Interdepartmental Programme of Postgraduate Studies in Information Systems, University of Macedonia

MASTER THESIS

ANALYSIS AND PROCESS OF HIERARCHICAL SEMI-STRUCTURED DATA

George N. Tsilinikos

Submitted as required for the Master’s Degree in Information Systems

October 2018

Abstract

Online analytical processing, or OLAP, is an approach to answering multi-dimensional analytical (MDA) queries swiftly in computing. One of the uses of OLAP engines is to process large volumes of hierarchical data and allow business users to analyze large and complex amounts of data in real-time.

Such is Mondrian, which is an open source OLAP engine developed by Julian Hyde and adopted by Pentaho business intelligence software of Hitachi Vantara.

Modrian uses HSQL Database to store the hierarchical data which are structured in Data Cubes using Hierarchies and Dimensions.

The goal of the analysis is to gain a better understanding of Modrian OLAP Engine and identify case where it could be applied.

In order to use Mondrian engine as standalone, Xmondrian was created. It is the .WAR file that was used to deploy the engine which contained the Mondrian engine along with the Foodmart and Steelwheels embedded Data Cubes and datasets that had been used.

The results of the analysis indicate that even though OLAP Engines are mainly used in BI analysis, Mondrian Engine can be of many uses and provide benefits to other fields too.

Contents

1 INTRODUCTION **5**
2 BIBLIOGRAPHY **6**
2.1 OLAP ANALYSIS 6
2.2 OLAP SCHEMA 6
2.2.1 Hierarchies *7*
2.2.2 Dimensions *8*
2.2.3 Measures *8*
2.2.4 Operations *9*
2.2.4.1 Roll up 9
2.2.4.2 Drill Down 9
2.2.4.3 Aggregate 9
2.2.4.4 Spread 9
2.3 OLAP VS OLTP 10
2.4 TYPES OF OLAP SYSTEMS 12
2.4.1 Multidimensional OLAP (MOLAP) *12*
2.4.2 Relational OLAP (ROLAP) *13*
2.4.3 MOLAP vs ROLAP *14*
2.4.4 Hybrid OLAP (HOLAP) *15*
3 METHODOLOGY **17**
3.1 INTRODUCTION 17
3.2 OLAP ANALYSIS BIBLIOGRAPHY 17
3.3 TOMCAT JAVA SERVER 17
3.4 MONDRIAN SCHEMA WORKBENCH 18
3.5 HYPERSQL 19
3.6 MULTI-DIMENSIONAL EXPRESSIONS (MDX) 20
3.7 XAVIER VISUALIZER TOOL 21
4 DATA ANALYSIS & INTERPRETATION OF FINDINGS **23**
4.1 PIVOT TABLE 23
4.2 PIE CHART 24
4.3 BAR CHART 24
5 EXAMPLES OF PLATFORMS THAT USE OLAP ANALYSIS **26**
5.1 PENTAHO MONDRIAN – APHSD MULTIDIMENSIONAL DATABASE 26
5.2 OLAP IN RETAIL - ORACLE RPAS 36

6 CONCLUSIONS 40
7 IMPROVEMENT PROPOSAL 41
8 WEB PAGES VISITED 42
9 TABLE OF FIGURES 43

1 Introduction

The main goal of the thesis is a high-level business and technical analysis of the open source OLAP engine, the Mondrian OLAP Server. An additional goal was to gain deeper understanding of the OLAP analysis, it’s benefits and it’s potential uses apart from BI.

The embedded Data Cubes were used along with the example Foodmart & Steelwheels schemas and datasets that were provide in Xmondrian .WAR file.

2 Bibliography

In this section is described the technologies that already exist and was used during the analysis of Modrian OLAP Server.

2.1 OLAP Analysis

Online analytical processing, or OLAP, is an approach to answering multi-dimensional analytical (MDA) queries swiftly in computing. One of the uses of OLAP engines is to process large volumes of hierarchical data and allow business users to analyze large and complex amounts of data in real-time.

2.2 OLAP Schema

OLAP Schema is a collection of data cubes, measures and hierarchies that define a set of tools to serve a specific business need.

An OLAP Cube is a multi-dimensional array of data. It is used in OLAP to store data in more than one dimensions.

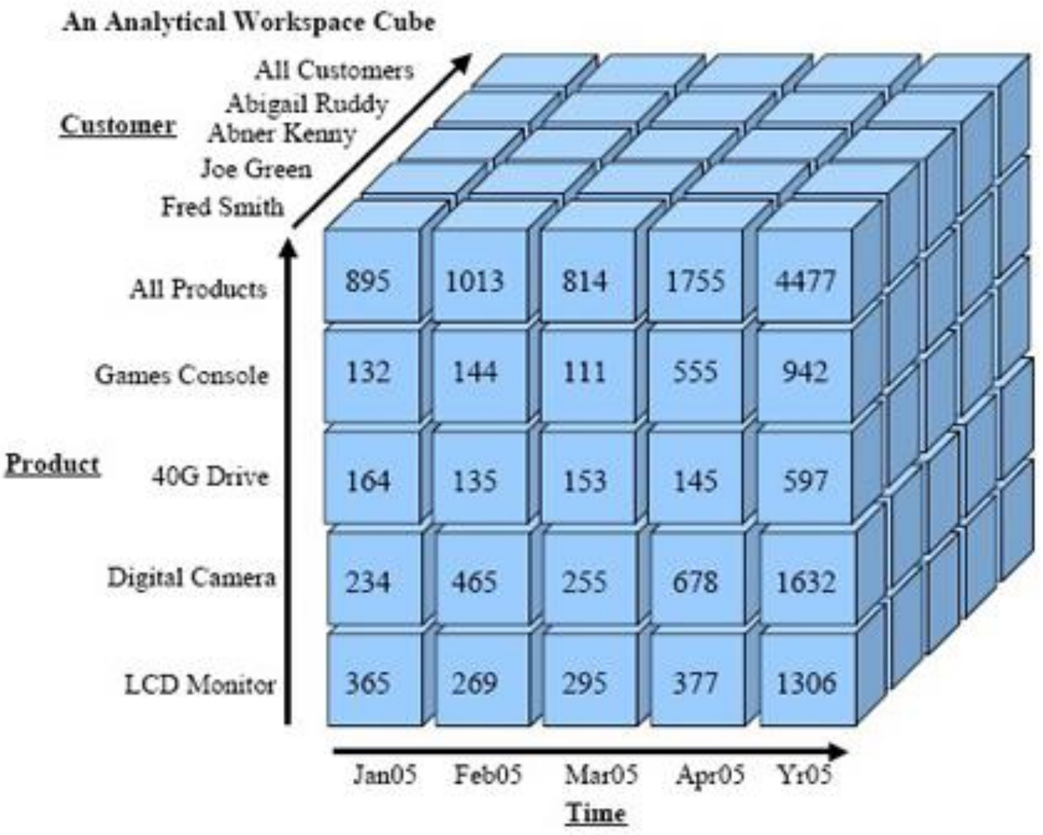


Figure 2-1 Example of an OLAP Cube

2.2.1 Hierarchies

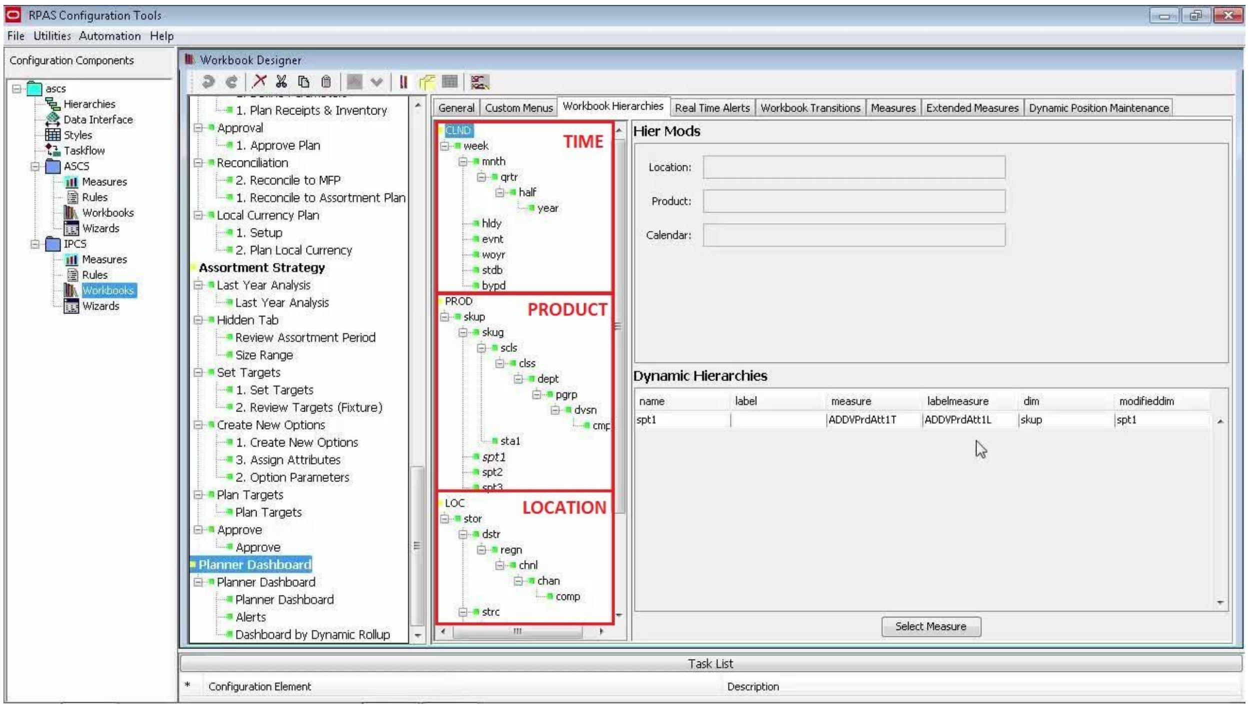
It a hierarchical structure that represents the parent-child relationship among the dimensions. Each dimension is consisted of the “summary” of it’s children.

By “position” of a dimension we are referring to a specific instance of this dimension. Thus, “January 2018” is a position of the “Month” dimension of TIME hierarchy.

TIME hierarchy is the most typical example of a hierarchy. It represents the time divisions of the Calendar and their aggregation path (e.g Day → Week → Month → Quarter → Year)

The most common lowest dimension is the “Day” dimension. “Week” dimension is consisted by Days the belong to that Week. For example, the days 3/9/2018-9/9/2018 aggregate to the first week of September 3/9/2018. (each week could be named by it’s first day). “Month” dimension is consisted by the weeks that belong to each month and so on.

