



RAPPORT DE TRAVAUX PRATIQUES

Filière : 2ITE

Niveau : 2ère Année

Analyse des données avec le serveur
Mondrian

Réalisé par :

Marcel Fassou HABA

Encadré par :

Pr. Hanine

INTRODUCTION

L'analyse de données occupe une place essentielle dans la prise de décision des organisations. Avec la croissance des volumes d'informations, il devient nécessaire d'utiliser des outils performants pour préparer, structurer et explorer les données. Les systèmes **OLAP (Online Analytical Processing)** répondent à ce besoin en permettant une analyse multidimensionnelle efficace.

Le **serveur OLAP Mondrian** s'inscrit dans cette logique. C'est un moteur open source développé en Java, capable d'interroger des entrepôts de données via le langage **MDX**. Intégré à des outils comme **JasperReport**, il facilite la création de rapports et tableaux de bord interactifs, permettant une exploration dynamique des données selon plusieurs axes d'analyse.

Ce **travail pratique** vise à illustrer l'utilisation du moteur **Mondrian** pour la préparation, l'analyse et l'exploration d'une grande quantité de données, à travers son intégration avec **JasperReport**. L'objectif principal de ce TP est savoir comment utiliser le serveur OLAP Mondrian pour la préparation de l'analyse et l'exploration d'une large quantité de données.

Le TP se divise en deux parties :

- **Première partie** : présentation de l'outil Mondrian et de son intégration dans un environnement BI.
- **Seconde partie** : réalisation pratique incluant la **création de la base de données**, la **configuration de JasperReport**, et l'**intégration du moteur Mondrian** pour l'analyse multidimensionnelle.

Ce travail permettra de mieux appréhender le processus complet d'analyse de données et l'apport du serveur OLAP Mondrian dans la prise de décision.

PRÉSENTATION DE L'OUTIL UTILISÉ

Le **serveur OLAP Mondrian** est un moteur **d'analyse multidimensionnelle** open source développé en **Java**, largement utilisé dans le domaine de la **Business Intelligence (BI)**. Il permet d'explorer, d'agréger et d'analyser de grandes quantités de données issues d'une **base relationnelle**, en les organisant sous forme de **cubes OLAP**. Cette représentation multidimensionnelle facilite la compréhension des phénomènes complexes en offrant une vision globale et détaillée selon différents axes d'analyse (temps, produit, région, etc.).

Mondrian repose sur le langage **MDX (Multidimensional Expressions)**, spécialement conçu pour interroger et manipuler les cubes OLAP. Grâce à lui, les utilisateurs peuvent effectuer des requêtes analytiques avancées, réaliser des comparaisons temporelles, ou encore observer l'évolution des indicateurs de performance.

JasperReport est un puissant outil de **reporting et de visualisation de données**, également open source. Il permet de créer des **rapports dynamiques, interactifs et visuellement attractifs** à partir de diverses sources de données. Son intégration avec Mondrian renforce considérablement la capacité d'analyse en combinant la **puissance de calcul OLAP** et la **richesse visuelle** du reporting.

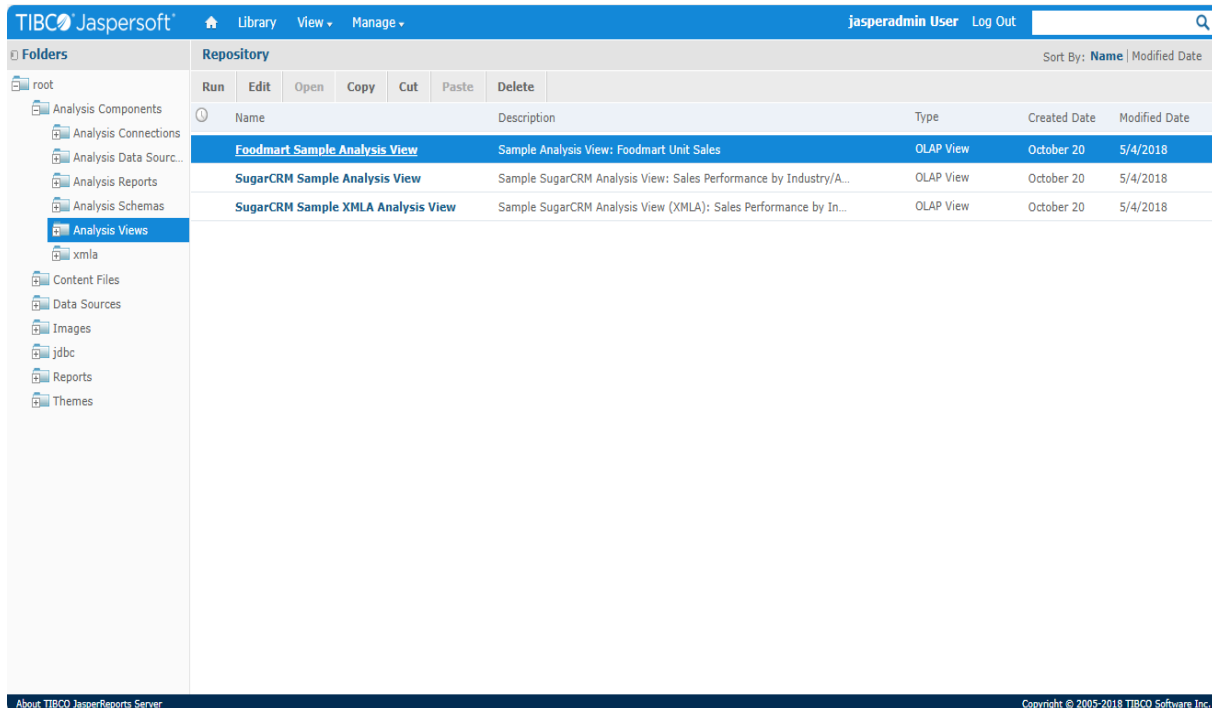
Dans cette intégration :

