UNIVERSITÉ PARIS 13 | USPC Université Sorbonne Paris Cité

FACULTÉ DES SCIENCES DHAR EL MAHRAZ
UNIVERSITÉ SIDI MOHAMED BEN ABDELLAH

Rapport du projet

Business Intelligence

Réalisé par :

Walid Taoutaou

Brahim Oulhaj

Encadré par :

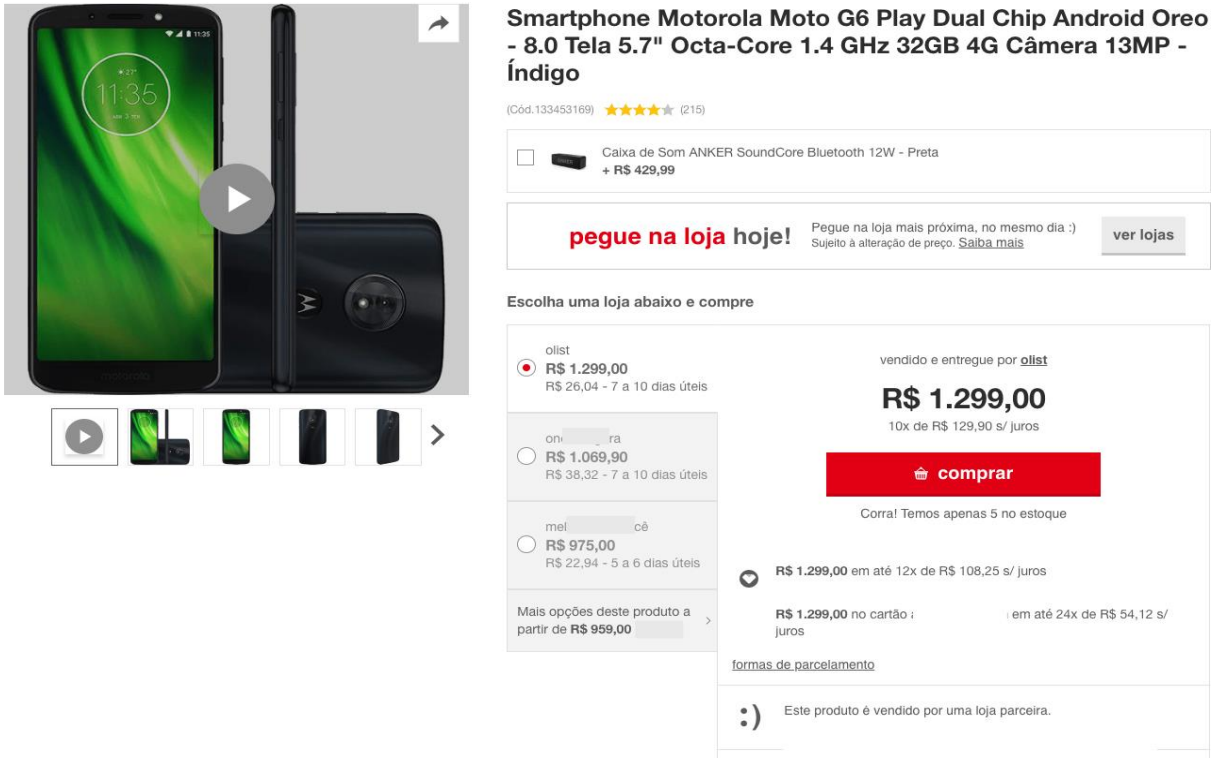
Pr. Riffi Jamal

Sommaire

Introduction et problématique

Le processus décisionnel est un projet qui se construit, il est né d'un besoin exprimé par les entreprises à cause du volume important des données à manipuler et à analyser car le système traditionnel ne suffit plus pour pérenniser l'activité de cette dernière, car celui-ci convient bien aux applications gérant l'activité quotidienne de l'entreprise, mais s'avère inadapté au Décisionnel. En entreposant les données, le processus décisionnel apporte la solution au problème de la croissance continuelle des données provenant de différentes sources et de différents formats en les homogénéisant et en les organisant dans un Data-Warehouse où elles sont historiées, résumées et consolidées. Le volume de données des entrepôts est important et va de centaines de giga-octets à des téraoctets, voir même encore davantage de nos jours.

Notre projet a été réalisé pour faire des statistiques pour un magasin brésilien en ligne, www.olist.com, dont le but d'améliorer le nombre de ventes et d'augmenter le chiffre d'affaire de ce magasin.

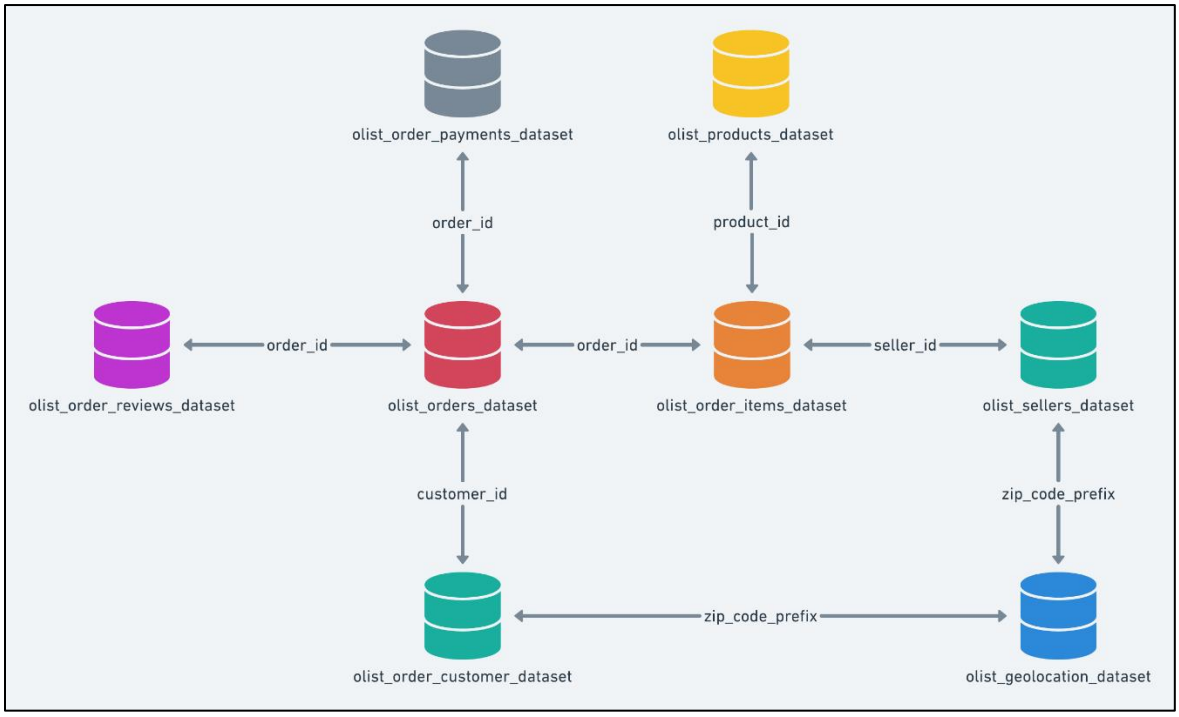


Analyse des données

Olist est un magasin qui permet les clients, dans tous les régions du brésil, d'acheter des produits en ligne. Chaque client a un identifiant et une location, la location se caractérise par un code postal, nom de la ville et la région. Le client peut commander des produits en choisi la quantité qui veut dans une date précise et le produit va être livrer dans une autre date. Les produits sont caractérisés par un identifiant, une catégorie, un nom, une description du produit, nombre des photos de ce produit, le poids en gramme et la langueur et la largeur. Une commande peut avoir un avis (Review) qui se caractérise par un ID, le Score (de 0 à 5), le titre, le message ou le commentaire et la date d'écriture du commentaire.