The most common hierarchies in retail are TIME, PRODUCT and LOCATION. Product hierarchy is consisted by the items and all their rollups such as “departments” while Location hierarchy is consisted by the stores and all their rollups such as “districts”.



name	label	measure	labelmeasure	dim	modifieddim
spt1		ADDVPrdAtt1T	ADDVPrdAtt1L	skup	spt1

Figure 2-2 Example of Hierarchies in Oracle's RPAS Configuration Tools

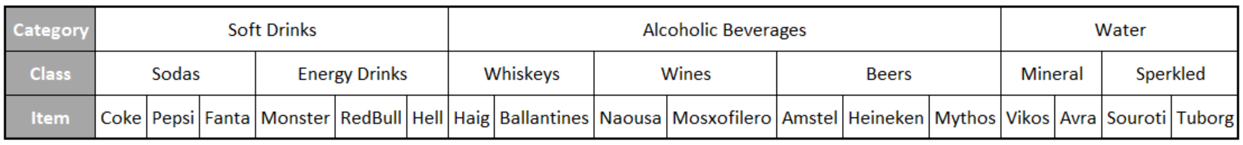
Source: http://learn.oracle.com/education/php/modules/cloud/ols1_player.php?streamId=1329&lold=22234&coId=105758

2.2.2 Dimensions

Each hierarchy is consisted of dimensions. Dimension is a level in which the data can be stored. The data can be represented at different levels (i.e. dimension) within a hierarchy by aggregating the values with the specified aggregation method (for more information please refer to section 2.2.4 Operations).

Dimension is the collection of all the positions that belong to a specific level of a hierarchy. For example, “Item” dimension in Product hierarchy is the collection of all items that have a unique ID. In retail they are often referred as SKU. An example of an item is the “Boutari Naousa Red Wine” which is a single bottle of wine. This item can be rolled up to a higher dimension such as “Wines” and subsequently to an even higher like “Alcoholic Bevarages”.

Higher dimension, is considered a bigger grouping of levels within a hierarchy.



Category	Soft Drinks						Alcoholic Beverages							Water			
Class	Sodas			Energy Drinks			Whiskeys		Wines		Beers			Mineral		Sperkled	
Item	Coke	Pepsi	Fanta	Monster	RedBull	Hell	Haig	Ballantines	Naousa	Moxsofilero	Amstel	Heineken	Mythos	Vikos	Avra	Souroti	Tuborg

Figure 2-3 Example of hierarchical relationship

2.2.3 Measures

Measures are multidimensional arrays in which the data can be stored. There can be multiple measures each one representing a business metric.Measures can be also considered as “entities” which define a specific measurement such as “Sales”.

Each measure stores the data at specific levels of hierarchies. For example, “Sales” measure can store the data at the Day dimension of the Time hierarchy, at the Item dimension of the Product hierarchy and at the Store dimension of the Location hierarchy. This means that this is the lowest levels in which the data will be stored. To move within a hierarchy, an aggregation method needs to be defined in order aggregate or spread the data to a higher or a lower dimension.

For example, let’s assuming that the “Sales” data is stored at item/store/**day.** In order to rollup to item/store/**week**, then if the aggregation method is “total” the data will be aggregated by summing the dales of each day within a week to that week. More details regarding the aggregation will be provided in the following section 2.2.4 Operations.

Measures can be represented by using one or more dimensions of the hierarchies.

- Sales per Item → *item,sales_value* (single dimension measure)
- Sales per Item per Store → *time,store,sales_value* (2-dimensional measure)
- Sales per Item per Store per Month → *item,store,month,sales_value* (3-dimensional measure)

- Sales per Item per Store per Month per Customer → *item,store,month,customer,sales_value* (4-dimensional measure etc.)

2.2.4 Operations

OLAP engines provide multiple tools to analyze and calculate the data and to move up or down in a hierarchy.

Such tools are described below:

2.2.4.1 Roll up

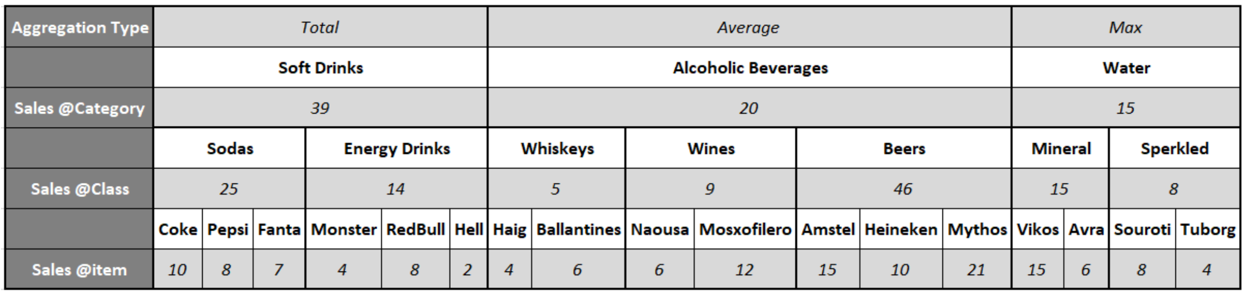
Rollup is an operation which allows the measures to be represented in a higher level within a hierarchy to achieve a higher level of granularity of the data.

2.2.4.2 Drill Down

On the other hand, when a lower level of granularity is required, the Drill-Down process is used to spread the values of the measure to lower level within the hierarchy. Then the data is represented in a more detailed form.

2.2.4.3 Aggregate

Aggregation is the method that defines the way that data will be rolled up to a higher dimension. Such ways are called “Aggregation Types” and the most common among them is the “total” and the “average”. When the aggregation type is “total” then the values of the lower positions of the hierarchy will be summed up to the higher dimension. Similarly, when the aggregation type is “average”, the value of the higher position will be calculated as the average of the values of the lower positions.

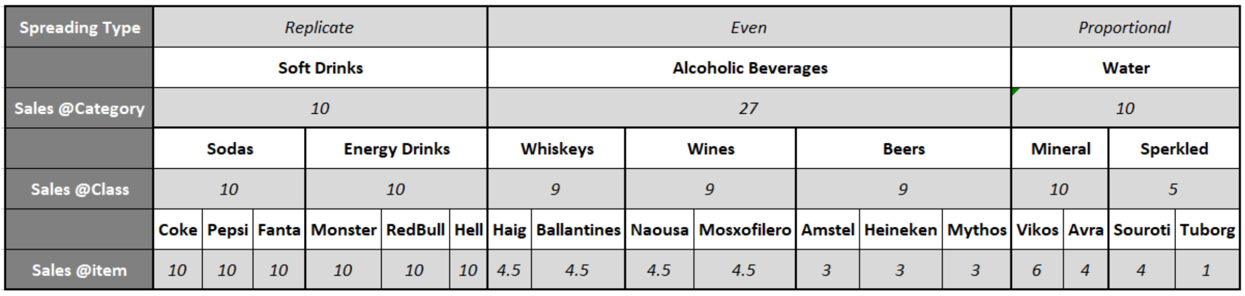


Aggregation Type	*Total*						*Average*							*Max*			
	Soft Drinks						**Alcoholic Beverages**							**Water**			
Sales @Category	*39*						*20*							*15*			
	Sodas			**Energy Drinks**			**Whiskeys**		**Wines**		**Beers**			**Mineral**		**Sperkled**	
Sales @Class	*25*			*14*			*5*		*9*		*46*			*15*		*8*	
	Coke	**Pepsi**	**Fanta**	**Monster**	**RedBull**	**Hell**	**Haig**	**Ballantines**	**Naousa**	**Mosxofilero**	**Amstel**	**Heineken**	**Mythos**	**Vikos**	**Avra**	**Souroti**	**Tuborg**
Sales @item	*10*	*8*	*7*	*4*	*8*	*2*	*4*	*6*	*6*	*12*	*15*	*10*	*21*	*15*	*6*	*8*	*4*

Figure 2-4 Example of the different aggregation types

2.2.4.4 Spread

On the contrary, spreading is the opposite. When the data is stored in a higher level and the spreading of the values to lower positions is required, then -based on the spreading method- the data will be distributed at the lower levels. The most common spreading types is “Replicate”, which replicates the value to all the lower positions, the “Even” which spreads the value evenly in all lower positions and “Proportional” which needs the measures to be initialized first. Then the value is spread among the lower position taking into account the previous value of each lower position and it's contribution to the higher value.



Spreading Type	*Replicate*						*Even*							*Proportional*			
	Soft Drinks						**Alcoholic Beverages**							**Water**			
Sales @Category	*10*						*27*							*10*			
	Sodas			**Energy Drinks**			**Whiskeys**		**Wines**		**Beers**			**Mineral**		**Sperkled**	
Sales @Class	*10*			*10*			*9*		*9*		*9*			*10*		*5*	
	Coke	**Pepsi**	**Fanta**	**Monster**	**RedBull**	**Hell**	**Haig**	**Ballantines**	**Naousa**	**Moxofilero**	**Amstel**	**Heineken**	**Mythos**	**Vikos**	**Avra**	**Souroti**	**Tuborg**
Sales @item	*10*	*10*	*10*	*10*	*10*	*10*	*4.5*	*4.5*	*4.5*	*4.5*	*3*	*3*	*3*	*6*	*4*	*4*	*1*

Figure 2-5 Example of different Spreading Types

2.3 OLAP vs OLTP

OLTP (On-line Transaction Processing) is the process where information systems store and manage all the transactional information, what data is entered in the databases and what is retrieved.

The main goal of OLTP is to provide a very fast query processing and to maintain the data integrity in multi-access environments. The transactions per second can be an indicative method to evaluate it's effectiveness.

On the other hand, in OLAP, the volume of the transactions is usually very low. OLAP is using the data the comes from the OLTP databases and it's queries are complex and involve aggregations. The historical data is aggregated and stored in Multi-dimensional schemas.

