flask*

app.py × style.css <> index.html

```python
import numpy as np
from flask import Flask, request, jsonify, render_template
import pickle

# Create flask app
flask_app = Flask(__name__)
model = pickle.load(open("model.pkl", "rb"))

@flask_app.route("/")
def Home():
    return render_template("index.html")

1 usage (1 dynamic)
@flask_app.route( rule: "/predict", methods=["POST"])
def predict():
    # Retrieve values from the form
    transporteurs = float(request.form["transporteurs"])
    clients = float(request.form["clients"])
    qt_bl = float(request.form["qt_bl"])
    ecart_qx = float(request.form["ecart_qx"])
    type_ecart = float(request.form["type_ecart"])
    count_type_ecart = float(request.form["count_type_ecart"])
    nb_entree_transporteur = float(request.form["nb_entree_transporteur"])
    qt_bl_nbp = float(request.form["qt_bl_nbp"])
    nombre_bl = float(request.form["nombre_bl"])
    remplisssage_par_quintaux = float(request.form["remplisssage_par_quintaux"])

    features = [transporteurs,clients,qt_bl,ecart_qx,type_ecart,count_type_ecart,nb_entree_transporteur,
                qt_bl_nbp,nombre_bl,remplisssage_par_quintaux]
    # Convert to numpy array
```

```python
@flask_app.route( rule: "/predict", methods=["POST"])
def predict():
    # Retrieve values from the form
    transporteurs = float(request.form["transporteurs"])
    clients = float(request.form["clients"])
    qt_bl = float(request.form["qt_bl"])
    ecart_qx = float(request.form["ecart_qx"])
    type_ecart = float(request.form["type_ecart"])
    count_type_ecart = float(request.form["count_type_ecart"])
    nb_entree_transporteur = float(request.form["nb_entree_transporteur"])
    qt_bl_nbp = float(request.form["qt_bl_nbp"])
    nombre_bl = float(request.form["nombre_bl"])
    remplisssage_par_quintaux = float(request.form["remplisssage_par_quintaux"])

    features = [transporteurs,clients,qt_bl,ecart_qx,type_ecart,count_type_ecart,nb_entree_transporteur,
                qt_bl_nbp,nombre_bl,remplisssage_par_quintaux]
    # Convert to numpy array
    features_array = np.array(features).reshape(1, -1)

    # Make prediction
    prediction = model.predict(features_array)

    # Render the result on the index.html template
    return render_template( template_name_or_list: "index.html", prediction_text=f"Le rapport prédit est: {prediction[0]}")
if __name__ == "__main__":