- **Mondrian** assure la **préparation et l'analyse multidimensionnelle** des données.
- **JasperReport** fournit une **interface graphique conviviale**, permettant de présenter les résultats sous forme de **rapports, tableaux de bord et graphiques interactifs**.

Cette collaboration offre un environnement complet où l'utilisateur peut **naviguer librement dans les données**, explorer les relations entre variables, effectuer des

analyses en profondeur (drill-down, roll-up, slice, dice), et produire des rapports clairs et pertinents pour la prise de décision.

La figure ci-dessous représente l'interface de JasperReport à travers JasperServer.



Atouts de Mondrian et JasperReport

- **Open source**, multiplateforme et extensible.
- Intégration fluide entre le moteur d'analyse OLAP et le module de reporting.
- Grande **puissance analytique** pour traiter et agréger de larges volumes de données.
- Visualisation interactive facilitant la compréhension des tendances et des indicateurs clés.

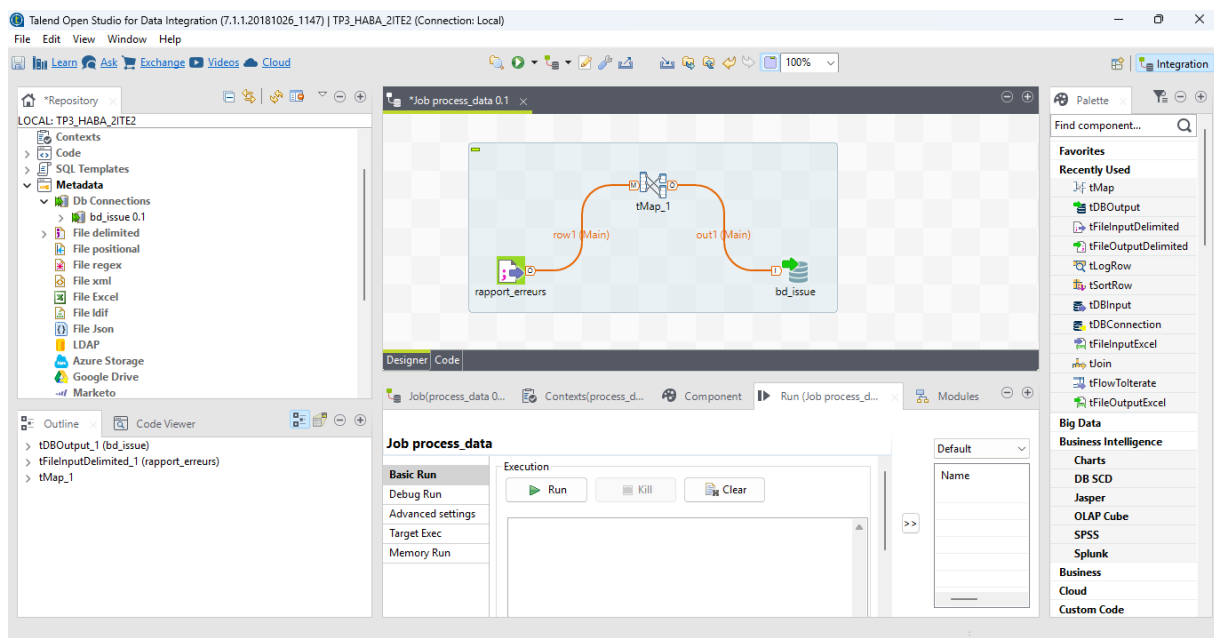
Ainsi, la combinaison **Mondrian–JasperReport** constitue une solution BI complète, offrant à la fois la **puissance de l'analyse multidimensionnelle** et la **clarté du reporting interactif**, indispensable pour explorer efficacement les données et soutenir les décisions stratégiques.