In the following table it is described the process of extracting the data from OLTP and loading it to OLAP:

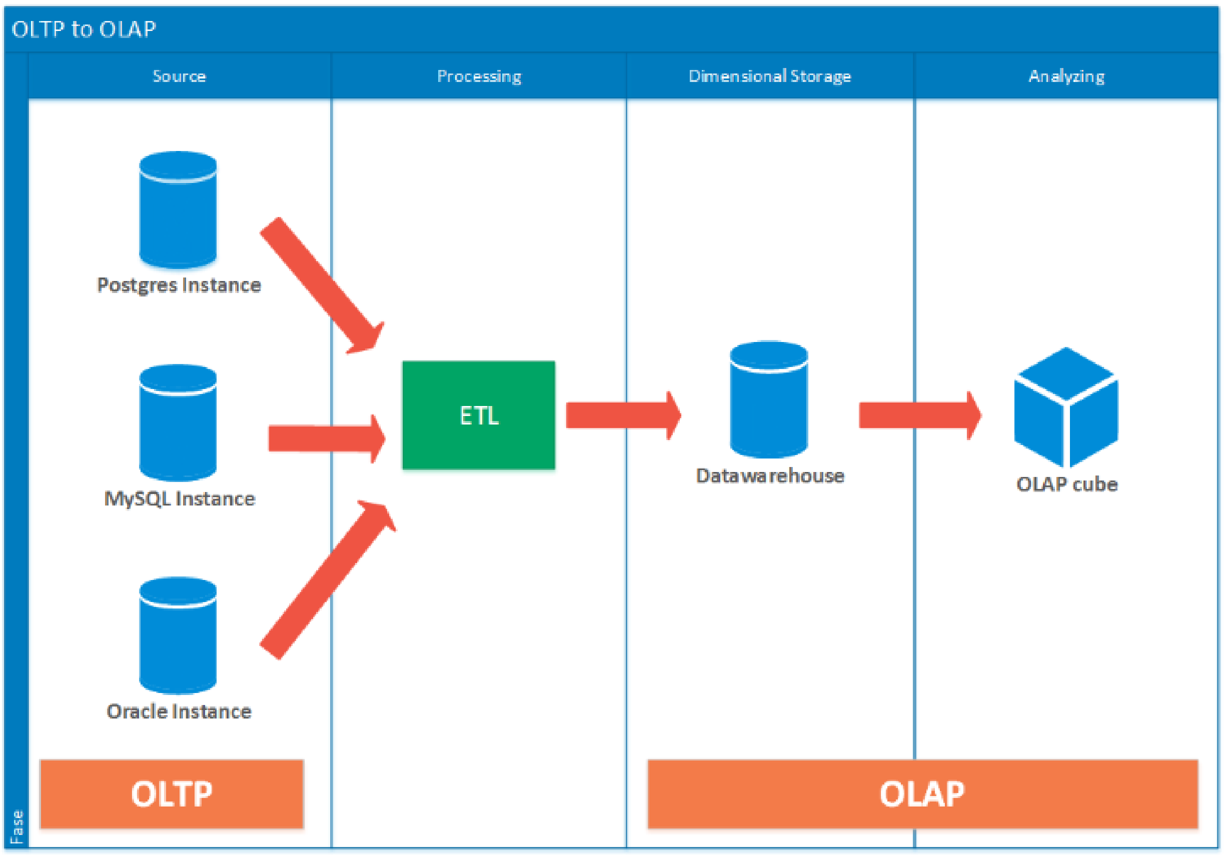
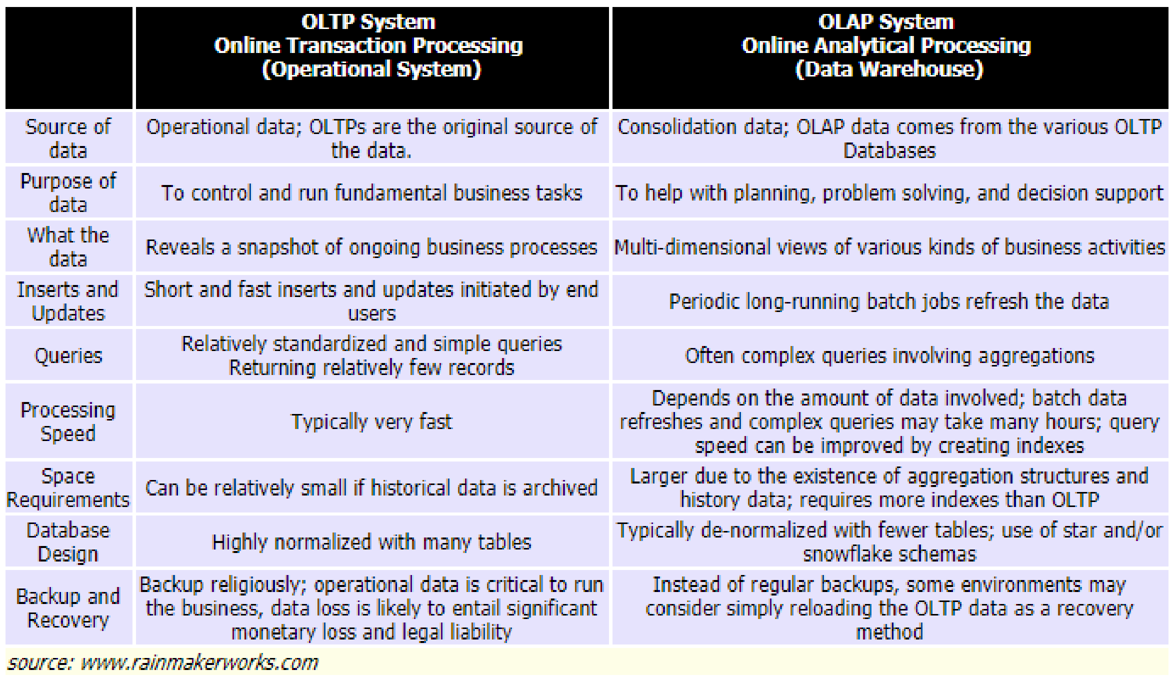


Figure 2-6 Data flow from OLTP to OLAP.

Source: https://blog.gabriela.io/2014/12/07/data-warehouse-experimenting/

The summary of the differences between OLTP and OLAP can be found in the table below:



	OLTP System Online Transaction Processing (Operational System)	OLAP System Online Analytical Processing (Data Warehouse)
Source of data	Operational data; OLTPs are the original source of the data.	Consolidation data; OLAP data comes from the various OLTP Databases
Purpose of data	To control and run fundamental business tasks	To help with planning, problem solving, and decision support
What the data	Reveals a snapshot of ongoing business processes	Multi-dimensional views of various kinds of business activities
Inserts and Updates	Short and fast inserts and updates initiated by end users	Periodic long-running batch jobs refresh the data
Queries	Relatively standardized and simple queries Returning relatively few records	Often complex queries involving aggregations
Processing Speed	Typically very fast	Depends on the amount of data involved; batch data refreshes and complex queries may take many hours; query speed can be improved by creating indexes
Space Requirements	Can be relatively small if historical data is archived	Larger due to the existence of aggregation structures and history data; requires more indexes than OLTP
Database Design	Highly normalized with many tables	Typically de-normalized with fewer tables; use of star and/or snowflake schemas
Backup and Recovery	Backup religiously; operational data is critical to run the business, data loss is likely to entail significant monetary loss and legal liability	Instead of regular backups, some environments may consider simply reloading the OLTP data as a recovery method

source: www.rainmakerworks.com

Figure 2-7 Differences between OLTP and OLAP. *Source: http://datawarehouse4u.info/OLTP-vs-OLAP.html*

2.4 Types of OLAP Systems

2.4.1 Multidimensional OLAP (MOLAP)

This the classic type of OLAP. It uses a multi-dimensional data model to analyze data. The data is stored in an optimized multi-dimensional array storage instead of a relational database.

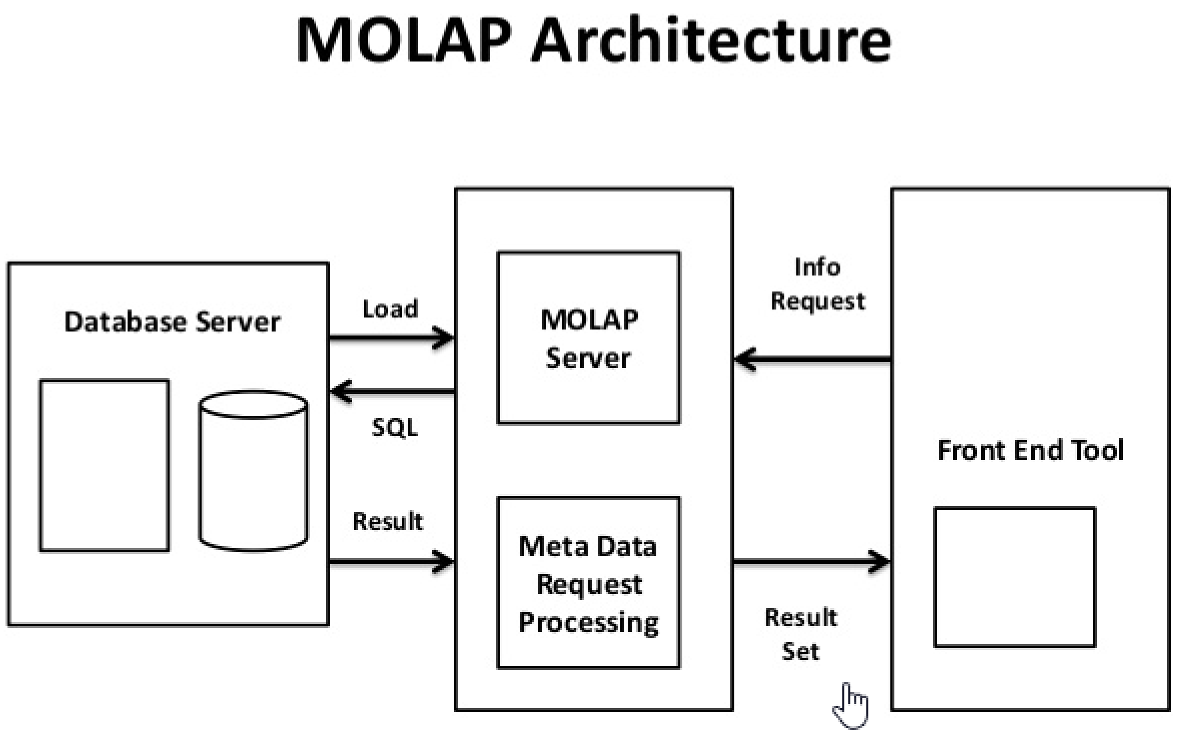


Figure 2-8 MOLAP Architecture

Source: https://www.researchgate.net/figure/MOLAP-architecture_fig19_319852408

The main advantages of MOLAP are:

- MOLAP Data Cubes are built for fast data retrieval and thus they allow “slicing and dicing” operations
- The performance of the queries is very fast due to the optimized storage, multidimensional indexing and caching
- The ability to quickly perform complex calculations because they have been pre-generated during creation of the cubes

The main disadvantage of MOLAP is:

- The data loading may take a significant amount of time when data volume is high

2.4.2 Relational OLAP (ROLAP)

Relational OLAP is the type of OLAP that performs dynamic multidimensional analysis of data stored in a relational database.