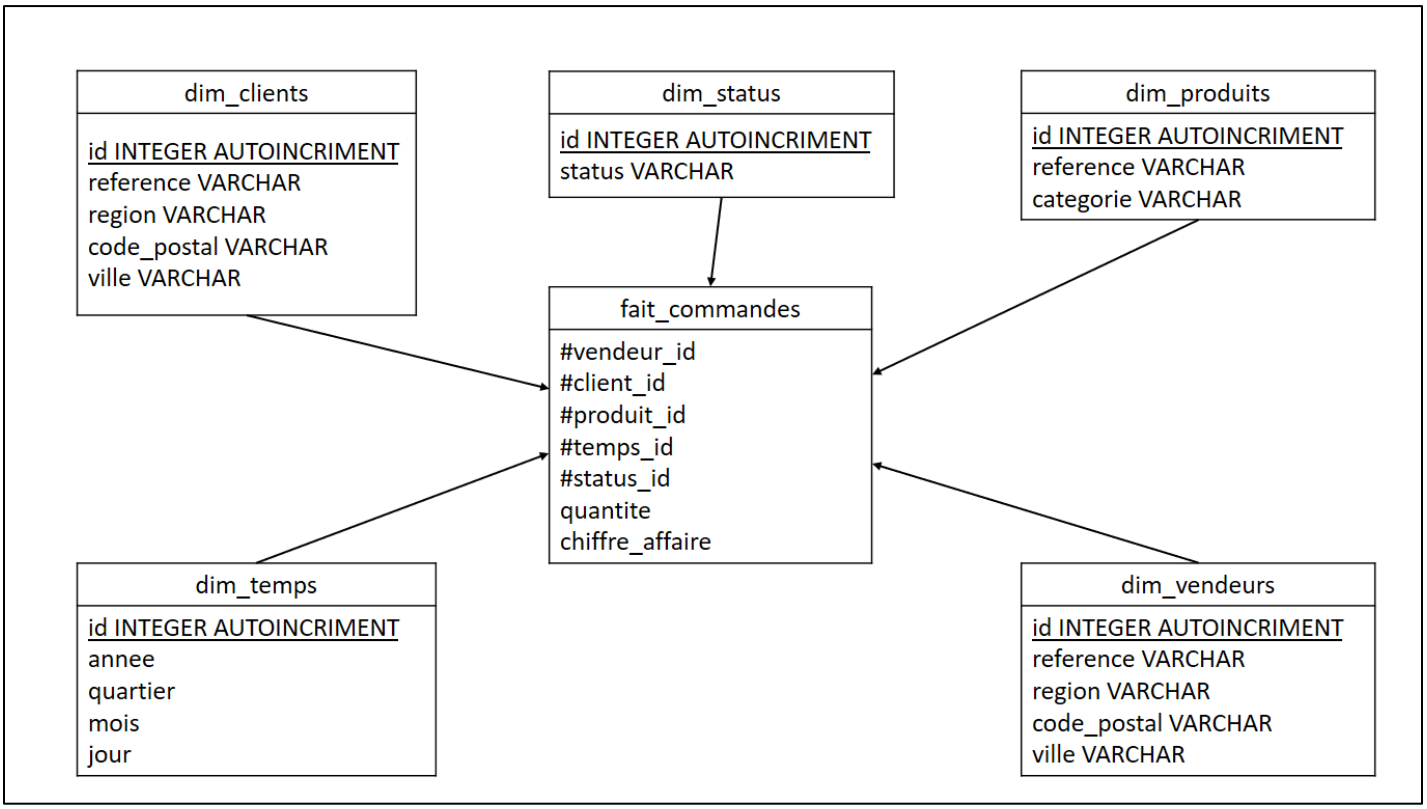
Cet ensemble de données a été fourni par Olist, le plus grand magasin du marché brésilien qui relie les petites entreprises de tout le Brésil. Les commerçants peuvent vendre leurs produits par l'intermédiaire du magasin Olist et les expédier directement aux clients en utilisant les partenaires logistiques Olist.

Il s'agit d'un ensemble de données publiques brésiliennes sur le e-commerce des commandes effectuées à Olist Store. L'ensemble de données contient des informations, en format CSV, sur les 100 k commandes de 2016 à 2018 effectuées sur de multiples marchés au Brésil. Ses caractéristiques permettent de visualiser un ordre de multiples dimensions : de l'état de l'ordre, le prix, les performances de paiement et de fret à l'emplacement du client, les attributs du produit et enfin les commentaires écrits par les clients. Ainsi, un ensemble de données de géolocalisation qui relie les codes postaux brésiliens aux coordonnées lat/lng.



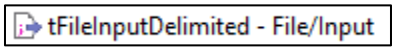
Data integration : ETL

Dans cette partie on va créer un entrepôt de données (Data-Warehouse) en utilisant le logiciel Talend qui permet l'intégration des données. Notre Data Warehouse se compose de cinq tables de dimensions, les tables dim_clients et dim_vendeurs représente la table qui contient les informations dont on a besion des clients et des vendeurs respectivement, la table dim_produits contient les produits vendues, dim_temps contient les dates des commandes des produits et finalement la table dim_status nous donne le statut de la commande est-ce qu'elle est livrée ou pas ou est-ce qu'elle est annulée ..etc, et une seule table de fait, qui représente les commandes effectuer, contient les clés étrangères des tables de dimensions et deux mesures, la quantité commandée et le chiffre d'affaire de la commande. La figure ci-dessous présente le modèle de l'entrepôt de données.

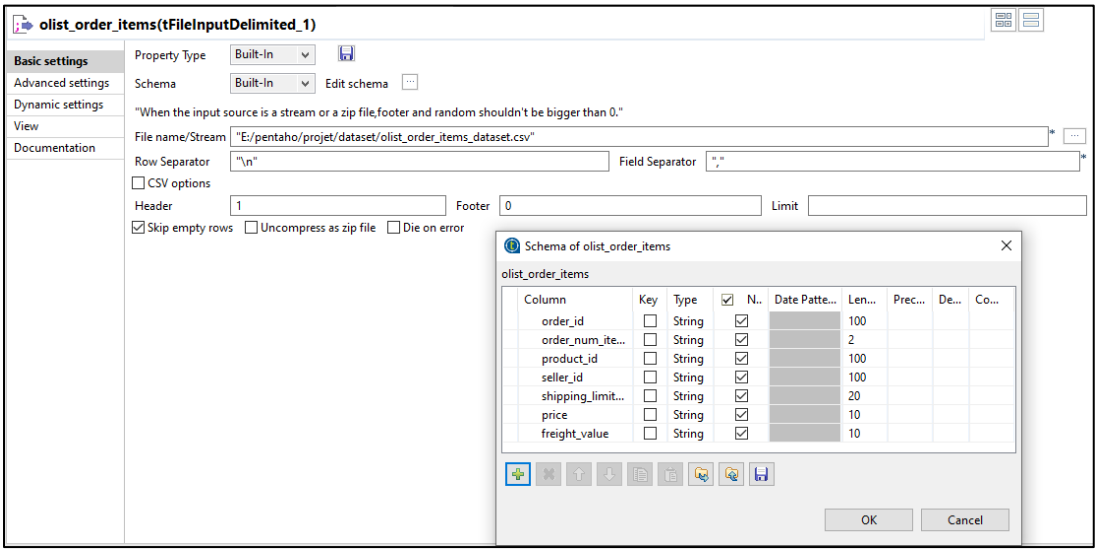


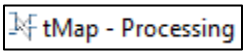
Pour construire cet entrepôt de données en va utiliser le logiciel open source 'Talend', c'est un logiciel qui permet l'intégration des données, en va l'utiliser pour extraire les données de puis les fichiers csv contenant les données du site e-commerce et les transformées en un entrepôt de données se forme des tables dans une base de données en MySQL.

Etape 1 : Extraction des données de puis les fichiers csv.