RÉALISATION ET RESULTATS

Création de la base de données

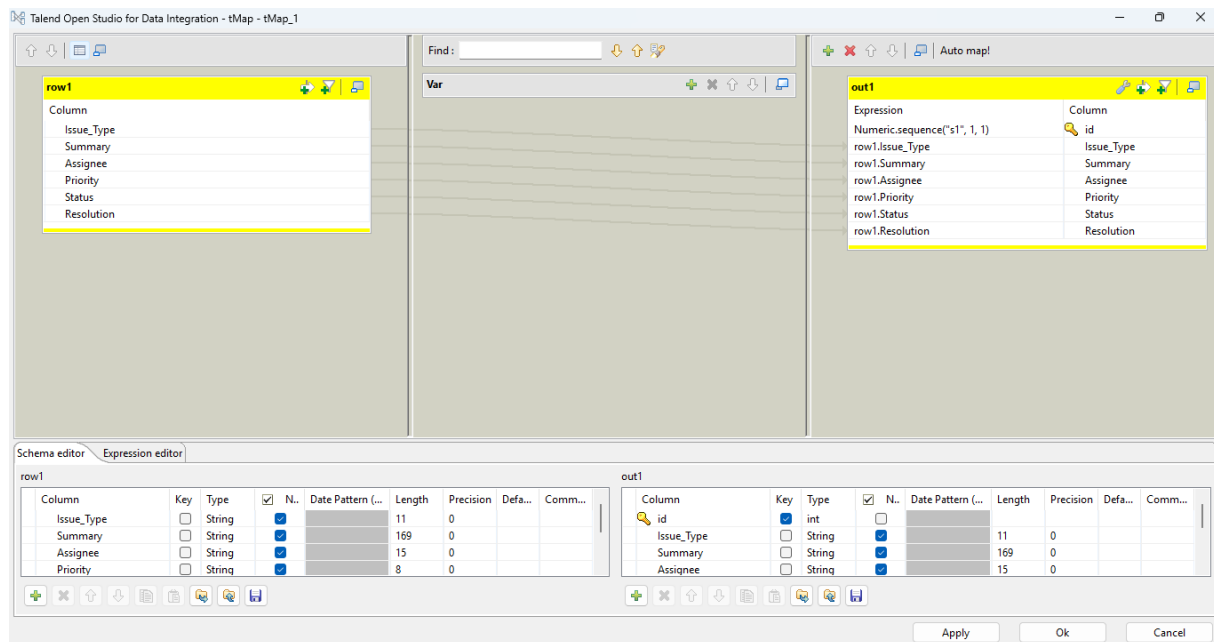
Dans cette partie, considérer comme la première étape, nous allons créer une base de données, en utilisant Talend Data Intégration, un outil que nous avons déjà utilisé dans le précédent TP. Nous allons copier les données du fichier délimité Rapport_Erreurs.csv dans une base de données mysql. Pour cela nous allons créer un job, ce job va nous permettre de mettre en place un pipeline de la préparation des sources de données à leur chargement dans la base de données (Processus ETL).

Cette image représente le job dédié à ce processus.



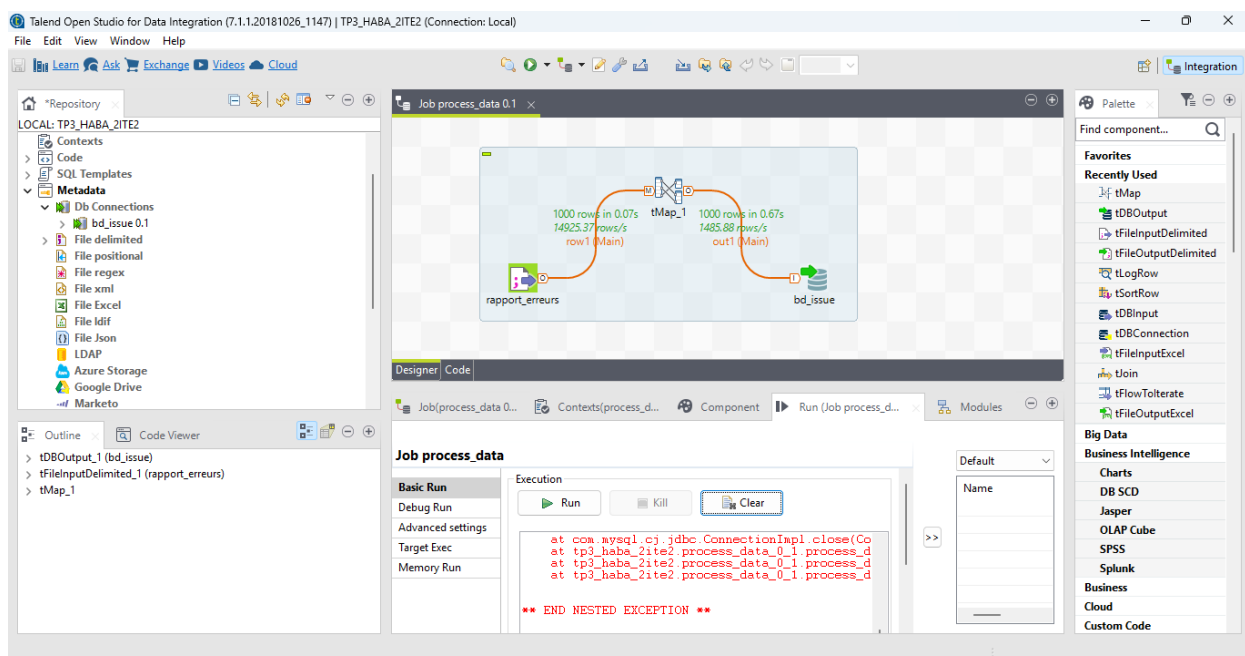
Etant donné que les données de départ manquent une colonne qui l'id, on utilise le composant **tMap** pour permettre l'ajout d'une nouvelle colonne id, qui sera généré automatiquement.