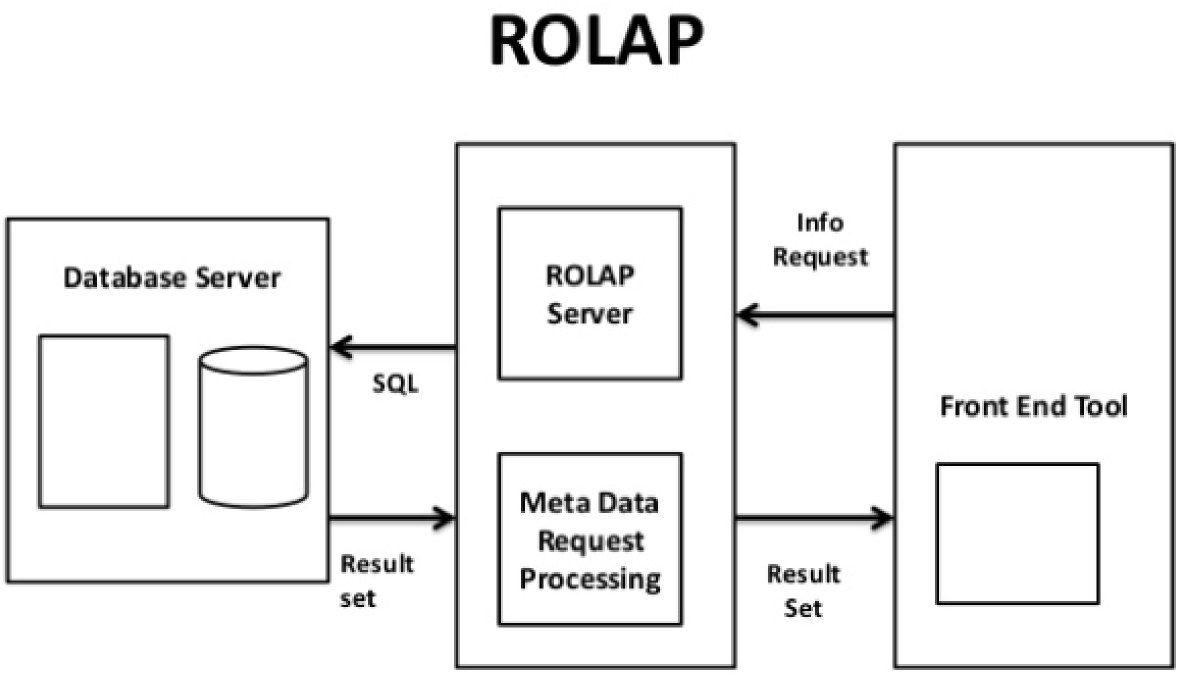


Figure 2-9 ROLAP Architecture

Source: https://www.researchgate.net/figure/ROLAP-architecture_fig20_319852408

The main advantages of ROLAP are:

- ROLAP is more scalable when it come to handle large data volumes, especially models with dimensions with very high cardinality
- The data are stored in a standard relational database and can be accessed by any SQL reporting tool

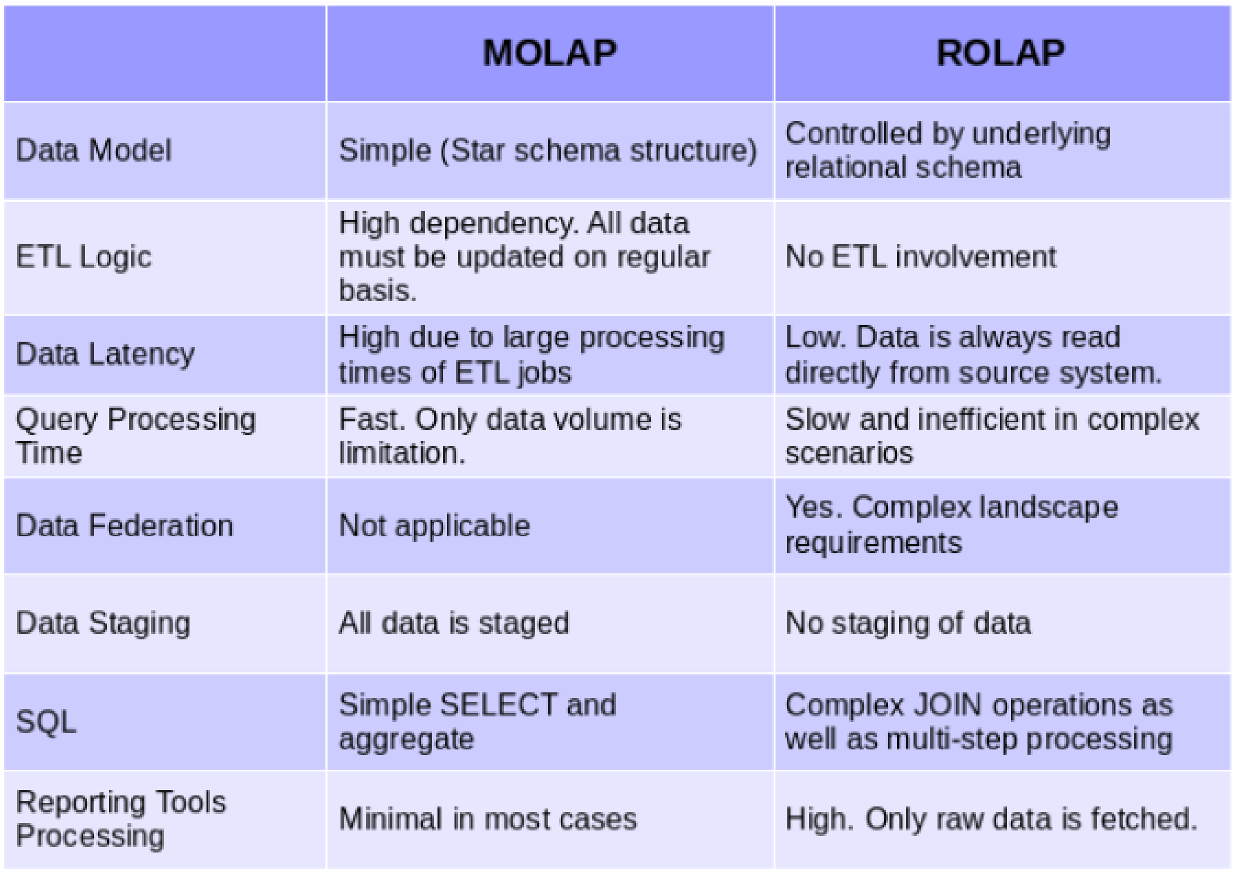
The main disadvantages of ROLAP are:

- The loading of aggregate tables must be managed by custom ETL code
- ROLAP tools are based on the SQL for the calculations and thus it is not suitable when it comes to heavy calculations that do not translate well to SQL

The main difference between MOLAP and ROLAP is that MOLAP requires that information first be processed before it is indexed directly into a multidimensional database. On the other hand, in ROLAP the information is entered directly into a relational database.

2.4.3 MOLAP vs ROLAP

In the following table are described the differences between MOLAP and ROLAP:



	MOLAP	ROLAP
Data Model	Simple (Star schema structure)	Controlled by underlying relational schema
ETL Logic	High dependency. All data must be updated on regular basis.	No ETL involvement
Data Latency	High due to large processing times of ETL jobs	Low. Data is always read directly from source system.
Query Processing Time	Fast. Only data volume is limitation.	Slow and inefficient in complex scenarios
Data Federation	Not applicable	Yes. Complex landscape requirements
Data Staging	All data is staged	No staging of data
SQL	Simple SELECT and aggregate	Complex JOIN operations as well as multi-step processing
Reporting Tools Processing	Minimal in most cases	High. Only raw data is fetched.

Figure 2-10 Differences between MOLAP and ROLAP.

Source: http://wisdomschema.com/2016/08/molap-rolap/

2.4.4 Hybrid OLAP (HOLAP)

The need to use the benefits of both MOLAP and ROLAP has led to the use of HOLAP that combines the capabilities of both aforementioned types and in addition provides the option to store a portion of the data in MOLAP and a portion of the data in ROLAP. That said, the more detailed and larger, volume-wise, data can be stored in relational databases whereas the smaller, more “aggregatable” data can be stored in the multidimensional cubes.

There are 2 modes in HOLAP:

- Vertical partitioning mode, where the aggregations are stored in MOLAP and the more detailed data are stored in ROLAP, leads to the optimization of the time of the cube processing
- Horizontal partitioning mode, where the most recent data (sliced by the TIME dimension) is stored in MOLAP and the older data are stored in ROLAP, leads to faster query performance

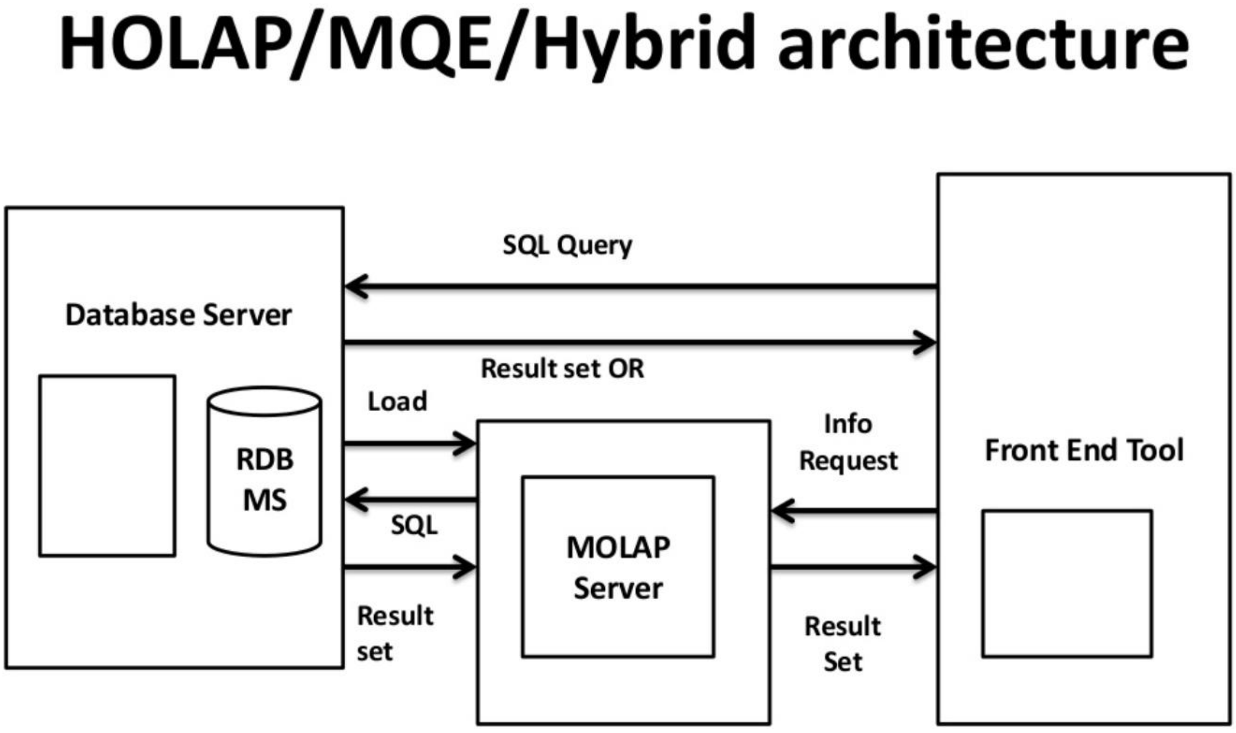


Figure 2-11 HOLAP Architecture

Source: https://www.researchgate.net/figure/HOLAP-architecture-The-main-advantages-of-HOLAP-include-o-High-performance-dimensional_fig21_319852408

3 Methodology

OLAP analysis is used vastly by many companies in different areas of expertise. Below are described the open source Mondrian OLAP engine as well as the Retail Predictive Application Server by Oracle.

