

Université Gaston Berger de Saint-Louis

UFR Sciences Appliquées et Technologie

Section Informatique

MADSI 2019

Entrepôt de données

TP 2 : OLAP, MDX, Analyse de données

1 Mondrian

Mondrian¹ est un serveur OLAP qui permet aux utilisateurs d'analyser de large quantité de données en temps réel. Les utilisateurs explorent les données métier en exécutant des requêtes complexes sur des bases volumineuses.

Le serveur JasperReport offre des fonctionnalités d'analyse de données en intégrant l'engin Mondrian. Il est possible ensuite d'écrire et d'exécuter des requêtes MDX pour accéder au cube OLAP, dont le schéma est décrit dans un fichier XML.

1.1 Schéma du cube OLAP

Un schéma représente une base de données multi-dimensionnelle. Il contient un modèle logique, qui consiste en des cubes, des hiérarchies et des membres, et le mapping de ce modèle en un modèle physique.

- Le modèle logique consiste en des structures permettant d'écrire des requêtes en utilisant le langage MDX: les cubes, dimensions, hiérarchies, niveaux et membres.

- Le modèle physique est la source de données présentée par le modèle logique.

C'est typiquement un schéma en étoile, représenté par un ensemble de tables dans une base de données relationnelles.

Dans Mondrian, les schémas sont représentés par des **fichiers XML**. Pour l'instant, ce fichier doit être écrit à la main. Nous présentons dans ce qui suit les éléments importants dans la structure d'un schéma Mondrian:

1.1.1 Cube

Un cube est une collection de mesures et dimensions. Il fait référence à une table des faits, ainsi que l'ensemble des mesures et des dimensions qui lui sont liées. La table des faits contient les colonnes à partir desquelles les mesures sont faites, ainsi que des références aux tables contenant les dimensions.

1.1.2 Mesure

Un cube peut définir une ou plusieurs mesures, chaque mesure ayant un nom, une colonne dans la table des faits, et un *aggregator*. Un *agregateur* représente le traitement à faire sur les données, par exemple *sum*, *count*, *min*, *max*, *avg*...

1.1.3 Membre

Un membre est un point dans une dimension, déterminé par un ensemble particulier de valeurs. Par exemple, 'Sor', 'Saint-Louis' et 'Sénégal' représentent trois membres de la hiérarchie *adresse*.

1.1.4 Hiérarchie

Une hiérarchie est un ensemble de membres organisés en une structure pour une analyse convenable. Par exemple, la hiérarchie *adresse* contient le numéro, la rue, la ville et le pays. La hiérarchie permet de construire des sous-totaux: le sous-total d'un pays représente la somme des sous-totaux de toutes les villes de ce pays.

1.1.5 Niveau

Un niveau représente la collection des membres qui ont la même distance par rapport à la racine de la hiérarchie.

1.1.6 Dimension

Une dimension est une collection de hiérarchies pour un même attribut de la table des faits.

1.2 Les requêtes d'exploration MDX

MDX (Multi-Dimensional Expressions) est le langage principal pour les requêtes, implémenté par Mondrian. Il ressemble un peu au langage SQL, mais sa structure est différente, vu qu'il traite des données dimensionnelles, pas opérationnelles.

Voici un exemple de requête MDX :

```
SELECT {[Measures].[Unit Sales], [Measures].[Store Sales]} ON COLUMNS,  
{[Product].members} ON ROWS  
FROM [Sales]  
WHERE [Time].[1997].[Q2]
```

2 Analyse

Dans ce TP, nous allons faire appel à l'engin Mondrian installé dans Jasper Report. Les outils et fichiers dont nous aurons besoin sont les suivants :

- **JasperReport Server 5.1.0** : Serveur de reporting qui utilise l'engin Mondrian pour l'analyse des données.
- **Talend TOS 5.2.0** : Pour la transformation des données (phase ETL)
- **Bug_report.csv** : fichier csv contenant les données initiales à traiter.
- **Schema.xml** : représente le schéma OLAP de notre base cible.
- **Mysql-connector-java 5.1.26** : connecteur MySQL pour JasperReport, plusieurs autres pilotes comme Postgresql, Oracle...