    flask_app.run(debug=True)
```

```css
/* styles.css */

body {
    font-family: Arial, sans-serif;
    background-color: #f0f0f0;
    margin: 0;
    padding: 0;
}

.container {
    max-width: 600px;
    margin: 20px auto;
    padding: 20px;
    background-color: #fff;
    border-radius: 5px;
    box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
}

form {
    margin-bottom: 20px;
}

label {
    display: block;
    margin-bottom: 5px;
}

input[type="text"] {
    width: 100%;
```

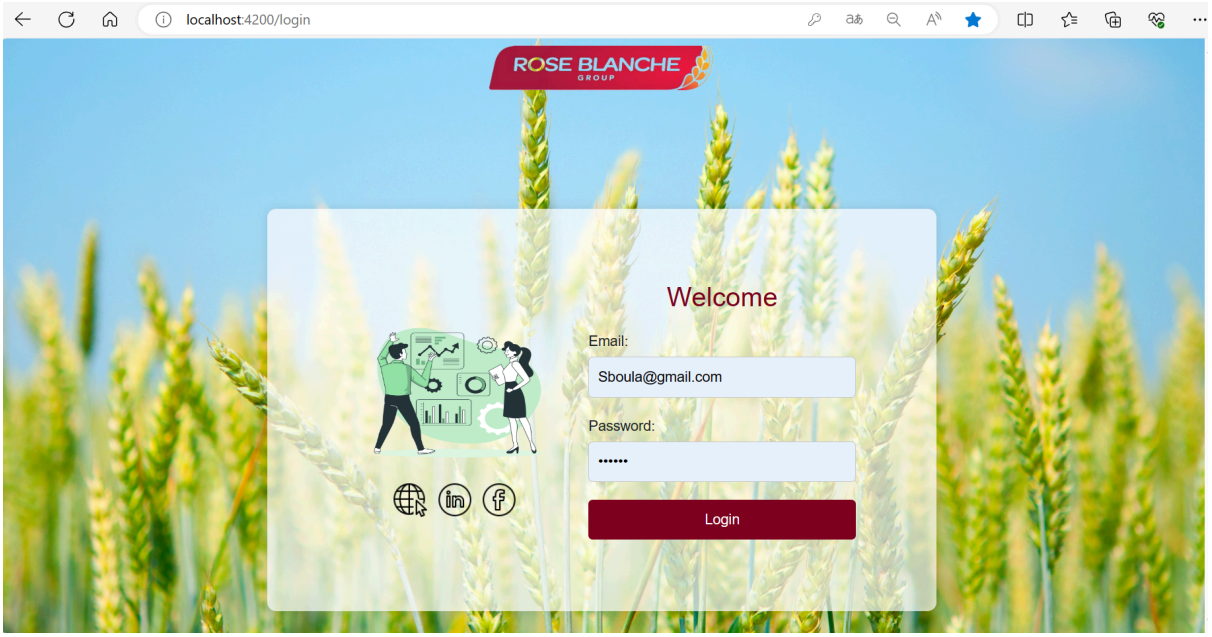
app.py style.css <> index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            margin: 0;
            padding: 0;
        }

        .container {
            max-width: 600px;
            margin: 20px auto;
            padding: 20px;
            background-color: #fff;
            border-radius: 5px;
            box-shadow: 0px 0px 10px rgba(0, 0, 0, 0.1);
        }

        form {
            margin-bottom: 20px;
        }

        label {
            display: block;
            margin-bottom: 5px;
```

We used Flask to retrieve values and predict the report, subsequently displaying it on Angular. Flask served as the backend framework, handling the retrieval of input data and invoking the predictive model to generate the report. Upon prediction, Flask seamlessly communicated the results to Angular, which served as the frontend framework responsible for presenting the report to users in an intuitive and interactive manner. This integration of Flask and Angular facilitated a smooth and efficient flow of data, ensuring that users could access accurate and timely reports with ease.

6. *Angular*

This iframe embeds a Power BI report directly into our application, allowing users to seamlessly access and interact with dynamic visualizations and insights without leaving the Angular environment. By leveraging this approach, we provide users with a cohesive experience, combining the power of Power BI with the familiarity and functionality of our Angular application interface

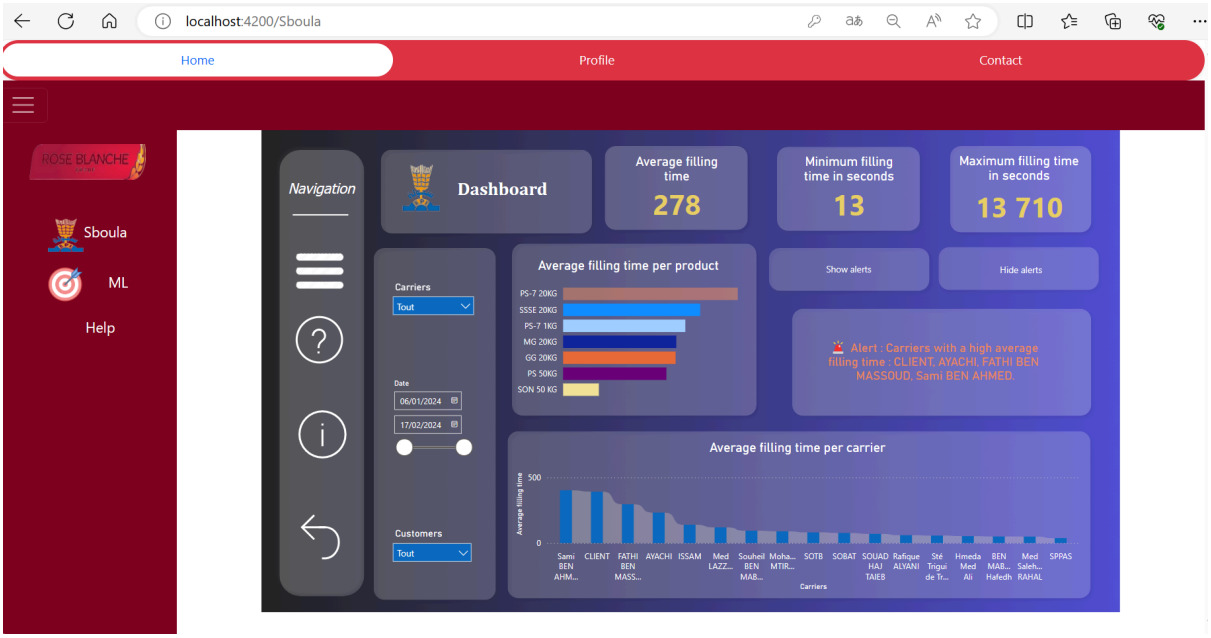


The screenshot above showcases the login page, serving as the gateway to our application. Here, users are prompted to input their credentials for authentication before gaining access to the main dashboard. This robust login interface ensures the security of sensitive data and restricts entry to authorized users only.