Dans cette étape on va ajouter un tFileInputDelimiter 

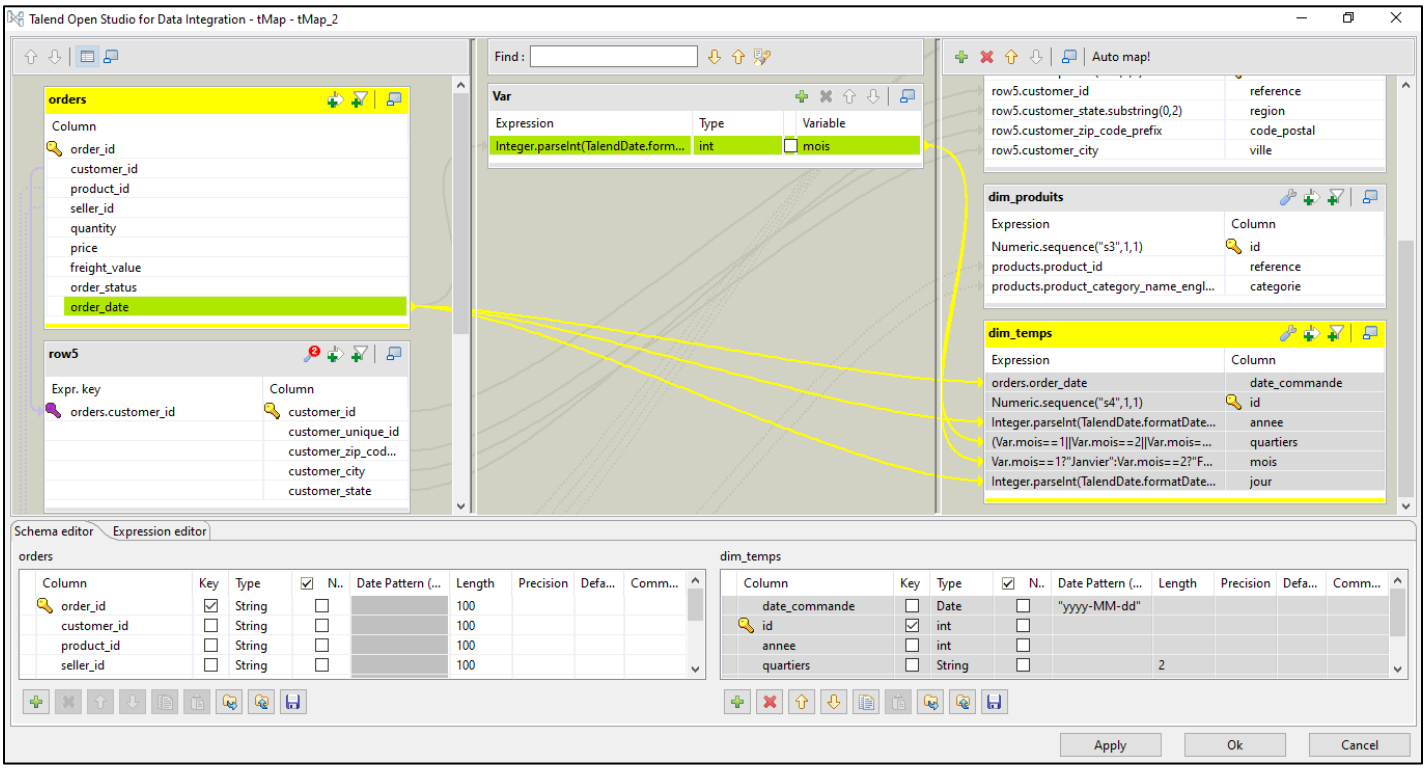
pour charger les données de puis chaque fichier csv.



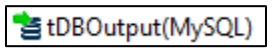
Etape 2 : Transformation des données en utilisant un tMap. 

Le tMap permet de transformer les données des fichiers csv, dans notre cas, à un autre format spécifique, par exemple on va transformer la date de la commande en année, quartier (3 mois), mois et jour.

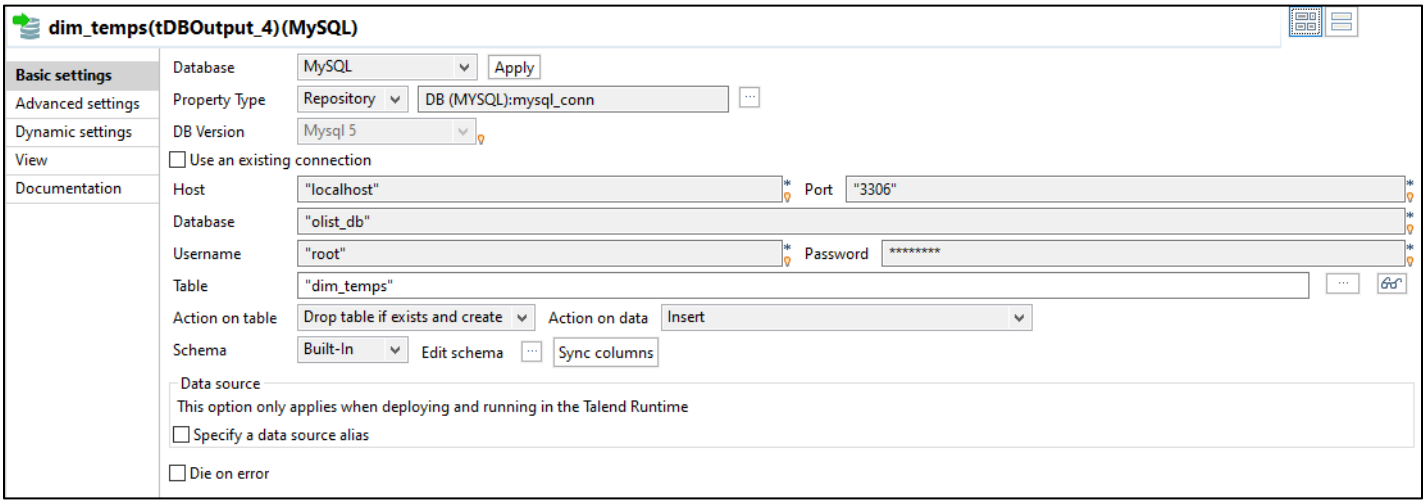
Cette figure représente la transformation des données des fichiers csv (à gauche) au données qui l'on va insérer dans des tables MySQL (à droite).

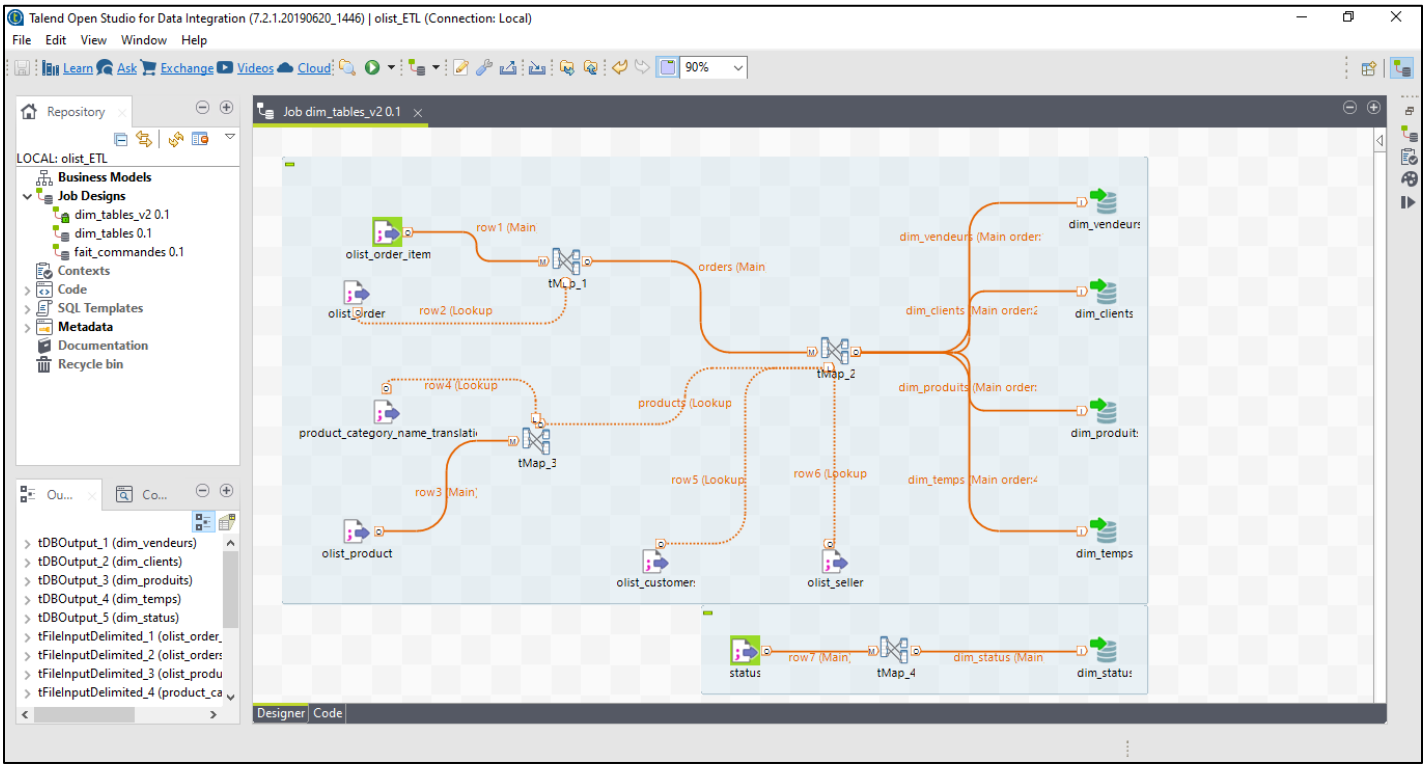


Etape 3 : Sauvegarder les données dans des tables MySQL.