La figure ci-après présente la configuration du composant **tMap**.



Après cette configuration, nous passons à l'exécution du workflow, on extrait les données, on les transforme et on les charge dans une base de données MySQL.

Ces images ci-après montrent la réussite de l'exécution du workflow et la présence des données dans la base de données.



A travers cette image, nous constatons la réussite de l'opération, vérifions dans la table créée si les données sont bien présentes.

Server: MySQL:3306 » Database: bd_issue » Table: fact_issue

Showing rows 0 - 24 (1000 total. Query took 0.0021 seconds.)

SELECT * FROM `fact\_issue`

Number of rows: 25 Filter rows: Search this table Sort by key: None

	id	Issue_Type	Summary	Assignee	Priority	Status	Resolution
☐	1	New Feature	Add execute (Remote) SSH commands step	Triage	Unknown	Open	UNRESOLVED
☐	2	Improvement	Update Version number in Splash Screen / Help About...	Grover	Blocker	In Progress	UNRESOLVED
☐	3	Bug	Need to fix incorrect labels on SSH2 step	Oscar	Severe	Open	UNRESOLVED
☐	4	Bug	Document default users for Pentaho Data Integratio...	Triage	Unknown	Open	UNRESOLVED
☐	5	Bug	Web Services Lookup step does not support basic au...	Big Bird	Severe	Ready For Test	UNRESOLVED
☐	6	Bug	Spoon.bat error: GOTO not expected	Big Bird	Blocker	Open	UNRESOLVED
☐	7	Bug	Error executing job entries in parallel from repos...	Triage	Unknown	Open	UNRESOLVED
☐	8	Improvement	Provide ability to limit the number of rows on the...	Triage	Unknown	Open	UNRESOLVED
☐	9	Bug	Unable to close openFile file with Excel Output	Count von Count	Blocker	Open	UNRESOLVED
☐	10	Improvement	The HTTP Job entry dialog has the HTTP headers har...	Oscar	Severe	Open	UNRESOLVED
☐	11	Bug	Mondrian Input doesn't check for Integer datatype	Triage	Unknown	Resolved	Fixed
☐	12	Bug	"CLONE -Checking ""Load all data from table"" fail...	Zoe	None	Open	UNRESOLVED

A partir de cette image, on peut bien constater que la table est remplie et que les données y sont présentes.

Configuration de JasperReport

Pour cette partie nous utiliser JasperReport pour l'analyse des données multidimensionnelles. Nous allons travailler avec MySQL, base de données que nous avons utilisé dès le départ.

Commençons d'abord comme indiqué, à ajouter une nouvelle source de données dans le répertoire Analysis Data Sources.

La figure ci-dessous illustre la réussite de cette opération.

TIBCO JasperSoft® Library View Manage jasperadmin User Log Out

New Data Source

Host (required): localhost

Port (required): 3306

Database (required): bd_issue

URL (required): jdbc:mysql://localhost:3306/bd_issue?tinyIntIsBit=false

Hint: jdbc:postgresql://localhost:5432/mydb

User Name: root

Password:

Time Zone: Africa/Casablanca - Western European Time

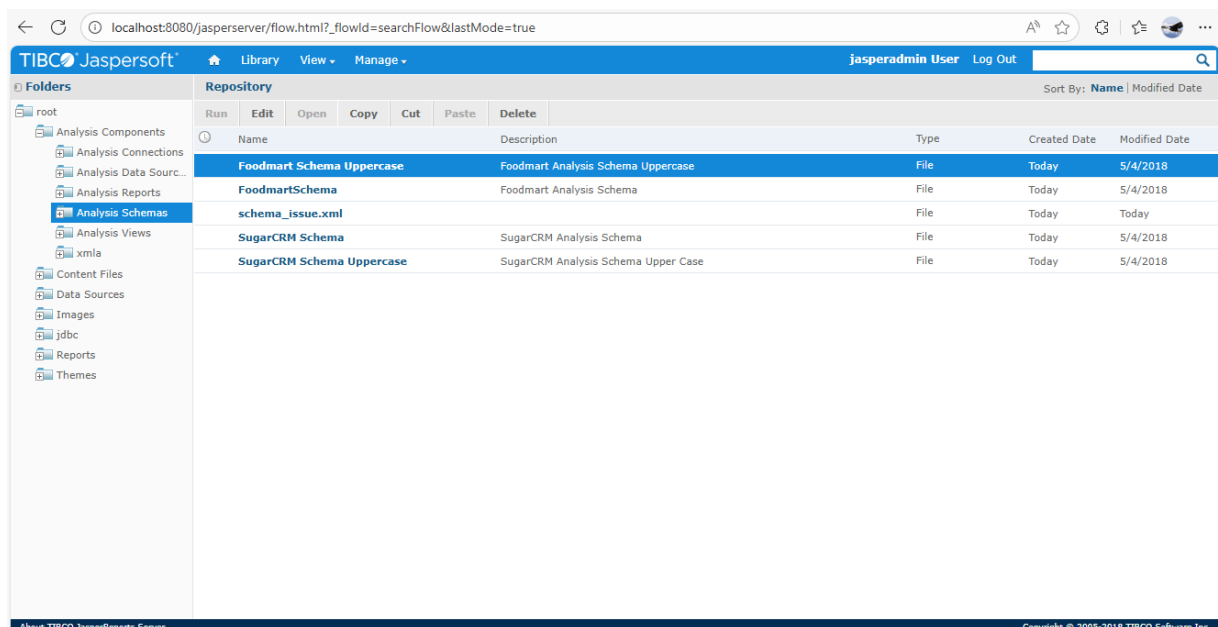
Hint: Do not change the time zone setting unless you know the database timestamp data is incorrect.

Test Connection Connection passed

Save Cancel

About TIBCO JasperReports Server Copyright © 2005-2018 TIBCO Software Inc.

Après l'ajout de la source de données, il faut bien sur, ajouté le schéma OLAP qui représente la structure du cube OLAP. Donc nous allons ajouter cela.



Nous allons faire une connexion à la base de données bd_issue, en définissant une connexion de type Mondrian.

The screenshot shows the 'Set Connection Type and Properties' form. The 'Connection Type' is set to 'Mondrian'. The form includes fields for Name, Resource ID, and Description, all of which are filled with 'bd_issue'. There is also a field for 'Select folder from repository to save' with the value '/analysis/connections'. The form has 'Previous', 'Next', and 'Cancel' buttons at the bottom.

The screenshot shows the TIBCO JasperReports Server web interface. The left sidebar displays a tree view of folders, including 'Analysis Data Sources'. The main area shows a table of the repository contents. The table has columns for Name, Description, Type, Created Date, and Modified Date. The data is as follows:

Name	Description	Type	Created Date	Modified Date
bd_issue		JDBC Data Source	October 20	October 20
Foodmart Data Source	Foodmart Data Source	JDBC Data Source	October 20	5/4/2018
Foodmart Data Source JNDI	Foodmart Data Source JNDI	JNDI Data Source	October 20	5/4/2018
SugarCRM Data Source	SugarCRM Data Source	JDBC Data Source	October 20	5/4/2018
SugarCRM Data Source JNDI	SugarCRM Data Source JNDI	JNDI Data Source	October 20	5/4/2018

Une fois cela effectué, nous créons une vue, il faut noter que la vue représente une requête particulière. Nous allons exécuter la requête MDX suivante :

SELECT Type.Children ON COLUMNS, Assignee.Children ON ROWS FROM Issue

L'exécution de cette requête nous donne ceci : (voir l'image ci-dessous)

Dimensions		Type			
Assignee	Assignee	Bug	Improvement	New Feature	Task
All Assignees	no nice error.	1			
	set permission	1			
	Abby Cadabby	5	1		1
	Bert	17			
	Big Bird	327	22	1	2
	Cookie Monster	79	8		
	Count von Count	42	5	2	
	Elmo	11	3	11	
	Ernie	19			
	Grover	12	2		
	Oscar	25	3		1
	Rosita	31	4	1	
	Telly Monster	14	1		
	Triage	51	11	10	3
	Unassigned	183	51	24	

Création des nouvelles vues pour chaque travail :

- Nombre de statuts par type de problèmes

La requête MDX à utiliser est donnée par :

SELECT {[Measures].[Issue Count]} ON COLUMNS, NON EMPTY CrossJoin([Type].[Type].Members,[Status].[Status].Members) ON ROWS FROM [Issue]

Define the Query

Select a language and enter the query

Query Language: MDX

Query String:

```
SELECT
{[Measures].[Issue Count]} ON COLUMNS,
NON EMPTY
CrossJoin(
[Type].[Type].Members,
[Status].[Status].Members) ON ROWS FROM [Issue]
```

Previous Submit Cancel

L'image suivante représente la vue associée à cette requête.