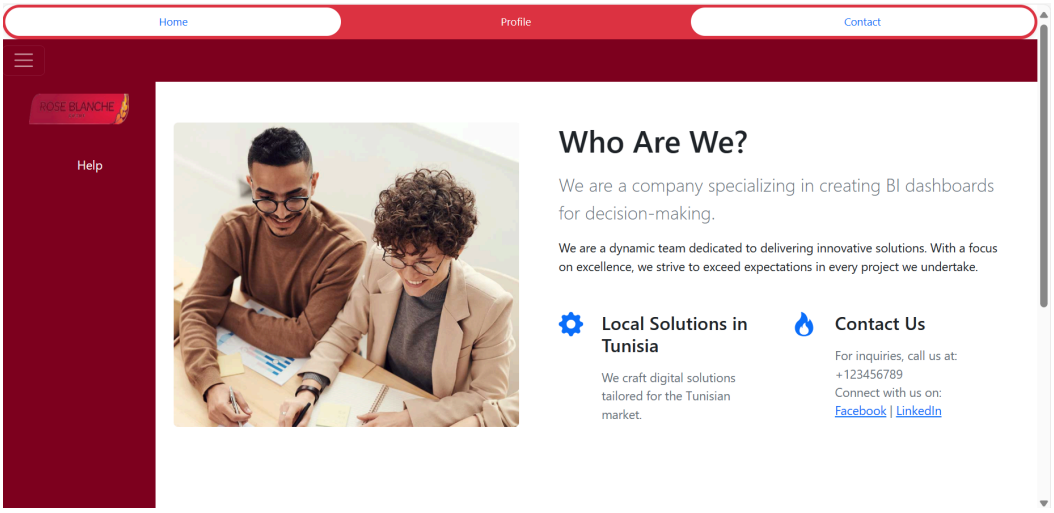
Upon successful authentication, users are seamlessly directed to the main dashboard, where they can leverage the advanced capabilities of Power BI integrated into our Angular application. This intuitive dashboard empowers users to visualize, explore, and analyze data interactively, facilitating informed decision-making and enhancing overall user experience.

Furthermore, our login system offers users the flexibility to access specific sections of the application based on their role or permissions. By entering their email and corresponding password, users can choose to view either the ALCO or Sboula sections, tailoring their experience to their specific needs and preferences.

This comprehensive login mechanism not only ensures the security of our application but also enhances usability by providing a personalized and intuitive user experience.



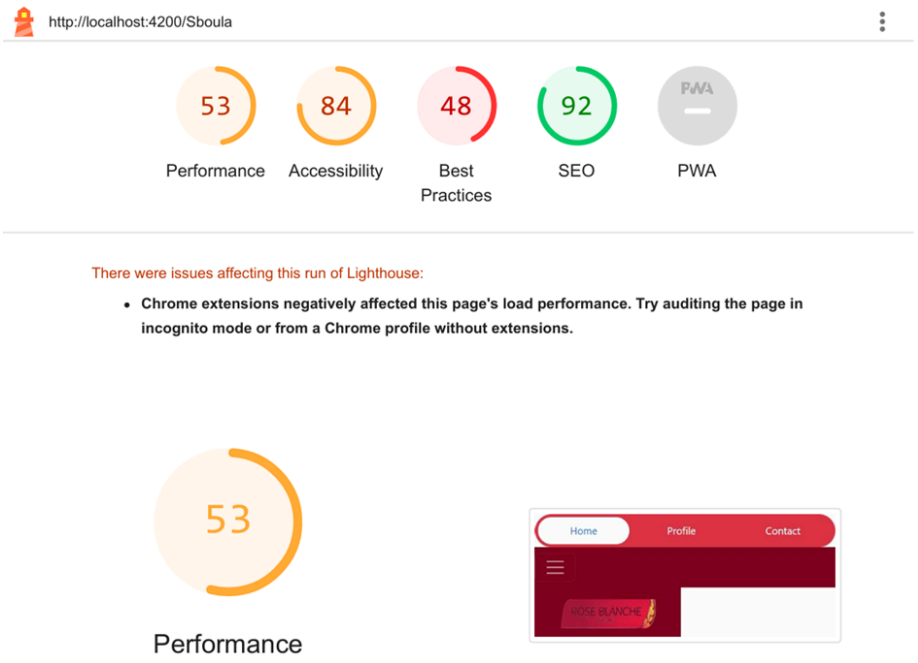
One of the key components of our project involved integrating advanced data visualization capabilities to enable in-depth analysis of collected information. To meet this requirement, we deployed Power BI as a business intelligence tool within our Angular application. The integration of Power BI into the application was accomplished using available Angular APIs and components, ensuring a seamless and cohesive integration process. Special attention was paid to the user interface to ensure a consistent and intuitive user experience.