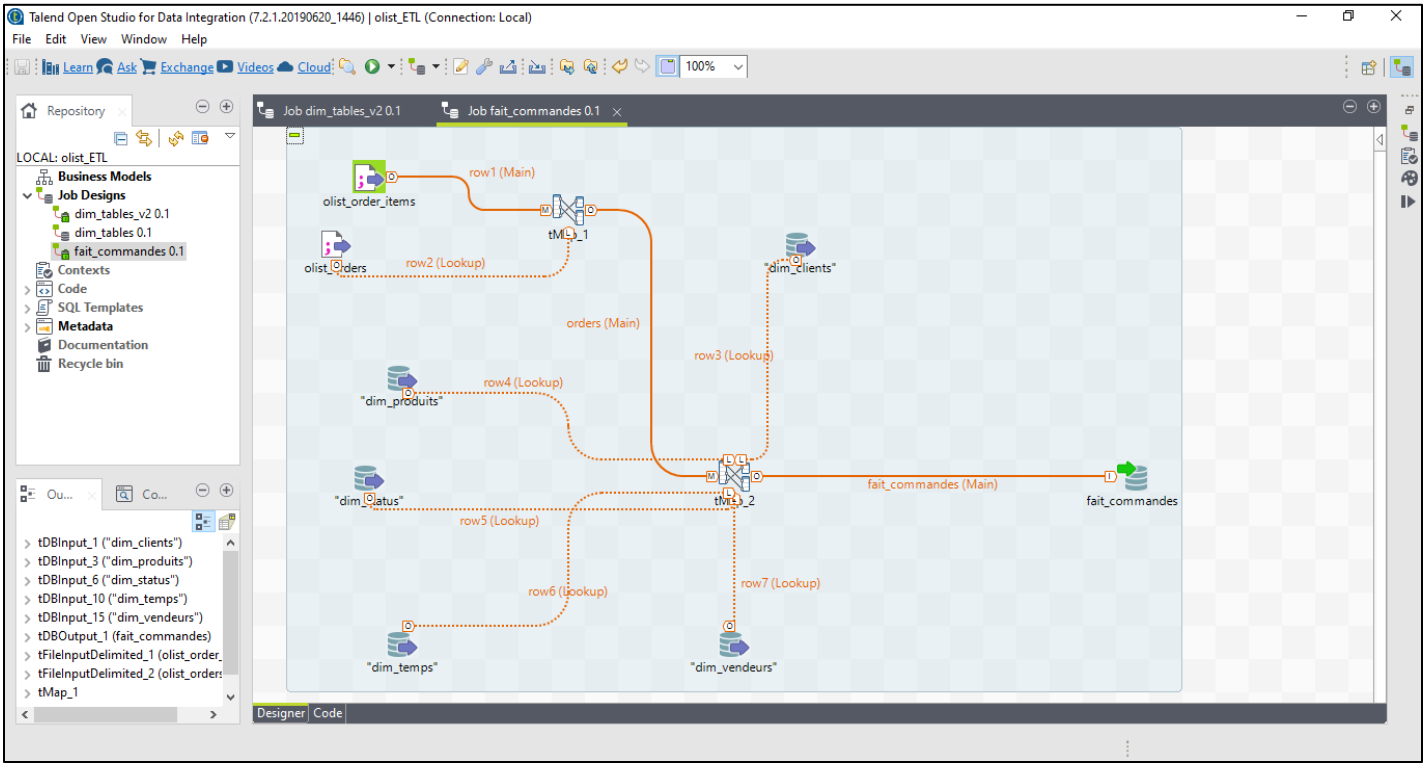
Dans cette étape on va utiliser un autre type de métadonnée, c'est 

, pour se connecter à une base de données déjà créer dans le SGBD MySQL.





Les tables de dimensions



La table de fait

Après l'exécution en va avoir des nouvelles tables dans la base de données MySQL :

	Table	Action	Rows	Type	Collation	Size	Overhead
☐	dim_clients	Browse Structure Search Insert Empty Drop	~102,327	InnoDB	utf8_unicode_ci	8.5 MiB	-
☐	dim_produits	Browse Structure Search Insert Empty Drop	~102,028	InnoDB	utf8_unicode_ci	7.5 MiB	-
☐	dim_status	Browse Structure Search Insert Empty Drop	7	InnoDB	utf8_unicode_ci	16.0 KiB	-
☐	dim_temps	Browse Structure Search Insert Empty Drop	~102,373	InnoDB	utf8_unicode_ci	5.5 MiB	-
☐	dim_vendeurs	Browse Structure Search Insert Empty Drop	~102,225	InnoDB	utf8_unicode_ci	8.5 MiB	-
☐	fait_commandes	Browse Structure Search Insert Empty Drop	~102,306	InnoDB	utf8_unicode_ci	6.5 MiB	-
	6 tables	Sum	~511,266	InnoDB	utf8_unicode_ci	36.6 MiB	0 B

Cube OLAP

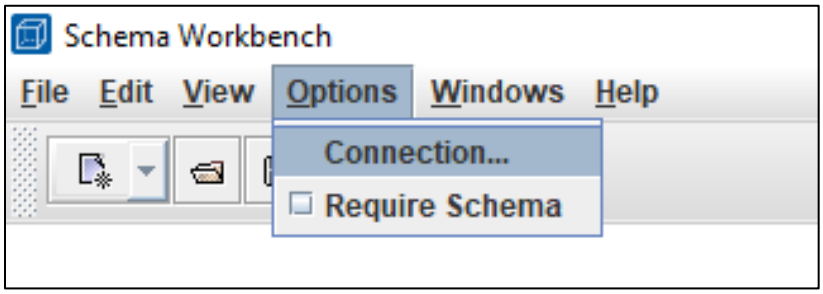
On va utiliser, dans cette partie, le logiciel '**Pentaho Schema Workbench**' pour créer le Cube OLAP, qu'est un fichier XML sous la forme ci-dessous :

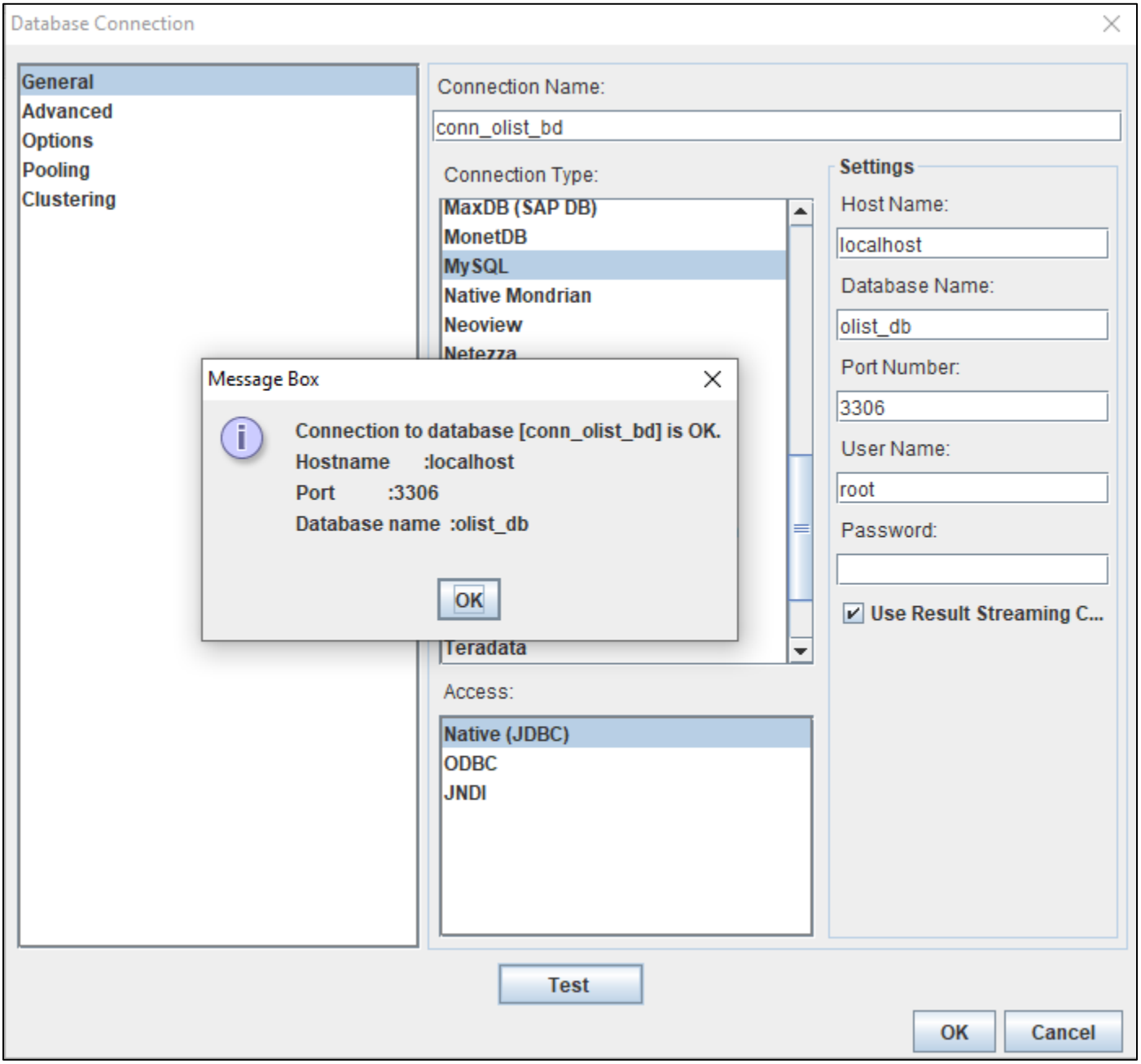
Le Cube OLAP contient au minimum une table de fait, une table de dimension et une mesure.

```
<Schema name="olist_schema" description="mini projet BI">
  <Cube name="cube_commands" visible="true" cache="true" enabled="true">

    <Table name="fait_commandes">
    </Table>

    <Dimension type="TimeDimension" visible="true" foreignKey="temps_id" highCardinality="false" name="dim_temps">
      <Hierarchy name="temps" visible="true" hasAll="true" primaryKey="id">
        <Table name="dim_temps">
        </Table>
        <Level name="annee" visible="true" table="dim_temps" column="annee" nameColumn="annee">
        </Level>
        <Level name="mois" visible="true" table="dim_temps" column="mois" nameColumn="mois">
        </Level>
        .
        .
      </Hierarchy>
    </Dimension>
    .
    .
    <Measure name="quantite" column="quantite" aggregator="sum" visible="true">
    </Measure>
    .
    .
  </Cube>
</Schema>
```