3.1 Introduction

In this section it described the methodology that was followed during the analysis.

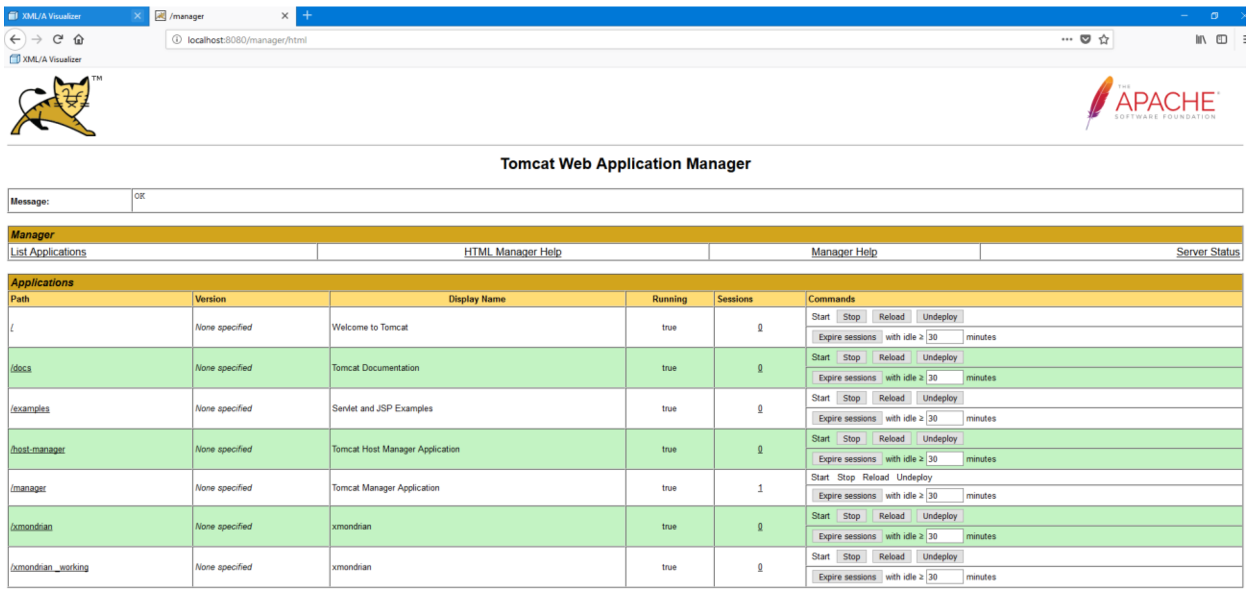
3.2 OLAP Analysis Bibliography

As a first step, many sources regarding OLAP have been searched through the internet. The analysis was based on Mondrian's documentation and Oracle's RPAS Administration Guide in which they are described the main concepts of OLAP analysis.

3.3 Tomcat Java Server

As it was mentioned, the Apache Tomcat was installed to deploy the XMondrian.

Apache Tomcat is an open source software that provides a web server environment on which applications written in Java can be deployed.



Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/xmondrian	None specified	xmondrian	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes
/xmondrian_working	None specified	xmondrian	true	0	Start Stop Reload Undeploy Expire sessions with idle ≥ 30 minutes

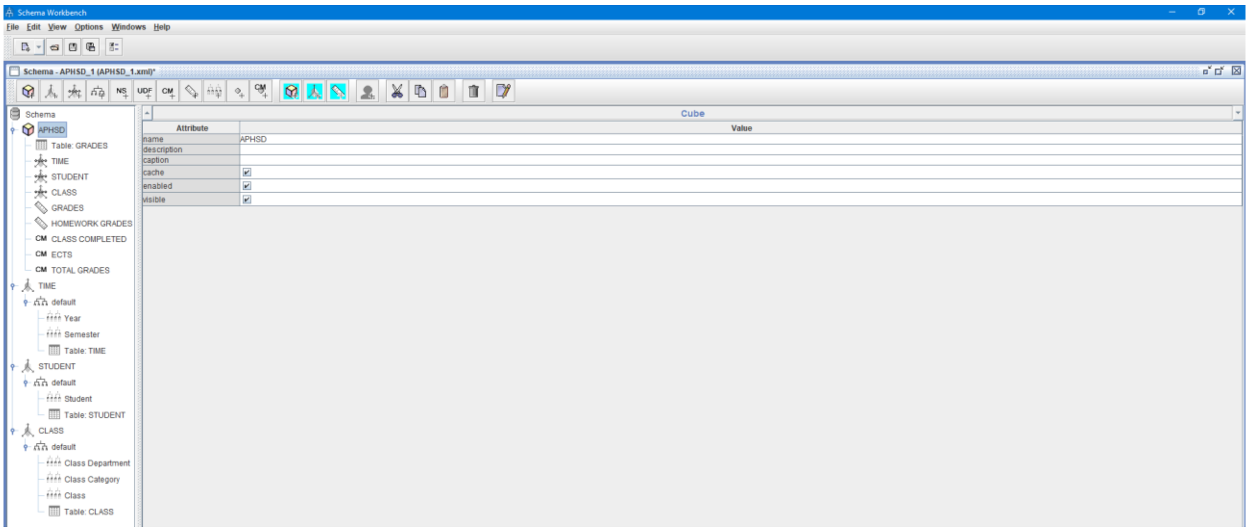
Figure 3-1 Apache Tomcat where Xmondrian was deployed

Deploying a quite easy process. Instructions on how a web application can be deployed in Tomcat 9.0 can be found here: *https://tomcat.apache.org/tomcat-9.0-doc/index.html*

3.4 Mondrian Schema Workbench

At this step, it was identified the process of loading data on XMondrian. The data can be loaded using XML documents to define the SQL queries which subsequently fetch the data from an HyperSQL database.

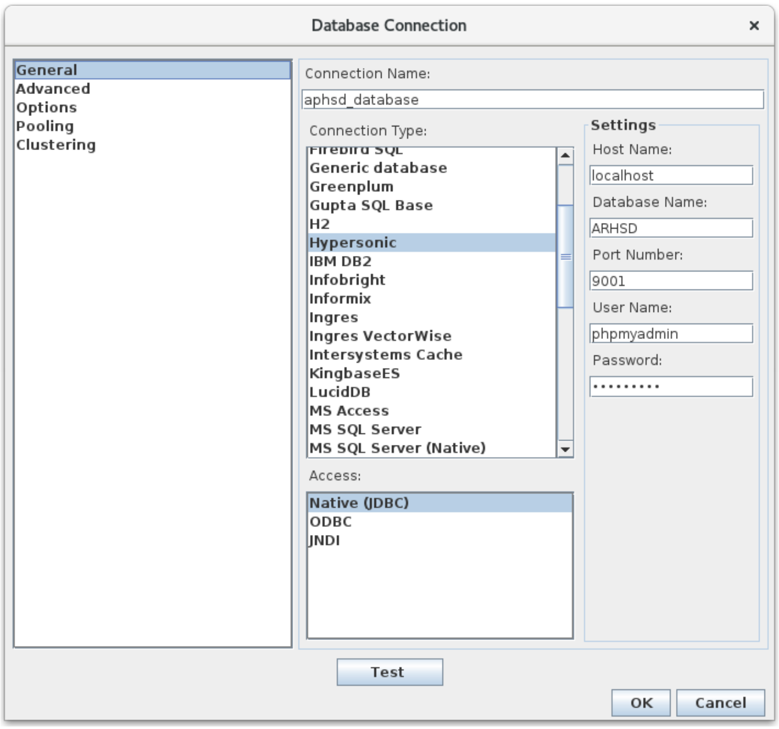
The tool that was used to create those XML file is the Mondrian Schema Workbench which is a tool to manipulate XML file in a user-friendly way.



Attribute	Value
name	APHSD
description	
caption	
cache	✓
enabled	✓
visible	✓

Figure 3-2 Mondrian Schema Workbench

As a first step a connection with a database is required. This can be done by clicking on the Menu panel on the **Options → Connection…**



Connection Name: aphsd_database

Connection Type: Hypersonic

Access: Native (JDBC)

Settings

Host Name: localhost

Database Name: ARHSD

Port Number: 9001

User Name: phpmyadmin

Password: ••••••••••

Figure 3-3 Database Connection

In the window that appears, the database information like *hostname, database name, Port number, User Name and Password* that will be used should be stated. The required information may vary as a lot of database providers can be used.

After all the required information is in place then the “**Test**” button should be pressed to verify that connection has been established.

Note: Schema Workbench cannot work unless a connection to a database is established.

Now the schema file is ready to be created.

As a first step the hierarchies can be created by clicking on the **Create Hierarchy** button on the toolbar. A table of the connected database should be selected from which the hierarchical data will be fetched and then by right clicking and selecting **Add Level**, the desired levels of the hierarchy can be created.

As soon as the hierarchies are defined, a cube can be created by clicking on the **Create Cube** button in the tool bar. The fact table of the database should be selected. In addition, the hierarchies that will be used within this cube can be added by clicking on **Create Used Dimensions** button.

The last required element is the measures which can be added by right clicking and selecting **Add Measure** that is defined to fetch the data from the fact table and represent a metric to be measured such as the Grades of the Students. Additionally, apart from the measures that are “loaded” in the application, the user can create calculated measures by clicking on the button **Create Calculated Member** and fill the desired formula of the calculation. For example for the calculation of the Total Grades and the ECTS, the following formulas can be used using MDX structure.