7. Audit

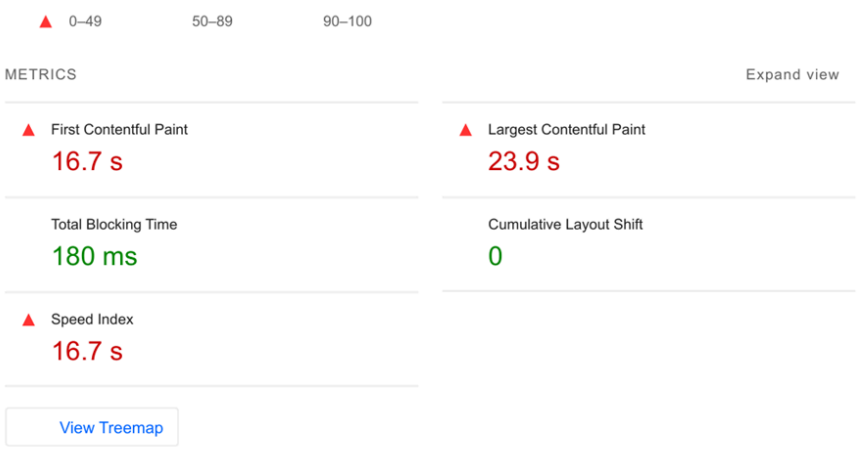
We carry out a website audit to identify the strengths and weaknesses of your page. It's like a health exam for your website! This improves its performance, its natural referencing (SEO) and its accessibility. Basically, an audit helps your website be faster, easier to find, and more enjoyable to use.

- **STPA**

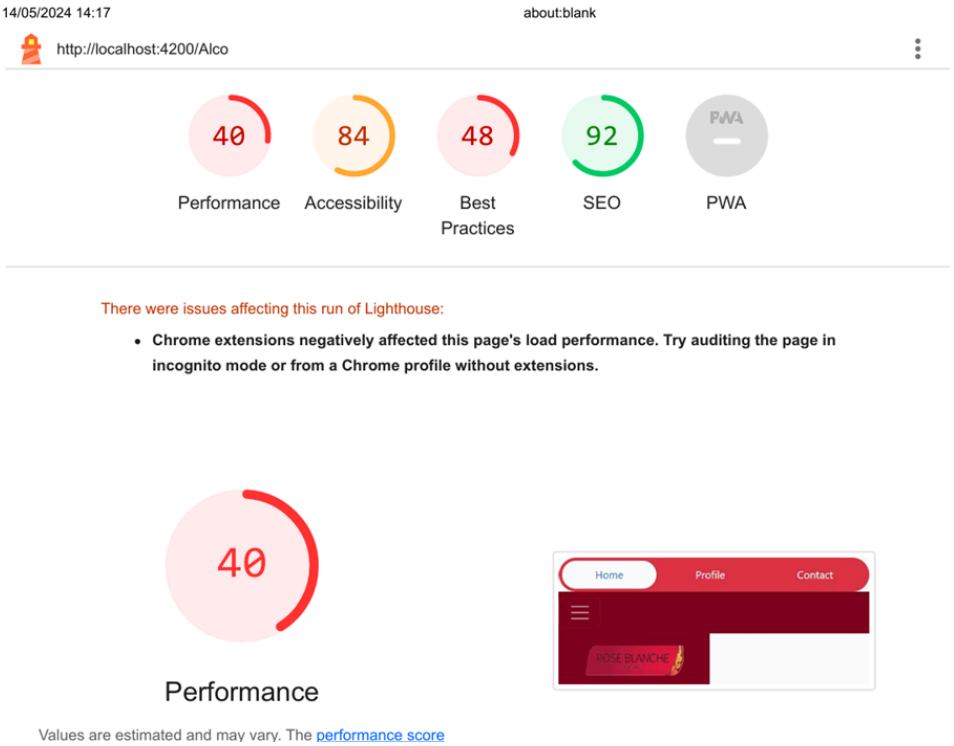


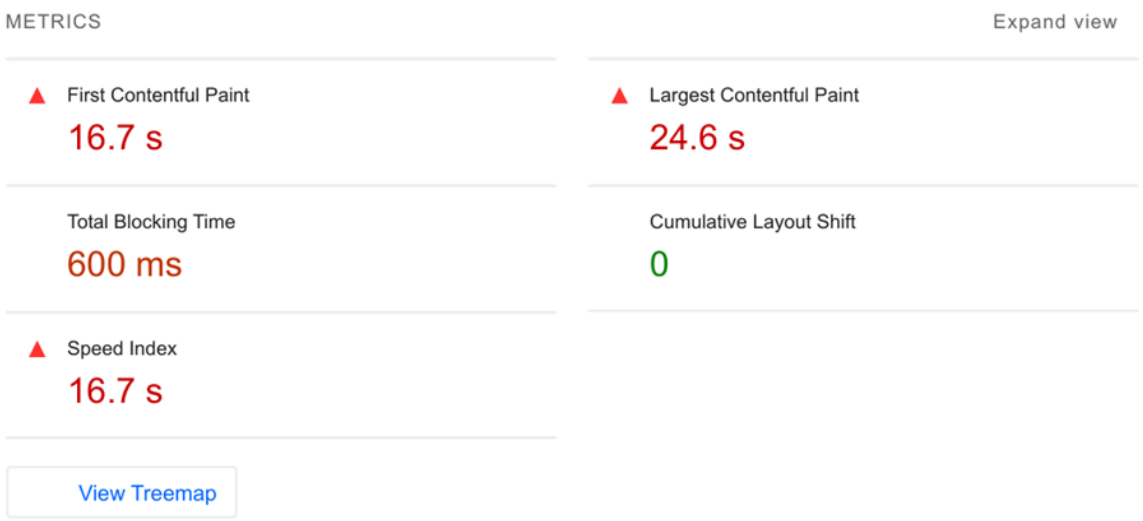
The audit conducted on the web page http://localhost:4200/Sboula revealed insightful performance metrics and overall score. With a global score of 53, the page demonstrated notable strengths in various areas. Performance scored impressively high at 84, indicating efficient loading times and responsiveness. The initial loading time stood at 53 ms, with an even faster 48 ms for mobile devices, showcasing optimized accessibility across platforms. Additionally, the speed index achieved an impressive 92, highlighting swift page rendering. The page excelled in accessibility, best practices, and SEO, scoring 100, 93, and 92, respectively. These scores underscore the adherence to industry standards, ensuring a seamless user experience and optimized visibility on search engines. While the audit result

indicates that the page is not applicable for PWA (Progressive Web App), its robust performance across other key metrics reaffirms its adherence to best practices and commitment to user satisfaction.



- **Alco**





The audit conducted on the web page http://localhost:4200/ALCO yielded insightful performance metrics and overall scores. The Lighthouse performance score stood at 48, indicating areas for potential optimization. Despite this, the page exhibited commendable performance in several aspects. The initial page load time was swift, clocking in at 53 ms, with an even faster load time of 48 ms for mobile devices. This highlights the page's efficiency in catering to diverse user devices and ensuring optimal