Etape 1 : connexion à la base de données MySQL.

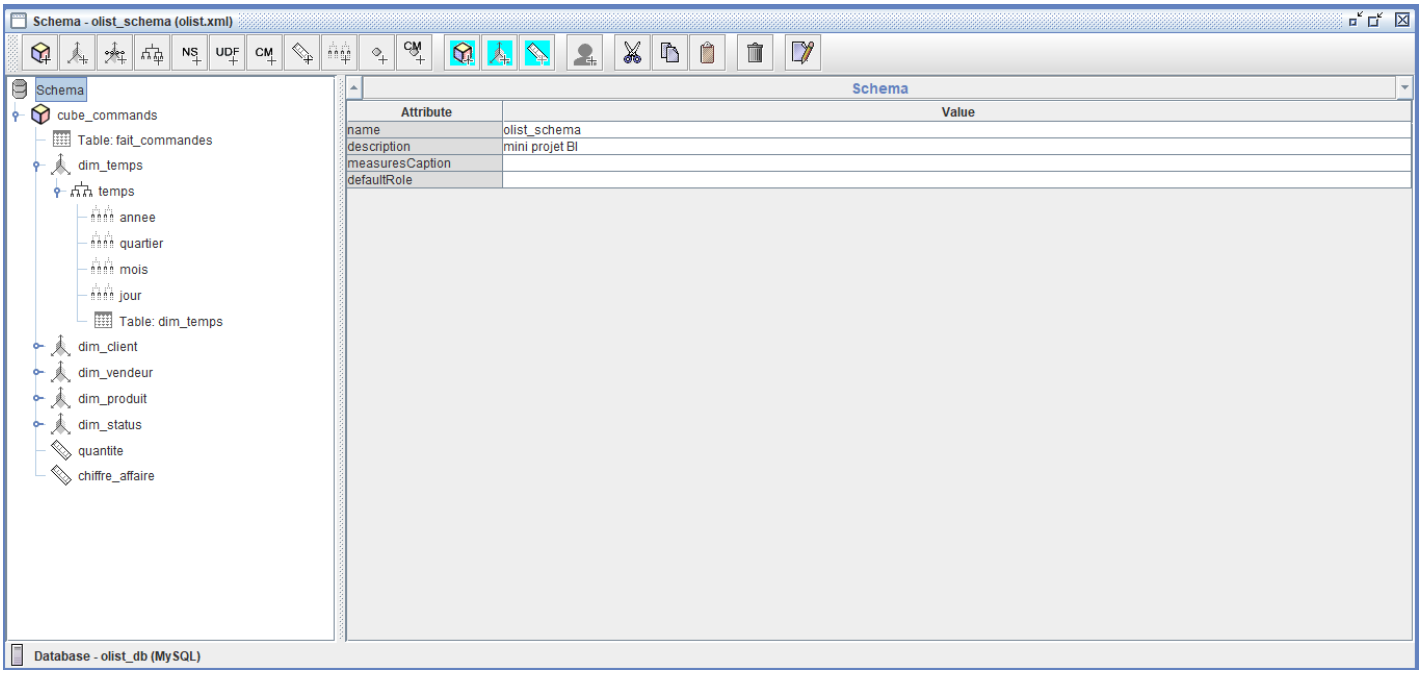
La 1ère étape pour créer un cube OLAP il faut connecter Schema Workbench avec la base de données MySQL dont elle y a les tables de dimensions et la table de fait.




Database Connection
General
Advanced
Options
Pooling
Clustering
Connection Name:
conn_olist_bd
Connection Type:
MaxDB (SAP DB)
MonetDB
MySQL
Native Mondrian
Neoview
Netezza
Teradata
Settings
Host Name:
localhost
Database Name:
olist_db
Port Number:
3306
User Name:
root
Password:
Use Result Streaming C...
Message Box
Connection to database [conn_olist_bd] is OK.
Hostname :localhost
Port :3306
Database name :olist_db
OK
Access:
Native (JDBC)
ODBC
JNDI
Test
OK
Cancel

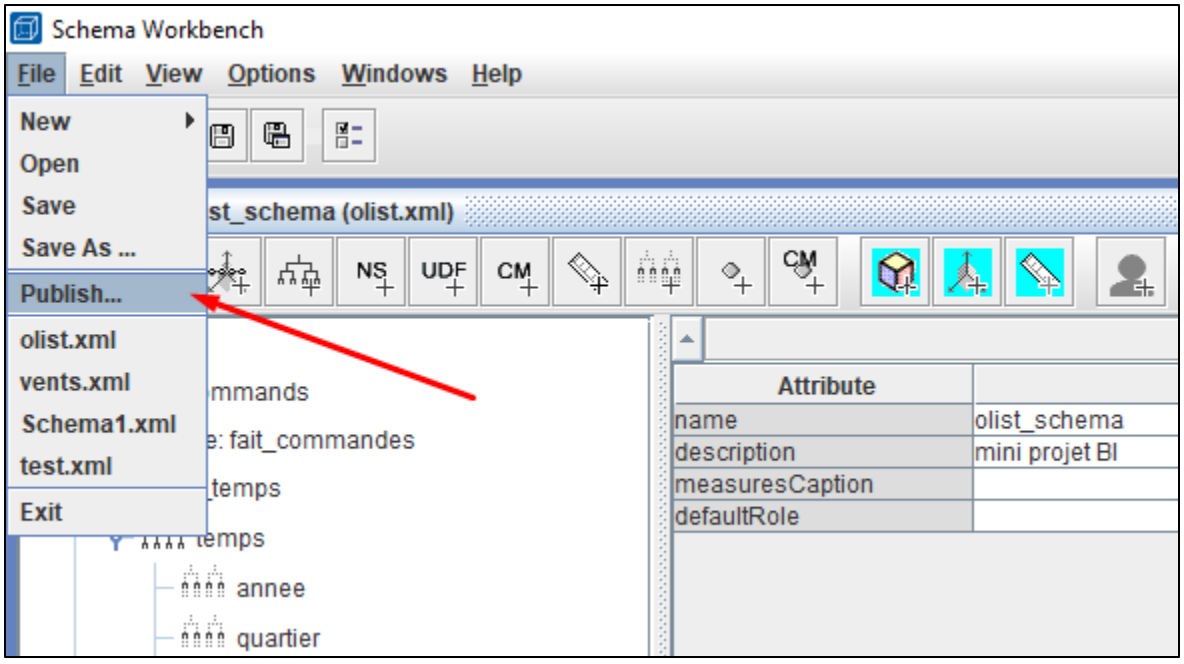
Etape 2 : créer le cube OLAP.

Le cube OLAP contient au minimum une table de fait, une table de dimension et une mesure.