- Montrer le nombre de problèmes non résolus (UNRESOLVED) par employé

Define the Query
Select a language and enter the query

Query Language: **MDX**

Query String:

```
SELECT  
  {[Measures].[Issue Count]} ON COLUMNS,  
  {[Assignee].[Assignee].Members} ON ROWS  
FROM [Issue]  
WHERE ([Resolution].[Resolution].&{UNRESOLVED})
```

Previous Submit Cancel

About TIBCO JasperReports Server Copyright © 2005-2018 TIBCO Software Inc.

OLAP View

nombre_de_probleme_non_resolus_par_employé

Dimensions	Measures
Assignee	Issue Count
All Assignees	no nice error.
	set permission
Abby Cadabby	5
Bert	9
Big Bird	30
Cookie Monster	4
Count von Count	27
Elmo	12
Ernie	5
Grover	7
Oscar	14
Rosita	11
Telly Monster	4
Triage	10
Unassigned	245

About TIBCO JasperReports Server Copyright © 2005-2018 TIBCO Software Inc.

- Montrons le nombre d'enregistrements dans la table par statut

```
SELECT {[Measures].[Issue Count]} ON COLUMNS, {[Status].[Status].Members} ON  
ROWS FROM [Issue]
```

Dans la page suivant nous montrerons la vue associée à cette requête .

TIBCO Jaspersoft™ Library View Manage

jasperadmin User Log Out

OLAP View

nombre total denregistrements selon le statut

Dimensions		Measures
Status	Status	Issue Count
All Status	Blocker	2
	Closed	521
	In Progress	3
	Open	339
	Ready For Test	43
	Reopened	6
	Resolved	86

Filter:

Back to OLAP views
Back to Repository Browser

Copyright © 2005-2018 TIBCO Software Inc.

- Montrons le pourcentage pour chacun des statuts

WITH

MEMBER [Measures].[Pourcentage] AS

([Measures].[Issue Count]) / ([Measures].[Issue Count], [Status].[All Status])

* 100, FORMAT_STRING = "Percent"

SELECT

{[Measures].[Issue Count], [Measures].[Pourcentage]} ON COLUMNS,

[Status].[Status].Members ON ROWS

FROM [Issue]

TIBCO Jaspersoft™ Library View Manage

jasperadmin User Log Out

OLAP View

Montrons le pourcentage pour chacun des statuts

Dimensions		Measures	
Status	Status	Issue Count	Pourcentage
All Status	Blocker	2	20.00%
	Closed	521	5210.00%
	In Progress	3	30.00%
	Open	339	3390.00%
	Ready For Test	43	430.00%
	Reopened	6	60.00%
	Resolved	86	860.00%

Filter:

Back to OLAP views
Back to Repository Browser

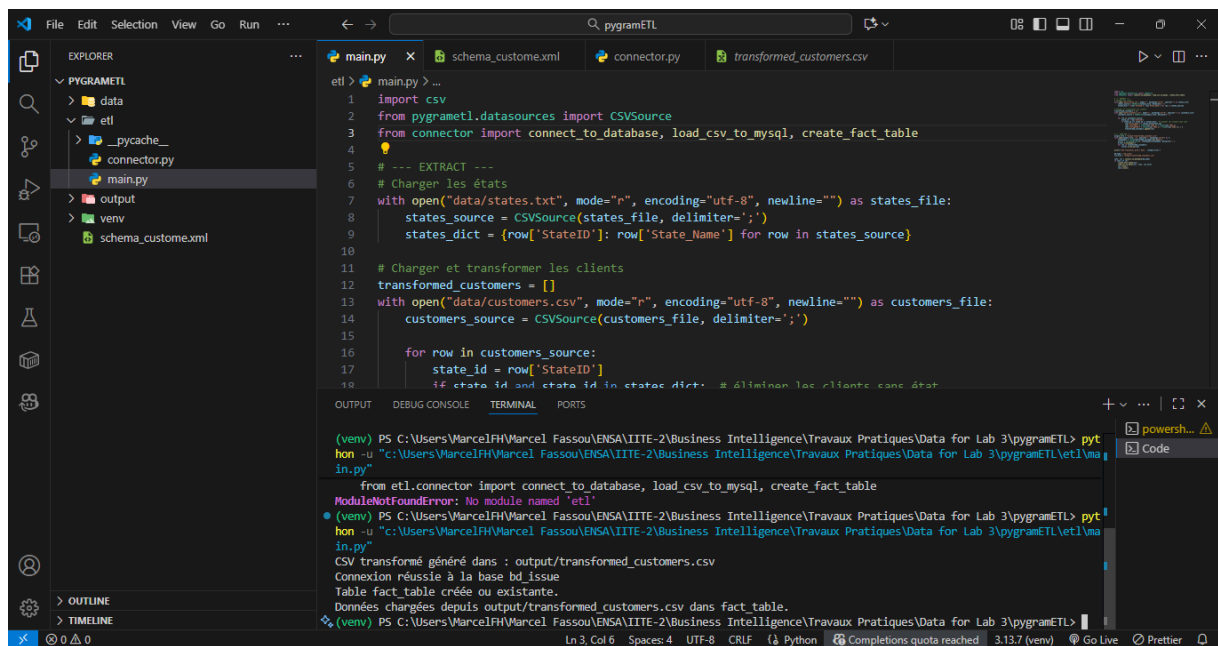
Copyright © 2005-2018 TIBCO Software Inc.