2.1 Etape 1 : Création de la base de données

La première étape consiste en la copie des données fournies dans le fichier délimité, dans une base de données de votre choix.

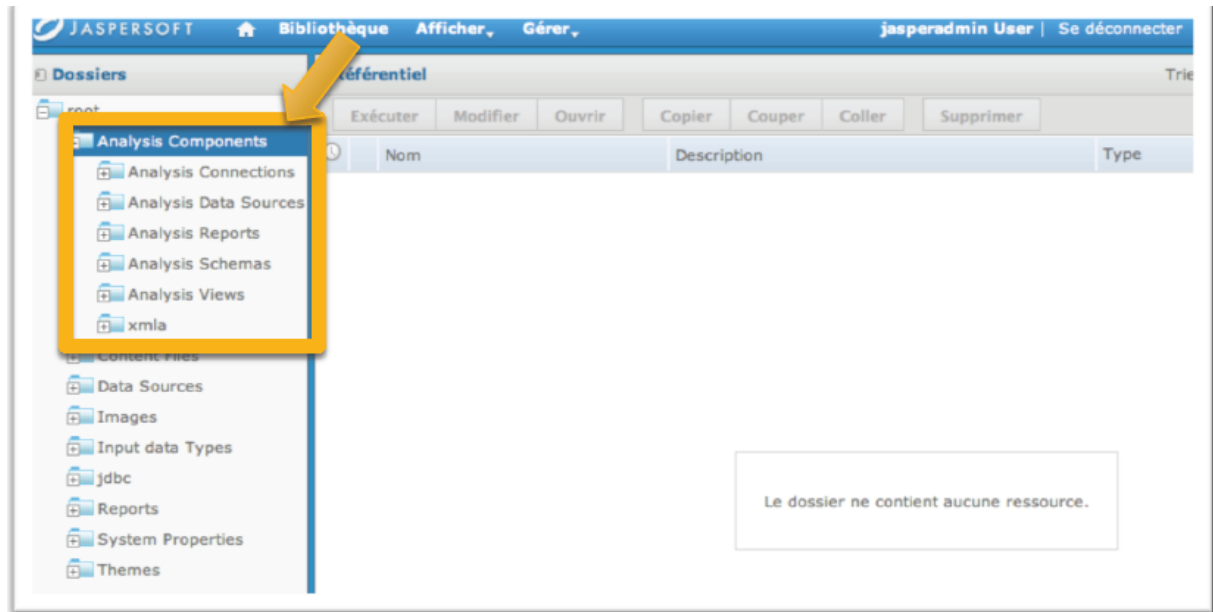
Activité 1 : Utiliser Talend Data Integration pour copier les données du fichier délimité *bug_report.csv* dans une base de données appelée *issue*, contenant une table appelée *fact_issue*. Cette table représentera la table des faits de votre cube. Nous donnons ci-joint la requête SQL permettant de créer la structure de cette table.

```
create table fact_issue(  
id tinyint primary key AUTO_INCREMENT,
```

Issue_Type varchar(11),
Summary varchar(255),
Assignee varchar(60),
Priority varchar(8),
Status varchar(14),
Resolution varchar(16)

),

2.2 Etape 2 : Configuration de JasperReport



2.2.1 Source de données

Pour se connecter à la base de données *issue* créée précédemment, on doit créer une nouvelle source de données.

Activité 2.

Ajouter une nouvelle source de données dans le répertoire *Analysis Data Sources*. Faire référence ici à la base *issue* créée précédemment. (On vous fournit le pilote pour MySQL dans le cas où votre SGBD est MySQL).