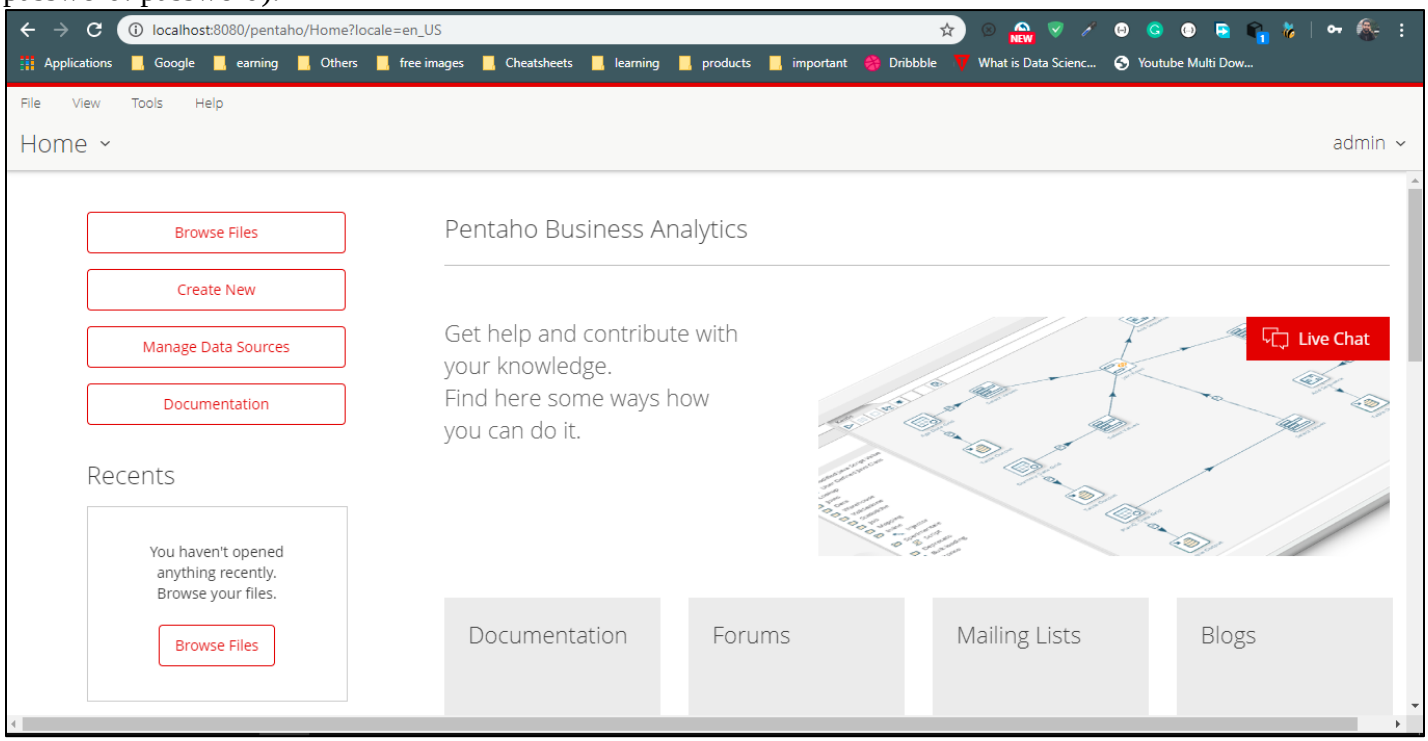
Etape 3 : Publication du cube.

Après la création du cube OLAP il faut le publier dans un serveur qui permet de créer des statistiques et des rapports, on va utiliser dans ce cas le serveur de Pentaho (Pentaho Server) avec le plugin Pivot4J qui facilite la tâche de Reporting.

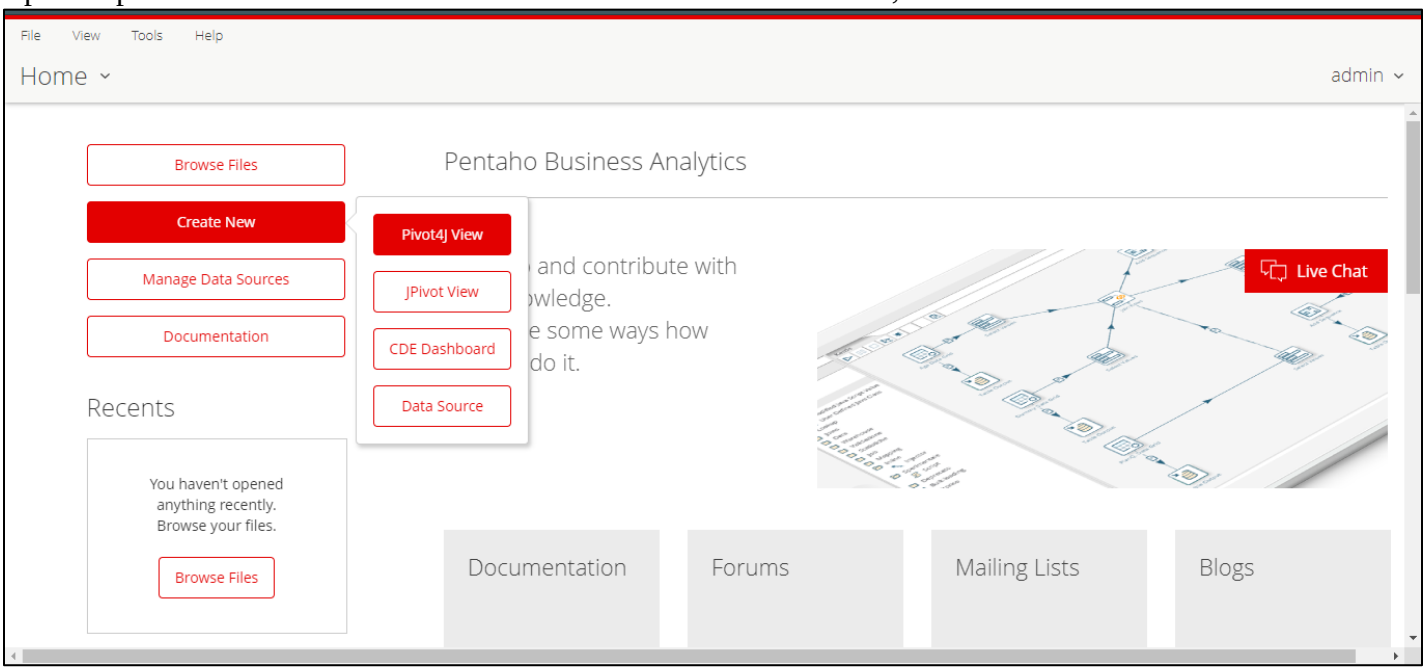


Reporting

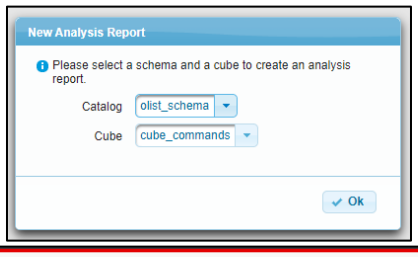
Pour aborder cette partie, il faut d'abord installer Pentaho Server et ajouter le plugin Pivot4J, puis on démarre le serveur et entrer le Url, localhost:8080/pentaho, dans le navigateur et s'authentifier (pseudo: admin, password: password).

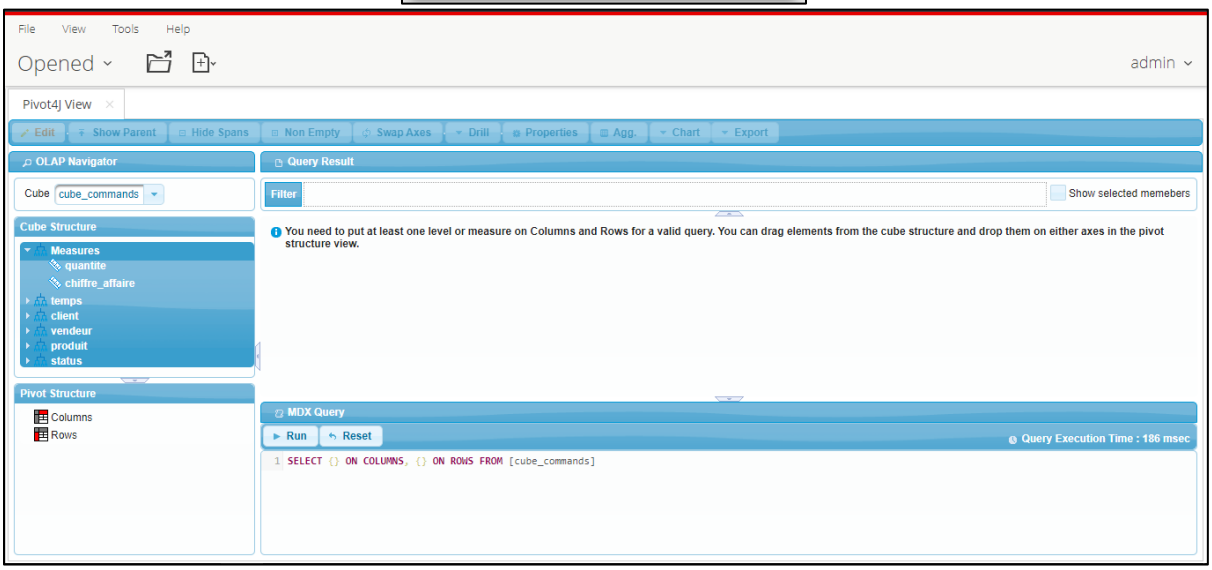


Après la publication du Cube OLAP via Pentaho Schema Workbench, on va créer une nouvelle vue Pivot4J.