Utilisation du Framework PygramETL pour charger deux fichiers

PygramETL est une **bibliothèque Python open source** conçue pour faciliter la **mise en œuvre de processus ETL (Extract, Transform, Load)**. Elle permet d'extraire des données depuis différentes sources, de les transformer selon des règles spécifiques, puis de les charger dans une base de données cible, le tout de manière **structurée et automatisée**.

Dans cette partie, nous allons utiliser PygramETL pour charger deux fichiers `clients.csv` et `etats.txt` vers une table **fact_customers** tout en respectant différentes consignes données.

Cette image illustre le plan de travail et données générées :



```
etl > main.py > ...
1 import csv
2 from pygrametl.datasources import CSVSource
3 from connector import connect_to_database, load_csv_to_mysql, create_fact_table
4
5 # --- EXTRACT ---
6 # Charger les états
7 with open("data/etats.txt", mode="r", encoding="utf-8", newline="") as states_file:
8     states_source = CSVSource(states_file, delimiter=',')
9     states_dict = {row['StateID']: row['State Name'] for row in states_source}
10
11 # Charger et transformer les clients
12 transformed_customers = []
13 with open("data/customers.csv", mode="r", encoding="utf-8", newline="") as customers_file:
14     customers_source = CSVSource(customers_file, delimiter=',')
15
16     for row in customers_source:
17         state_id = row['StateID']
18         if state_id and state_id in states_dict: # éliminer les clients sans état
```

```
(venv) PS C:\Users\MarcelFH\Marcel Fassou\ENSA\IITE-2\Business Intelligence\Travaux Pratiques\Data for Lab 3\pygramETL> py
hon -u "c:\Users\MarcelFH\Marcel Fassou\ENSA\IITE-2\Business Intelligence\Travaux Pratiques\Data for Lab 3\pygramETL\etl\ma
in.py"
from etl.connector import connect_to_database, load_csv_to_mysql, create_fact_table
ModuleNotFoundError: No module named "etl"
(venv) PS C:\Users\MarcelFH\Marcel Fassou\ENSA\IITE-2\Business Intelligence\Travaux Pratiques\Data for Lab 3\pygramETL> py
hon -u "c:\Users\MarcelFH\Marcel Fassou\ENSA\IITE-2\Business Intelligence\Travaux Pratiques\Data for Lab 3\pygramETL\etl\ma
in.py"
CSV transformé généré dans : output\transformed_customers.csv
Connexion réussie à la base bd issue
Table fact table créée ou existante.
Données chargées depuis output\transformed_customers.csv dans fact table.
```

Les données générées sont :