- ***Total Grades*** *= ([Measures].[Grades] * 0.6) + ([Measures].[Homework Grades] * 0.4)*
- ***ECTS*** *= [Measure].[Total Grades] + [Measures].[Total Grades] * 0.5*

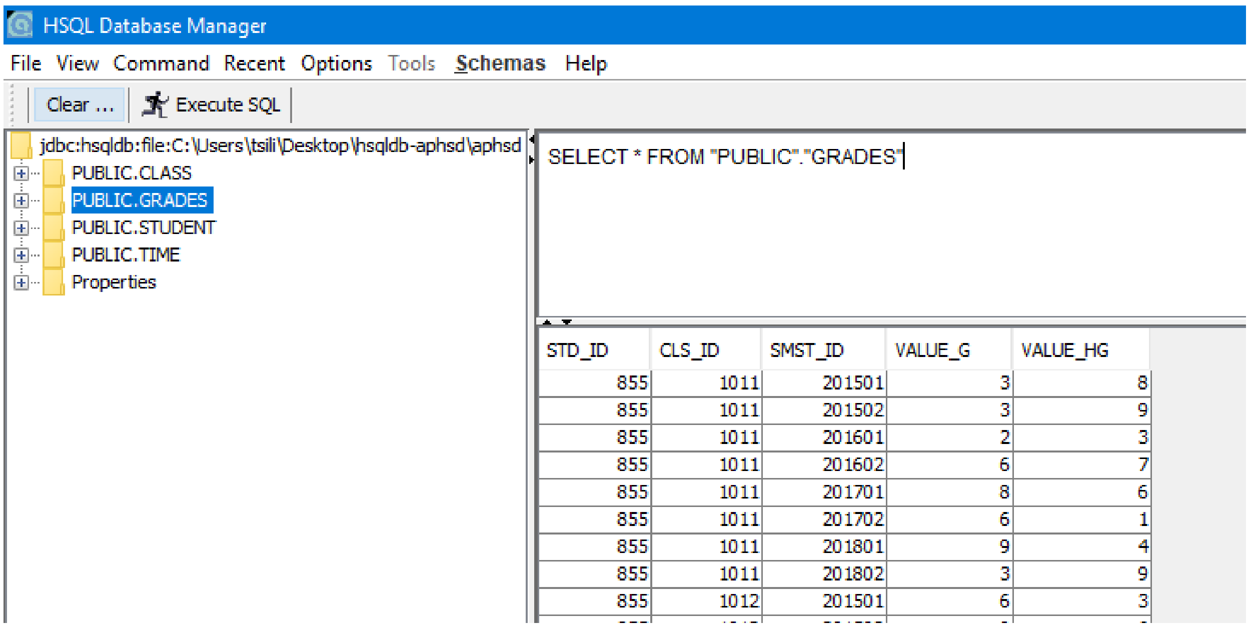
Meaning that the Total Grades can be calculated as the weighted sum of the measures Grades and Homework Grades and accordingly, the ECTS can be calculated as the Total Grades increased by 150%.

For more information regarding the Schema Workbench, the Pentaho Mondrian documentation can be found here: *https://mondrian.pentaho.com/documentation/workbench.php*

3.5 HyperSQL

Mondrian uses HSQLDB among many others to store the data and thus the documentation was used to gain better understanding of HSQLDB.

HyperSQL is a relational database software written in Java. Some of it’s benefits is it’s small size, it’s speed and the ability to perform all the calculations in memory. However for the purposes of this implementation, the data was stored in an hsql database file on which the Xmondrian was connected and was extracting the data.



jdbc:hsqldb:file:C:\Users\tsili\Desktop\hsqldb-aphsd\aphsd
PUBLIC.CLASS
PUBLIC.GRADES
PUBLIC.STUDENT
PUBLIC.TIME
Properties

```
SELECT * FROM "PUBLIC"."GRADES"
```

STD_ID	CLS_ID	SMST_ID	VALUE_G	VALUE_HG
855	1011	201501	3	8
855	1011	201502	3	9
855	1011	201601	2	3
855	1011	201602	6	7
855	1011	201701	8	6
855	1011	201702	6	1
855	1011	201801	9	4
855	1011	201802	3	9
855	1012	201501	6	3

Figure 3-4 HSQL Database Manager

The HSQLDB Manager tool can be downloaded from here:
https://sourceforge.net/projects/hsqldbmanager/

Instructions and useful information can be found here: http://hsqldb.org/doc/util-guide/dbm-chapt.html

3.6 *Multi-Dimensional Expressions (MDX)*

Multi-dimensional expressions or MDX is a query language which is an extension of SQL but with the main difference that the queries that are written in MDX are used to fetch data from multi-dimensional databases or cubes.

In each query it needs to be defined the level of the measure or the hierarchy that the data needs to be extracted from.

```
SELECT
    { [Measures].[Sales Amount],
        [Measures].[Tax Amount] } ON COLUMNS,
    { [Date].[Fiscal].[Fiscal Year].&[2002],
        [Date].[Fiscal].[Fiscal Year].&[2003] } ON ROWS
FROM [Adventure Works]
WHERE ( [Sales Territory].[Southwest] )
```

Figure 3-5 MDX Query example

Xmondrian "translates" the drag'n'drop functionality to MDX queries.

For a complete documentation regarding MDX you can visit the following:
https://docs.microsoft.com/en-us/sql/analysis-services/multidimensional-models/mdx/multidimensional-model-data-access-analysis-services-multidimensional-data?view=sql-server-2017

3.7 Xavier Visualizer Tool

Xavier is the tool to present the data analysis in graphical way. Visual presentation is an essential piece of OLAP analysis since it provides the information in a way which a user is not required to have any technical expertise to interpret the data.

Xmondrian can be used as a standalone application that utilizes the functionality of the Mondrian OLAP Engine without having to install all Pentaho Mondrian components and applications.

The most useful feature of Xavier is that it works in an interactive way. This means that in the graphical user interface, the user can create different pivot tables of the measures by choosing, dynamically, different dimensions.

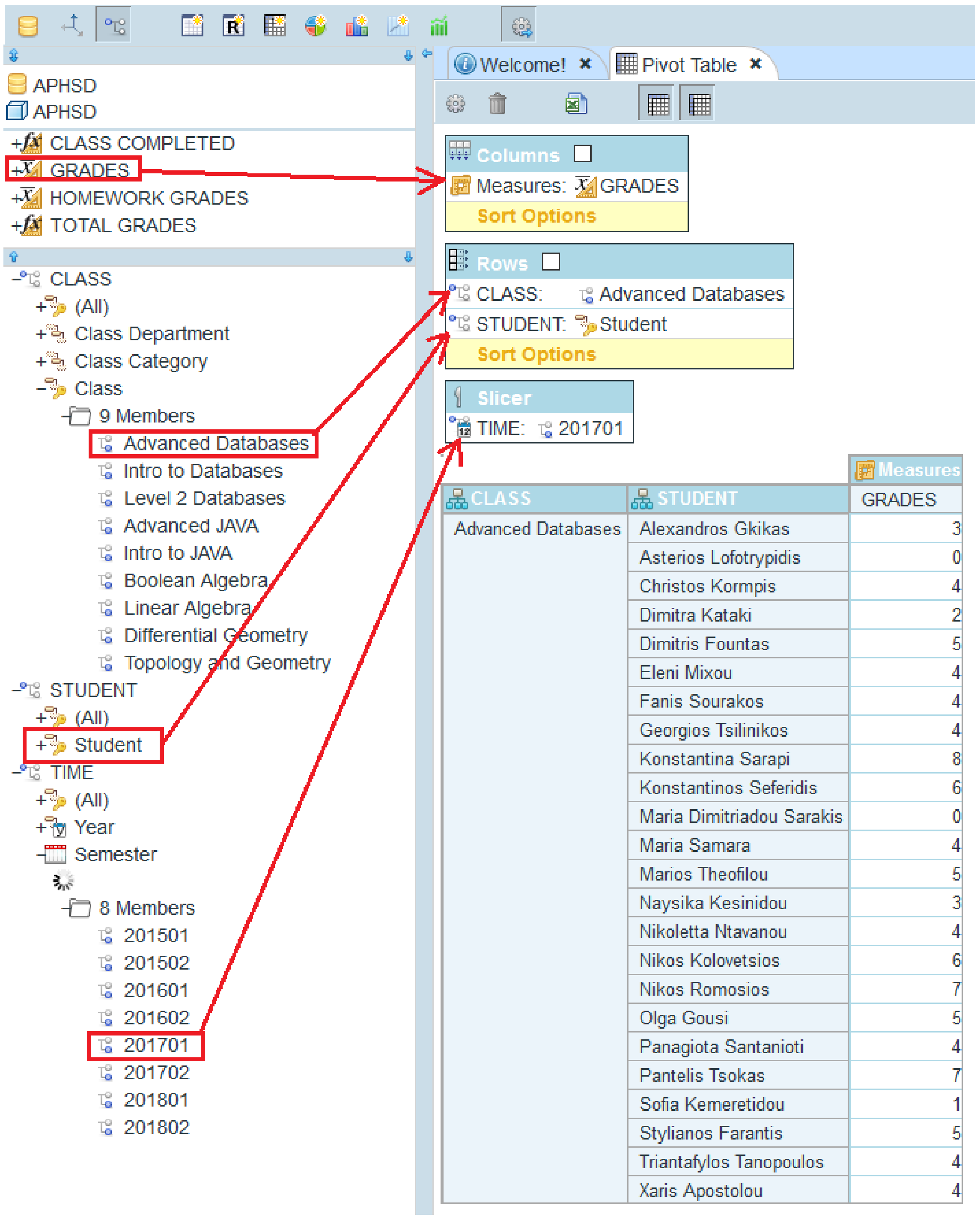
On top of that, the data can be presented in different views, like Bar charts or Pie charts. The users can switch between the views of their preference to achieve optimal data visualization.

The following view can be easily created by simply dragging and dropping the measures, hierarchies or dimensions in the corresponding column, row or slicer tiles.

GRADES measure is dropped in the Column tile to show the grades values in the X-axis.

Advanced Databases class is dropped in the Row tile to show the class in the Y-axis. Additionally, the **Student** Dimension can be also dropped in the Row tile and show the grades of the students one-by-one for the Advanced Databases class. (Note: If the “All Students” was dropped, then the view would should the average grades of all the students for this class).

Lastly, as a slicer, the first semester of 2017, **2018701** will dropped in the slicer tile to filter only the grades of all students for that class for this semester.



Measures: GRADES

CLASS: Advanced Databases
STUDENT: Student

TIME: 201701

- CLASS
  - (All)
  - Class Department
  - Class Category
  - Class
    - 9 Members
      - Advanced Databases
      - Intro to Databases
      - Level 2 Databases
      - Advanced JAVA
      - Intro to JAVA
      - Boolean Algebra
      - Linear Algebra
      - Differential Geometry
      - Topology and Geometry
- STUDENT
  - (All)
  - Student
- TIME
  - (All)
  - Year
  - Semester
    - 8 Members
      - 201501
      - 201502
      - 201601
      - 201602
      - 201701
      - 201702
      - 201801
      - 201802

		Measures
CLASS	STUDENT	GRADES
Advanced Databases	Alexandros Gkikas	3
	Asterios Lofotrypidis	0
	Christos Kormpis	4
	Dimitra Kataki	2
	Dimitris Fountas	5
	Eleni Mixou	4
	Fanis Sourakos	4
	Georgios Tsilinikos	4
	Konstantina Sarapi	8
	Konstantinos Seferidis	6
	Maria Dimitriadou Sarakis	0
	Maria Samara	4
	Marios Theofilou	5
	Naysika Kesinidou	3
	Nikoletta Ntavanou	4
	Nikos Kolovetsios	6
	Nikos Romosios	7
	Olga Gousi	5
	Panagiota Santanioti	4
	Pantelis Tsokas	7
	Sofia Kemeretidou	1
	Stylianos Farantis	5
	Triantafylos Tanopoulos	4
	Xaris Apostolou	4