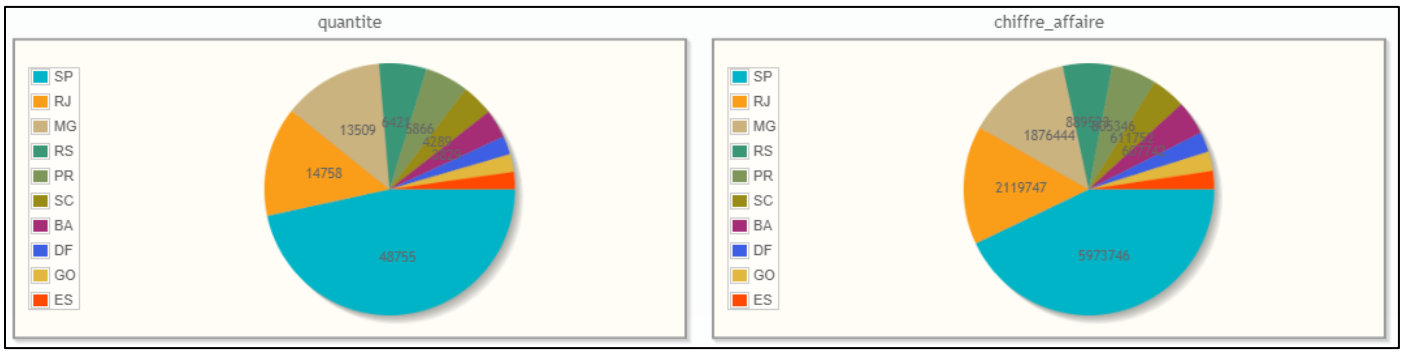
Après on va sélectionner le cube et on clique sur Ok.





Maintenant nous pouvons aborder la réalisation du rapport en répondant aux questions ci-dessous.

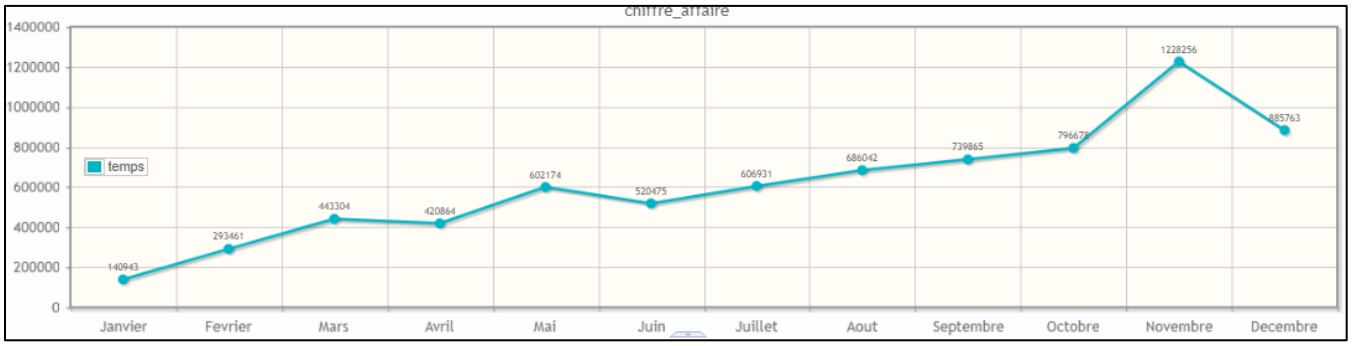
- **Quelles sont les chiffres d'affaire et les quantités des 10 tops régions de Brésil ?**



la requête MDX correspondante à cette question :

```
SELECT {[Measures].[quantite], [Measures].[chiffre_affaire]} ON COLUMNS, TOPCOUNT(Order({[dim_client.client].[AC], [dim_client.client].
[AL], [dim_client.client].[AM], [dim_client.client].[AP], [dim_client.client].[BA], [dim_client.client].[CE], [dim_client.client].[DF],
[dim_client.client].[ES], [dim_client.client].[GO], [dim_client.client].[MA], [dim_client.client].[MG], [dim_client.client].[MS],
[dim_client.client].[MT], [dim_client.client].[PA], [dim_client.client].[PB], [dim_client.client].[PE], [dim_client.client].[PI],
[dim_client.client].[PR], [dim_client.client].[RJ], [dim_client.client].[RN], [dim_client.client].[RO], [dim_client.client].[RR],
[dim_client.client].[RS], [dim_client.client].[SC], [dim_client.client].[SE], [dim_client.client].[SP], [dim_client.client].[TO]},
[Measures].[chiffre_affaire], DESC), 10.0) ON ROWS FROM [cube_commands] WHERE [dim_status.status].[delivered]
```

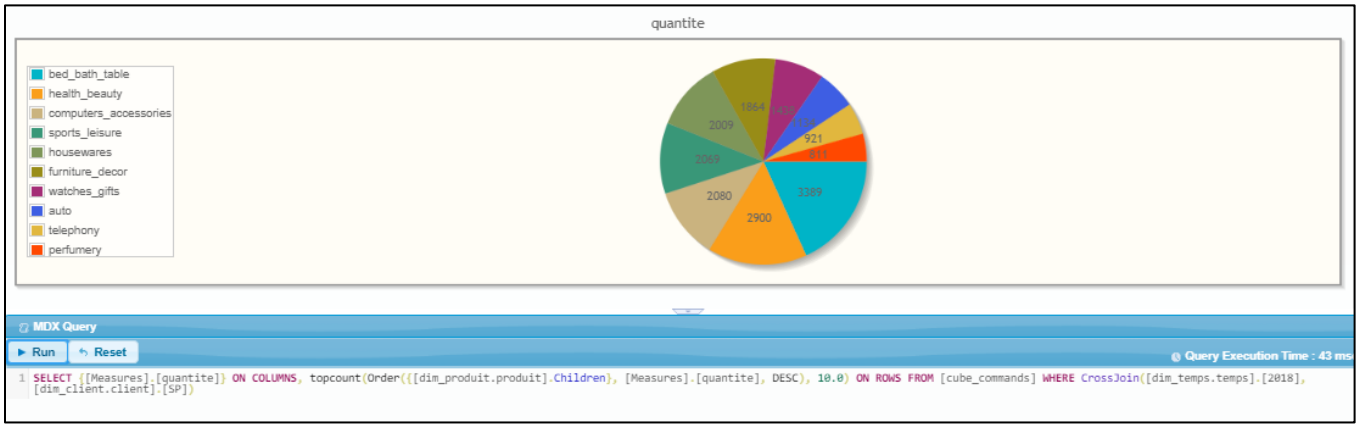
- **Comment le chiffre d'affaire s'évolue en fonction des mois dans en 2017 ?**