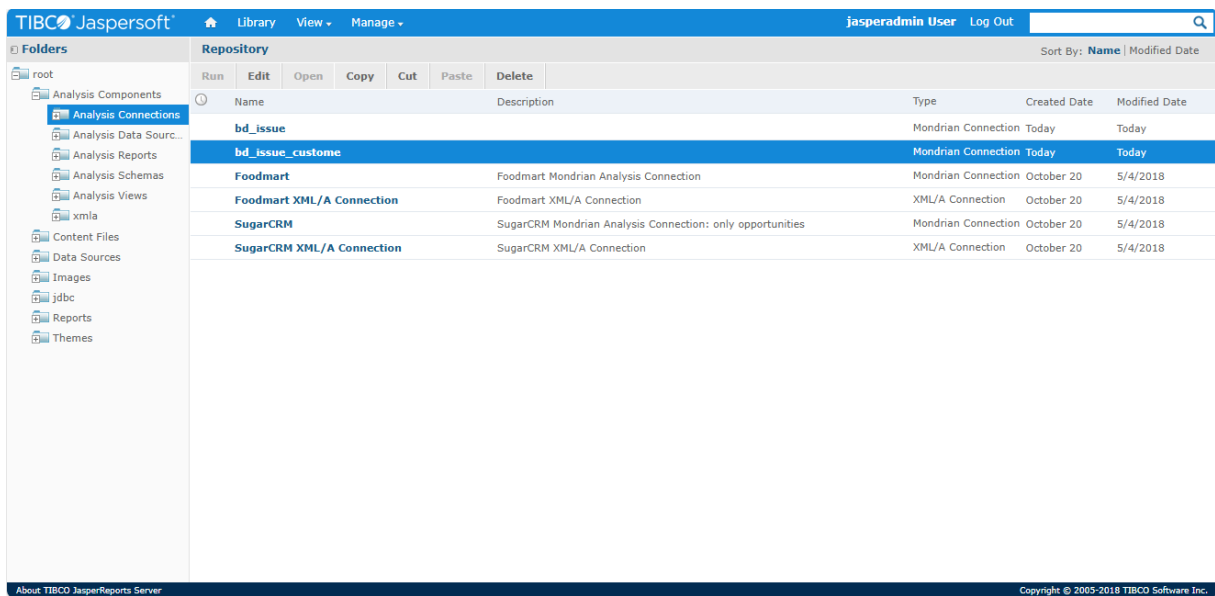
The image shows a PyCharm IDE window with a CSV file named 'transformed_customers.csv' open. The file contains 33 rows of data, each representing a customer record. The columns are: Customer Id, CustomerName, Customer_adresse, StateID, id2, tempsInsc, SUM1, SUM2, State_Name, TotalSUM, and AverageSUM. The data is sorted by Customer Id. The IDE interface includes a sidebar on the left with 'EXPLORER', 'PYGRAMETL', and 'output' views. The 'output' view shows the file path 'output > transformed_customers.csv > data'. The bottom status bar shows 'Col 11: AverageSUM', 'Ln 26 Col 108', 'Spaces 4', 'UTF-8', 'CR/LF', 'C/S (semicolons)', 'Comments quota resched', 'Go Live', and 'Prettier'.

Le code source se trouve en lien ou en fichier zip au mail qui a été envoyé.

[illegible]

Nous allons ajouter le schéma général, puisque nous possédons déjà une connexion sur le

TIBCO® Jaspersoft™ Library View Manage JasperAdmin User Log Out



Maintenant qu'une nouvelle connexion est créée nous allons présenter quelques vues OLAP

1. Chiffre total par état pour voir quels états génèrent le plus de valeur

```
SELECT
    {[Measures].[TotalSUM]} ON COLUMNS,
    [State].[State_Name].Members ON ROWS
FROM [CustomerFact]
```

State	State_Name	TotalSUM
All States		12,289,446
	Alabama	9,381,071
	Alaska	10,094,377
	Arizona	8,878,589
	Arkansas	10,402,424
	California	9,735,899
	Colorado	11,619,494
	Connecticut	12,158,328
	Delaware	9,472,414
	Florida	11,433,579
	Georgia	9,791,922
	Hawaii	9,809,732
	Idaho	10,136,576
	Illinois	7,913,380
	Indiana	10,092,137

2. SUM1 et SUM2 par Client et État

```
SELECT
    {[Measures].[SUM1], [Measures].[SUM2]} ON COLUMNS,
    CROSSJOIN([Customer].[CustomerName].Members, [State].[State_Name].Members)
ON ROWS
FROM [CustomerFact]
```

TIBCO Jaspersoft Library View Manage jasperadmin User Log Out

OLAP View

SUM1 et SUM2 par Client et État

Dimensions				Measures	
Customer	CustomerName	State	State_Name	SUM1	SUM2
All Customers	AA Advertising	All States			
			Alabama		
			Alaska	8,388	46,090
			Arizona		
			Arkansas	8,800	3,701
			California		
			Colorado		
			Connecticut	74,305	162,746
			Delaware	151,969	73,613
			Florida	82,688	21,324
			Georgia	49,201	24,680
			Hawaii	94,588	135,280
			Idaho	126,366	120,096
			Illinois	75,136	13,601
			Indiana		

Copyright © 2005-2018 TIBCO Software Inc.

Conclusion :

En conclusion, ce TP a permis de mettre en œuvre un processus complet d'**ETL**, en extrayant les données depuis plusieurs sources, en les transformant (calcul de TotalSUM et AverageSUM) et en les chargeant dans la table de faits fact_customer de MySQL. Nous avons ensuite construit un **cube OLAP Mondrian**, avec des dimensions pertinentes (Customer, State, DateInscription, id2) et des mesures clés, pour faciliter l'analyse multidimensionnelle. Des requêtes **MDX** ont illustré différents axes de décision, comme l'analyse des totaux par état, des moyennes par date d'inscription ou encore la répartition des mesures par client et état. L'intégration avec **JasperReport** a permis de visualiser ces analyses sous forme de rapports clairs et interactifs. Ce TP a ainsi démontré comment combiner **intégration de données, modélisation OLAP et reporting** pour obtenir des informations exploitables. En résumé, il fournit une base solide pour la **prise de décision stratégique** à partir des données clients.