accessibility. Moreover, the speed index achieved an impressive 92, showcasing rapid page rendering and user interaction. In terms of accessibility, the page excelled with a perfect score of 100, emphasizing its commitment to inclusive design principles. Additionally, the SEO score reached 92, indicating strong visibility and adherence to search engine optimization best practices. While there may be opportunities for enhancement in performance, the ALCO web page demonstrates robust accessibility and SEO compliance, underscoring its dedication to delivering an exceptional user experience and maximizing online visibility.

V. Conclusion

In summary, the integration of a Business Intelligence (BI) system at La Rose Blanche represents a pivotal milestone in enhancing decision-making processes and operational efficiency within the organization. By establishing a meticulously crafted staging area, the company ensures the integrity and accuracy of the data flowing into the data warehouse, thereby laying a robust foundation for generating invaluable insights essential for steering strategic business initiatives. Leveraging the capabilities of Talend further amplifies these efforts by simplifying the complexities associated with data management and transformation. Through Talend's intuitive interface and comprehensive features, La Rose Blanche can efficiently process, cleanse, and integrate data from disparate sources, facilitating a holistic view of business operations and customer insights. This streamlined approach not only enhances the agility and responsiveness of the organization but also empowers decision-makers with timely and actionable insights, enabling them to navigate the complexities of the competitive market landscape effectively. As La Rose Blanche continues to evolve and innovate, the integration of a BI system coupled with the utilization of Talend serves as a catalyst for driving sustainable growth and fostering a data-driven culture within the organization.