Figure 3-6 Drag'n'drop functionality of Xavier

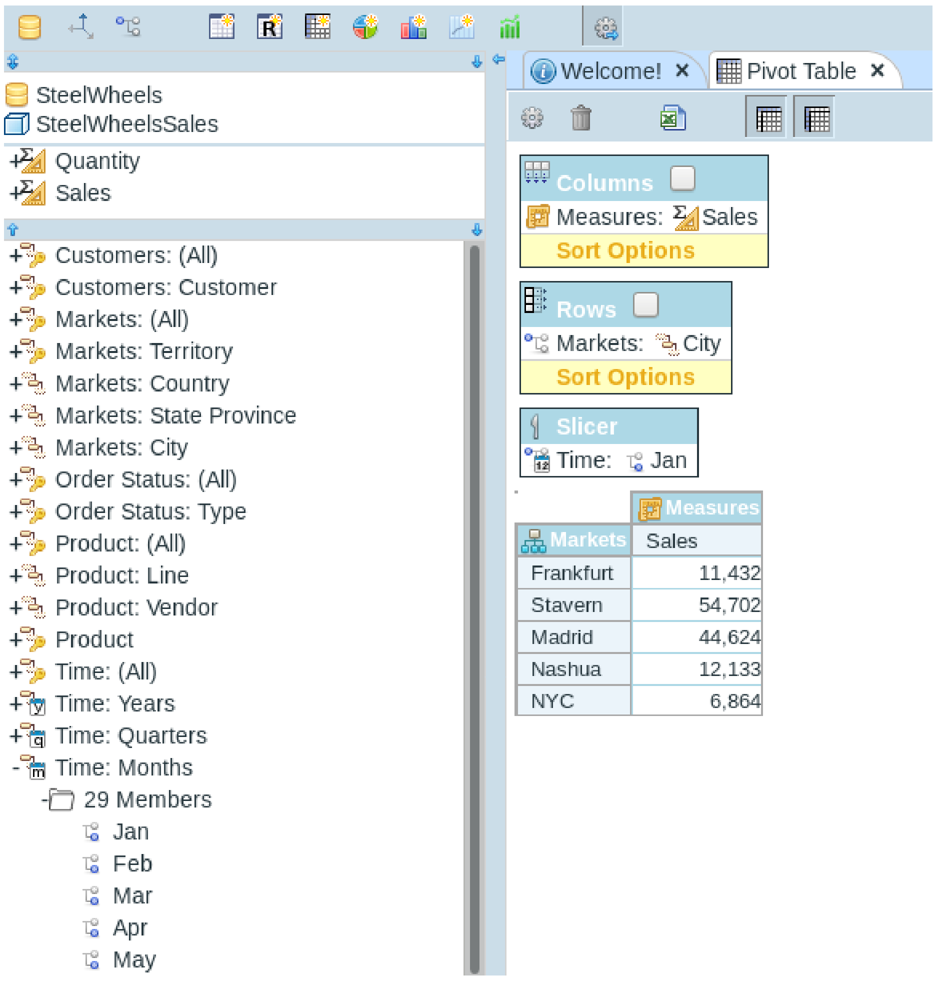
4 Data Analysis & Interpretation of findings

In this section it is described the high-level analysis of the embedded datasets that were provided within XMondrian along with the created dataset. The concepts of the OLAP analysis are described and presented with specific examples.

Mondrian can be used as a DSS (Decision Support System) to provide different views of data as well as a graphical user interface to provide a graphical presentation of the data in configurable dashboards.

4.1 Pivot Table

The most common view in Mondrian is the Pivot Table. It looks like excel spreadsheet, however it provides an interactive way to present the data. More specifically, the user can move within a hierarchy by rolling up, in case a higher level of granularity is required, of by drilling down, in case a more detailed view of the data is required.



SteelWheels
SteelWheelsSales

+ Quantity
+ Sales

+ Customers: (All)
+ Customers: Customer
+ Markets: (All)
+ Markets: Territory
+ Markets: Country
+ Markets: State Province
+ Markets: City
+ Order Status: (All)
+ Order Status: Type
+ Product: (All)
+ Product: Line
+ Product: Vendor
+ Product
+ Time: (All)
+ Time: Years
+ Time: Quarters
- Time: Months
  - 29 Members
    - Jan
    - Feb
    - Mar
    - Apr
    - May

Columns
Measures: Sales
Sort Options

Rows
Markets: City
Sort Options

Slicer
Time: Jan

	Measures
Markets	Sales
Frankfurt	11,432
Stavern	54,702
Madrid	44,624
Nashua	12,133
NYC	6,864

Figure 4-1 Sales per store in January

4.2 Pie Chart

The bar chart is the type of view in which the data can be presented in a graphical way to provide the user the portion of the total metric which is presented. This view can be used in dashboard in decision support systems to provide the required information to a user when it comes to take decisions.

In the following figure, an example of the average sales per country is presented and thus a strategic decision can be made to stop selling in the country with the lowest sales.

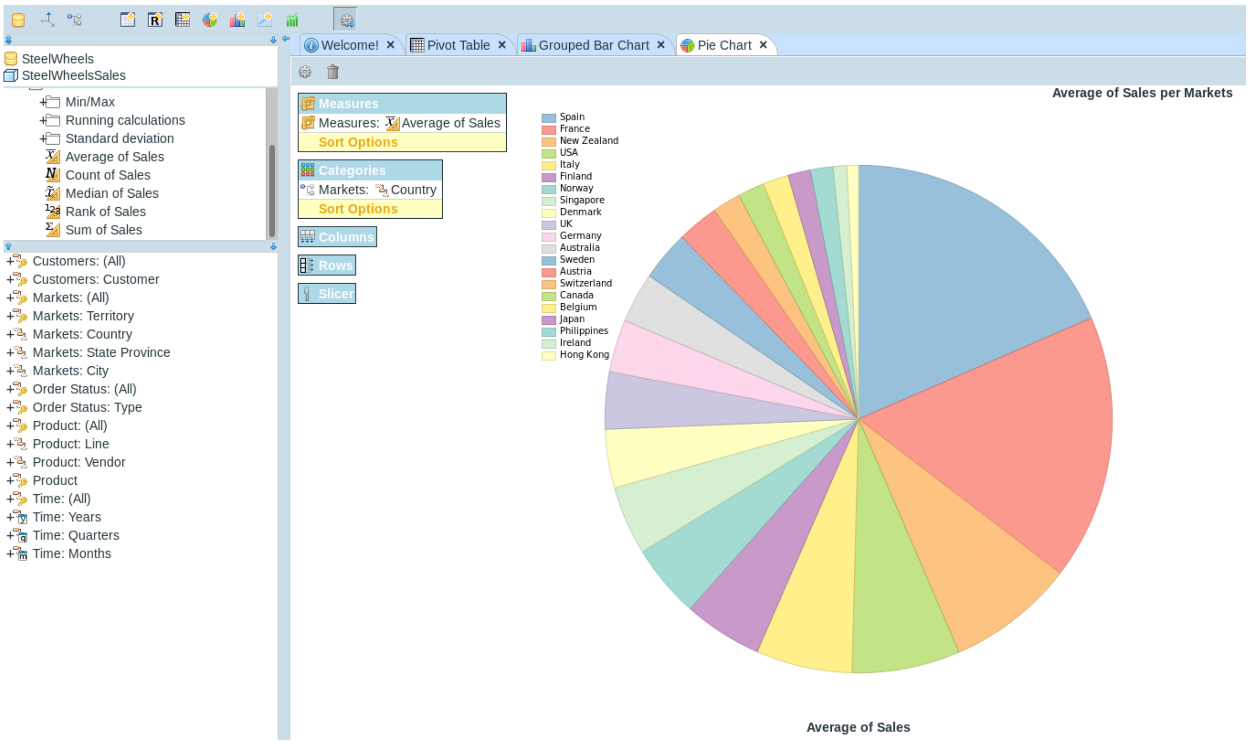


Figure 4-2 Average Sales per country

4.3 Bar Chart

In case a different view is needed, where the data needs to be compared visually, the bar chart can be used.

The data can be presented in different bars and thus again such view can provide the required information for a decision to be taken.

For example, the summary of sales per product (or per product class) can presented in a bar chart to compare the total sales of each product class and decided which class is the top seller and needs to be allocated in more stores.

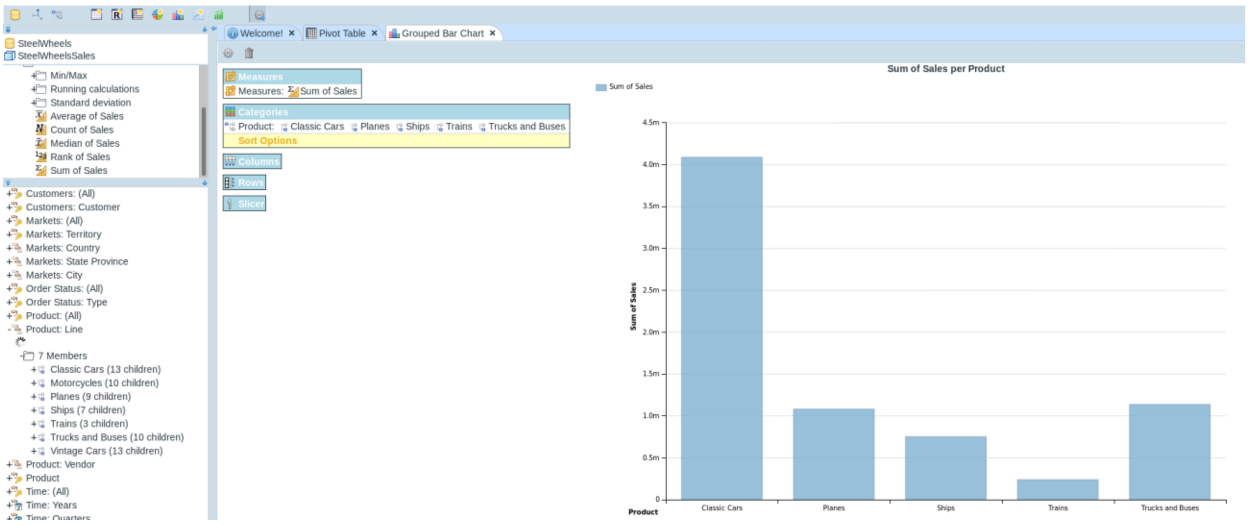


Figure 4-3 Sum of sales per Product class

5 Examples of platforms that use OLAP Analysis

Most companies decided to switch from excel spreadsheets to sophisticated OLAP engines, due the complexity of using and maintaining their data in excel. OLAP Cubes provide a structured storage of the data as well as a user friendly interface and easy switching between different views. In addition, OLAP provide much faster calculation performance in big volumes of data, either in batch calculation or adhoc.

5.1 Pentaho Mondrian – APHSD multidimensional database

Mondrian is an open source OLAP engine that was designed by Julian Hyde and was adopted by Pentaho company. It provides an MDX engine which is a query language, similar to SQL, but for multidimensional databases.

For the purposes of this thesis, Xmondrian was used to recreate a scenario in which the Mondian OLAP Engine can be used. Xmondrian is .war file that contains a couple of OLAP tools along with the embedded Cube and Dataset of Foodmart and SteelWheels.