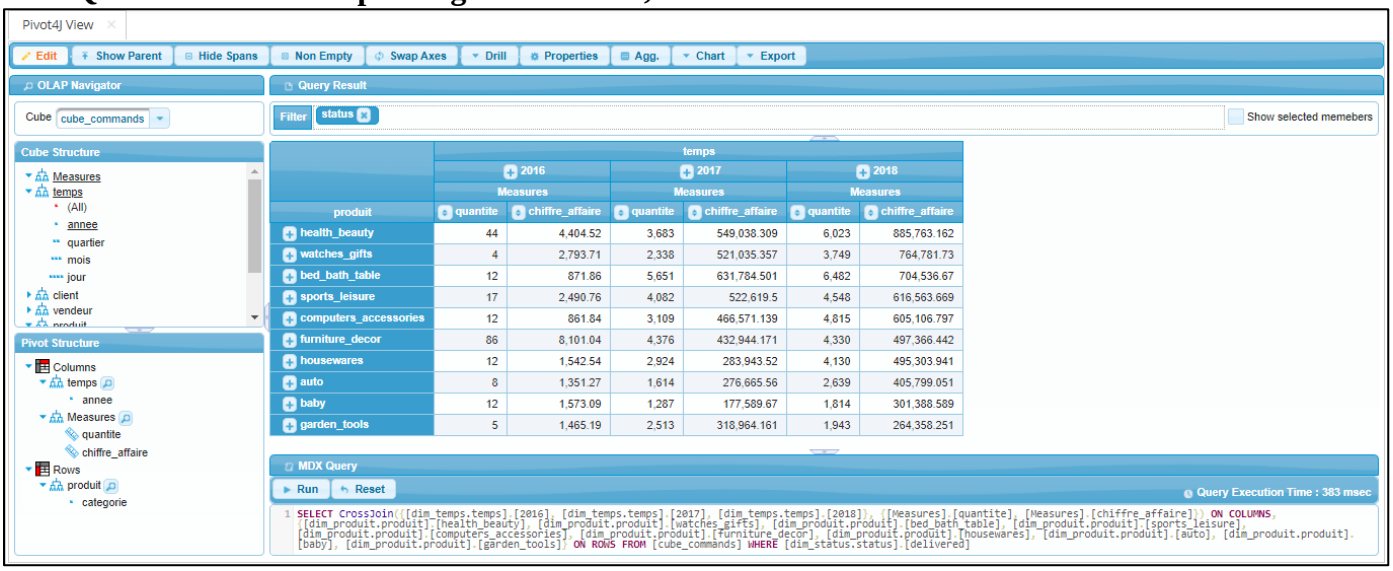
la requête correspondante :

```
SELECT {[Measures].[chiffre_affaire]} ON COLUMNS, {[dim_temps.temps].[2017].[Q1].[Janvier], [dim_temps.temps].[2017].[Q1].[Fevrier], [dim_temps.temps].[2017].[Q1].[Mars],
[dim_temps.temps].[2017].[Q2].[Avril], [dim_temps.temps].[2017].[Q2].[Mai], [dim_temps.temps].[2017].[Q2].[Juin], [dim_temps.temps].[2017].[Q3].[Juillet], [dim_temps.temps].[2017].
[Q3].[Aout], [dim_temps.temps].[2017].[Q3].[Septembre], [dim_temps.temps].[2017].[Q4].[Octobre], [dim_temps.temps].[2017].[Q4].[Novembre], [dim_temps.temps].[2017].[Q4].[Decembre]}
ON ROWS FROM [cube_commands]
```

- **Quelle sont les tops 10 catégories des produits qui ont été vendu dans la région de Sao Paulo en 2018 ?**



- **Quelles sont les 10 tops catégories en 2016, 2017 et 2018 ?**



Data Mining

Le *Data Mining* est en fait un terme générique englobant toute une famille d'outils facilitant l'exploration et l'analyse des données contenues au sein d'une base décisionnelle de type Data Warehouse ou DataMart. Les techniques mises en action lors de l'utilisation de cet instrument d'analyse et de prospection sont particulièrement efficaces pour extraire des informations significatives depuis de grandes quantités de données.

Dans cette partie de Data Mining on va étudier la possibilité de vendre un produit appartient à une catégorie précise dans un mois précis à un client dans une région précise.

Pour faire cette étape on a extrait les informations nécessaires pour trainer le modèle, en utilisant les tables de dimensions et la table de fait :

```sql
SELECT dp.categorie, dt.mois, dv.region v_reg, dc.region c_reg, SUM(fc.chiffre_affaire) ca
FROM fait_commandes fc
JOIN dim_vendeurs dv ON dv.id = fc.vendeur_id
JOIN dim_clients dc ON dc.id = fc.client_id
JOIN dim_temps dt ON dt.id = fc.temps_id
JOIN dim_produits dp ON dp.id = fc.produit_id
JOIN dim_status ds ON ds.id = fc.status_id
WHERE ds.status IN ('delivered','shipped','approved') AND dp.categorie IS NOT NULL
GROUP BY dp.categorie, dt.mois, dv.region, dc.region
```

Puis l'enregistrer sous format CSV pour appliquer l'algorithme de Data Mining, l'arbre de décision dans notre cas.
Un arbre de décision est un outil fort pratique lorsqu'il s'agit de répartir une population en groupes homogènes selon des critères bien précis, les variables de segmentation.

Après la préparation du fichier csv contenant le Dataset, on va le lire en utilisant la bibliothèque Pandas de Python, puis ajouter un autre colonne « classe » contient des 0 si le CA est inférieur à un seuil, donc la possibilité du vente et faible, et en 1 sinon, donc le vente est très possible.
Le seuil qu'on a choisi la moyenne de la valeur absolue du CA moins la moyenne des CA.

```
## --Creation de classes
data['classe'] = (data['ca'] > np.mean(abs(data['ca'] - np.mean(data['ca'])))).astype(int)
data.head(4)
```

	categorie	mois	v_reg	c_reg	ca	classe
0	agro_industry_and_commerce	Aout	MG	BA	1134.439941	1
1	agro_industry_and_commerce	Aout	MG	MG	164.289993	0
2	agro_industry_and_commerce	Aout	PR	SP	1124.659988	1
3	agro_industry_and_commerce	Aout	SC	PR	47.620003	0

Ensuite, on a transformé les données texte en nombres pour que l'algorithme puissent s'entrainer avec elles.

```
categories = dict()
mois = dict()
v_reg = dict()
c_reg = dict()
for i in range(len(np.unique(data['categorie']))): categories[np.unique(data['categorie'])[i]] = i
for i in range(len(np.unique(data['mois']))): mois[np.unique(data['mois'])[i]] = i
for i in range(len(np.unique(data['v_reg']))): v_reg[np.unique(data['v_reg'])[i]] = i
for i in range(len(np.unique(data['c_reg']))): c_reg[np.unique(data['c_reg'])[i]] = i
```

```
data['categorie'] = np.array([categories[c] for c in data['categorie']])
data['mois'] = np.array([mois[c] for c in data['mois']])
data['v_reg'] = np.array([v_reg[c] for c in data['v_reg']])
data['c_reg'] = np.array([c_reg[c] for c in data['c_reg']])
data
```

Notre dataset maintenant est prêt, il faut juste séparer les caractéristiques et les classes, et les réparties en Train et Test, et de s'entrainer le modèle avec l'algorithme d'arbre de décision et le tester avec les données de teste pour voire la précision du modèle.

```
#------------ load data : ------------
print('loading data ...')
x = data.iloc[:,0:-2]
y = data.iloc[:,-1]
#------------ split data : ------------
x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.3)
#------------ SVM ------------
#clf = SVC(C=10000, kernel = 'rbf')
#clf.fit(x,y)
#print(f'Train accuracy: {clf.score(x,y)*100}%')
#joblib.dump(clf, fname)
#------------ DTC ------------
clf = DecisionTreeClassifier()
clf.fit(x_train,y_train)
print(f'Train accuracy: {clf.score(x_train,y_train)*100}%')
# ----- test score :------

y_pred = clf.predict(x_test)
print(f'test accuracy: {accuracy_score(y_pred, y_test)*100}%')
```

```
loading data ...
Train accuracy: 100.0%
test accuracy: 82.07911683532659%
```


On peut maintenant utiliser ce modèle pour prédire, pour un vendeur, s'il peut vendu son produit appartient à une catégorie dans une région à un mois précises.

Conclusion

L'informatique décisionnelle est un domaine en pleine évolution pour l'aide à la prise de décisions aux seins des entreprises. Elle est devenue un besoin capital, car la simple logique de produire pour répondre à une demande, ne suffit plus pour pérenniser l'activité de celle-ci, de ce fait sont apparus les entrepôts de données.

A travers ce projet, nous avons exposé les étapes de mise en place et de conception d'un entrepôt de données, qui facilitera la prise de décisions et permettre aux décideurs tout comme aux utilisateurs de sélectionner et filtrer des données utiles à la prise de décision.

Nous avons réparti ce projet en 6 parties principales.

- Dans la première partie, nous avons essayé de chercher des données et les comprendre en suite on a créé le modèle Entité-Association pour bien présenter les données pour faciliter la tâche de la deuxième partie.
- Nous avons créé le schéma en étoile de notre entrepôt de données.
- On a attaqué la partie ETL, en utilisant l'outils Talend.
- On a créé le Cube OLAP en utilisant Pentaho Schema Workbench.
- Publier le cube dans le serveur Pentaho pour faire des histogrames, des courbes … en utilisant le plugin Pivot4J.
- En fin, on a transformé les données du Data Warehouse en un fichier CSV contenant les informations nécessaires pour appliquer l'algorithme de Data Mining.