2.2.2 Création d'un schéma OLAP

Le schéma OLAP représente la structure du cube OLAP. Pour notre exemple, le schéma est représenté dans le fichier *schema.xml*, dont la structure est la suivante:

```

<Schema name="IssueSchema">
  <Cube name="Issue">
    <Table name="fact_issue"/>
    <Dimension name="Type">
      <Hierarchy hasAll="true" allMemberName="All Types">
        <Level name="Type" column="issue_type" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Dimension name="Assignee">
      <Hierarchy hasAll="true" allMemberName="All Assignees">
        <Level name="Assignee" column="assignee" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Dimension name="Priority">
      <Hierarchy hasAll="true" allMemberName="All Priorities">
        <Level name="Priority" column="priority" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Dimension name="Status">
      <Hierarchy hasAll="true" allMemberName="All Status">
        <Level name="Status" column="status" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Dimension name="Resolution">
      <Hierarchy hasAll="true" allMemberName="All Resolution">
        <Level name="Resolution" column="resolution" uniqueMembers="true"/>
      </Hierarchy>
    </Dimension>
    <Measure name="Issue Count" column="id" aggregator="count" formatString="Standard"/>
  </Cube>
</Schema>

```

Nom de la table des faits

Dimension Type

Mesure du nombre des id

Activité 3.

Ajouter le schéma OLAP *schema.xml* dans le répertoire *Analysis Schemas*.

2.2.3 Connexion à la base *issue*

Pour parcourir la base *issue*, nous devons définir une nouvelle connexion de type Mondrian, et lui attribuer le schéma OLAP ajouté précédemment.

Activité 4.

Ajouter une nouvelle connexion de type Mondrian à cette source de données dans le répertoire *Analysis Connections*. Lui associer le schéma OLAP et la source de données définies précédemment.

2.2.4 Création de la vue

Une vue représente une requête particulière, comme demandée par le décideur. A partir de cette vue, plusieurs types de sorties peuvent s'afficher, comme des tableaux, des graphes...

Pour notre exemple, nous allons créer une vue permettant d'afficher le nombre de problèmes traités par un employé, classés par type de problème. Pour cela, la requête MDX suivante sera exécutée:

```

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

```


Activité 5.

Ajouter une nouvelle vue dans le répertoire *Analysis View*, exécutant la requête MDX précédente. Visualiser les différents affichages en sortie. Exporter les différentes sorties dans un format PDF, et les enregistrer dans un répertoire qui porte votre nom, pour le donner à votre enseignante.

3. Projet qui fera office de note de CC (Un rapport détaillé sera rendu)

En considérant les mêmes données, créer les vues suivantes :




- Montrer le nombre de statuts par type de problèmes.

Dimensions 		Status					
		All Status					
(Tout)	Type	Closed	In Progress	Open	Ready For Test	Reopened	Resolved
All Types	Bug	429	2	224	38	5	52
	Improvement	27	1	56	2	1	11
	New Feature	9		27			9
	Task	2		2	2		1





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

Dimensions 		Resolution
		All Resolution
(Tout)	Assignee	UNRESOLVED
All Assignees	Abby Cadabby	5
	Bert	9
	Big Bird	29
	Cookie Monster	4
	Count von Count	25
	Elmo	11
	Ernie	5
	Grover	7
	Oscar	13
	Rosita	11
	Telly Monster	4
	Triage	10
	Unassigned	219
	Zoe	8

- Montrer le nombre d'enregistrements dans la table par statut.

Dimensions 		Mesures
(Tout)	Status	 Status Count
 All Status		900
All Status	Closed	467
	In Progress	3
	Open	309
	Ready For Test	42
	Reopened	6
	Resolved	73

- Montrer le pourcentage pour chacun des statuts.

Dimensions 		Mesures	
(Tout)	Status	 Percentage	 Status Count
 All Status		Infinity	900
All Status	Closed	0,519	467
	In Progress	0,003	3
	Open	0,343	309
	Ready For Test	0,047	42
	Reopened	0,007	6
	Resolved	0,081	73

Indication : Dans cette exercice on pensera à utiliser WITH MEMBER qui permet de définir des membres calculés. A.CurrentMember permet d'obtenir le membre courant. A.Parent permet d'avoir le parent de A (Dans le cas des hierarchies).

Exemple :

```
(with member [Measures].[Status Count] as '[Measures].[Issue Count]' member
[Measures].[Pourcentage] as '([Status].CurrentMember / [Status].CurrentMember.Parent)',
FORMAT_STRING = "0.00%"
select {[Measures].[Pourcentage], [Measures].[Status Count] } ON COLUMNS,
[Status].Children ON ROWS from [Issue]
```