In our case a new Cube and dataset were created to provide another example of the use of Mondrian.

A new star schema was created to represent the grades of the students for each class and for each semester.

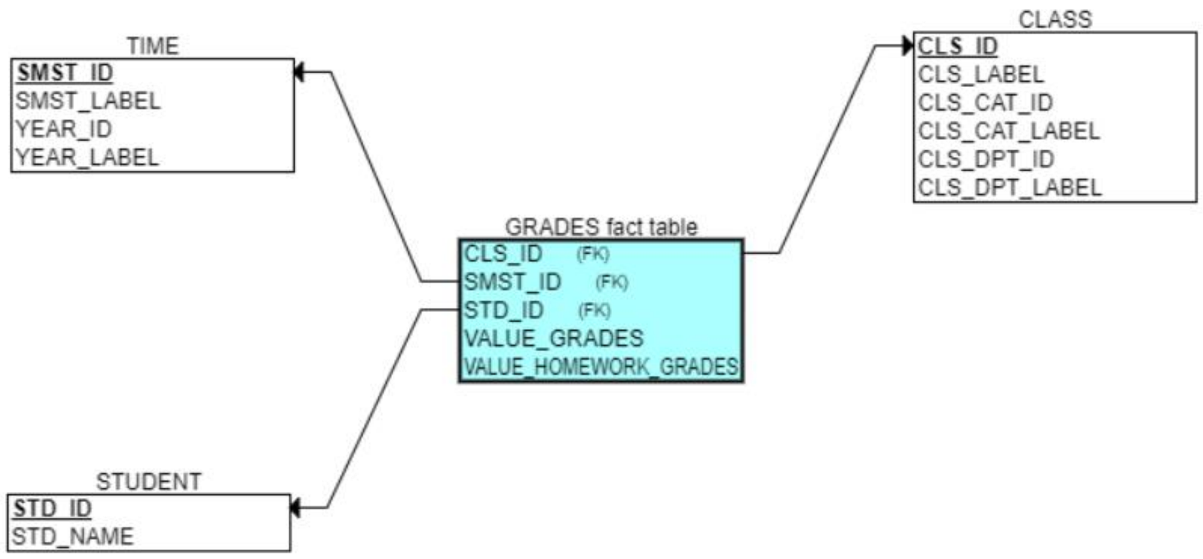
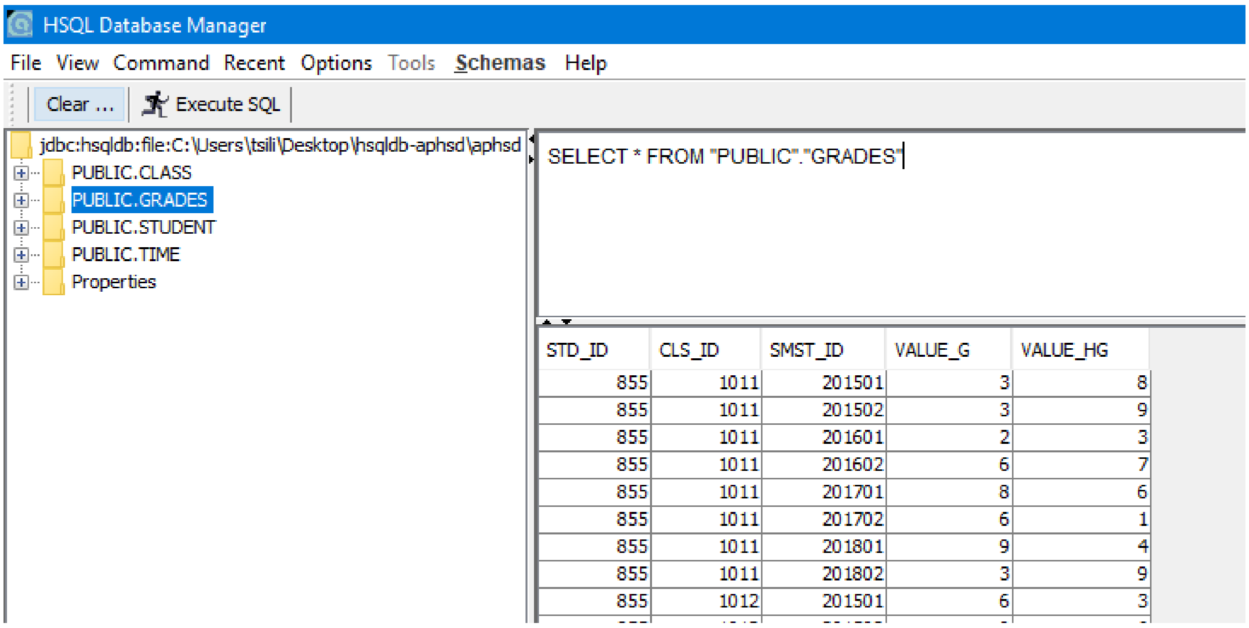


Figure 5-1 Grades Star Schema

This schema was translated in SQL and was loaded in a HSQL database file.


HSQL Database Manager
File View Command Recent Options Tools Schemas Help
Clear ... Execute SQL
jdbc:hsqldb:file:C:\Users\tsili\Desktop\hsqldb-aphsd\aphsd
PUBLIC.CLASS
PUBLIC.GRADES
PUBLIC.STUDENT
PUBLIC.TIME
Properties

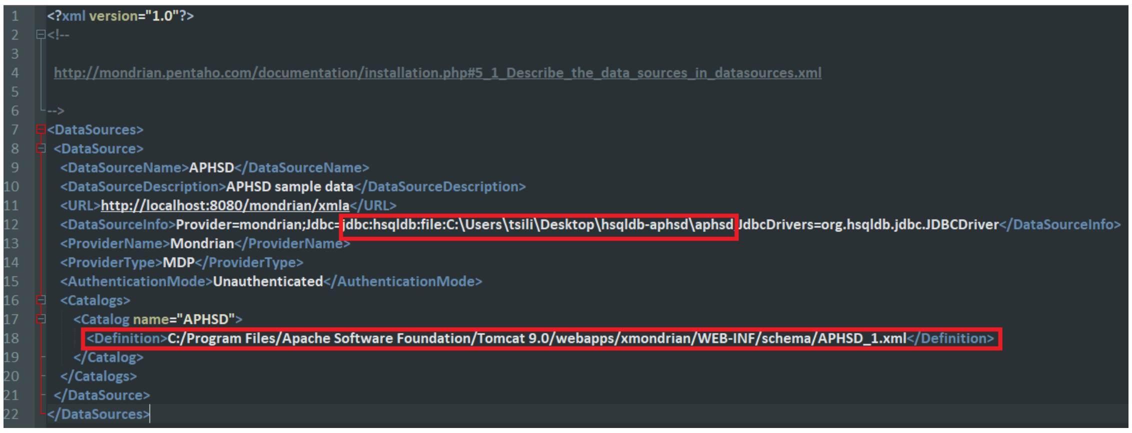
```
SELECT * FROM "PUBLIC"."GRADES"
```

STD_ID	CLS_ID	SMST_ID	VALUE_G	VALUE_HG
855	1011	201501	3	8
855	1011	201502	3	9
855	1011	201601	2	3
855	1011	201602	6	7
855	1011	201701	8	6
855	1011	201702	6	1
855	1011	201801	9	4
855	1011	201802	3	9
855	1012	201501	6	3

Figure 5-2 HSQLDB Manager

Xmondrian connects to this database file and fetches the data in the web application. There are 3 base files that are required from Xmondrian:

- **Datasources.xml**
  In this XML file is specified jdbc driver, the database that the Xmondrian will connect to and the directory in which the schema will be found.
  In the following screenshot are highlighted the path for the connection to the database file and path where the application can find the schema XML file.



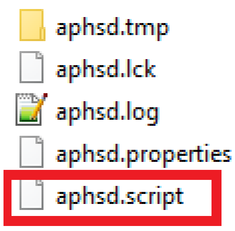
```
<?xml version="1.0"?>
<!--

 http://mondrian.pentaho.com/documentation/installation.php#5_1_Describe_the_data_sources_in_datasources.xml

-->
<DataSources>
 <DataSource>
  <DataSourceName>APHSD</DataSourceName>
  <DataSourceDescription>APHSD sample data</DataSourceDescription>
  <URL>http://localhost:8080/mondrian/xmla</URL>
  <DataSourceInfo>Provider=mondrian;Jdbc=jdbc:hsqldb:file:C:\Users\tsili\Desktop\hsqldb-aphsd\aphsd;JdbcDrivers=org.hsqldb.jdbc.JDBCDriver</DataSourceInfo>
  <ProviderName>Mondrian</ProviderName>
  <ProviderType>MDP</ProviderType>
  <AuthenticationMode>Unauthenticated</AuthenticationMode>
  <Catalogs>
    <Catalog name="APHSD">
     <Definition>C:/Program Files/Apache Software Foundation/Tomcat 9.0/webapps/xmondrian/WEB-INF/schema/APHSD_1.xml</Definition>
    </Catalog>
  </Catalogs>
 </DataSource>
</DataSources>
```

Figure 5-3 datasources.xml

- **<database_file>** (i.e. aphsddb)
  The database file from which the Xmondrian will fetch the data

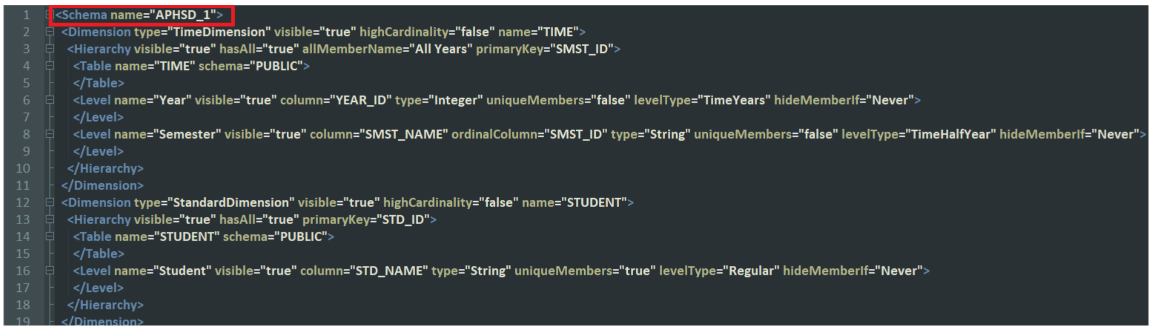


```
aphsd.tmp
aphsd.lck
aphsd.log
aphsd.properties
aphsd.script
```

Figure 5-4 HSQLDB database file

- **<schema>.xml** (i.e. APHSD_1.xml)
  The XML file which is considered as the blueprint of the application. It describes the structure and the components of the multidimensional schema and specifies the way that the data will be fetched from the relational database and will be presented to the user.

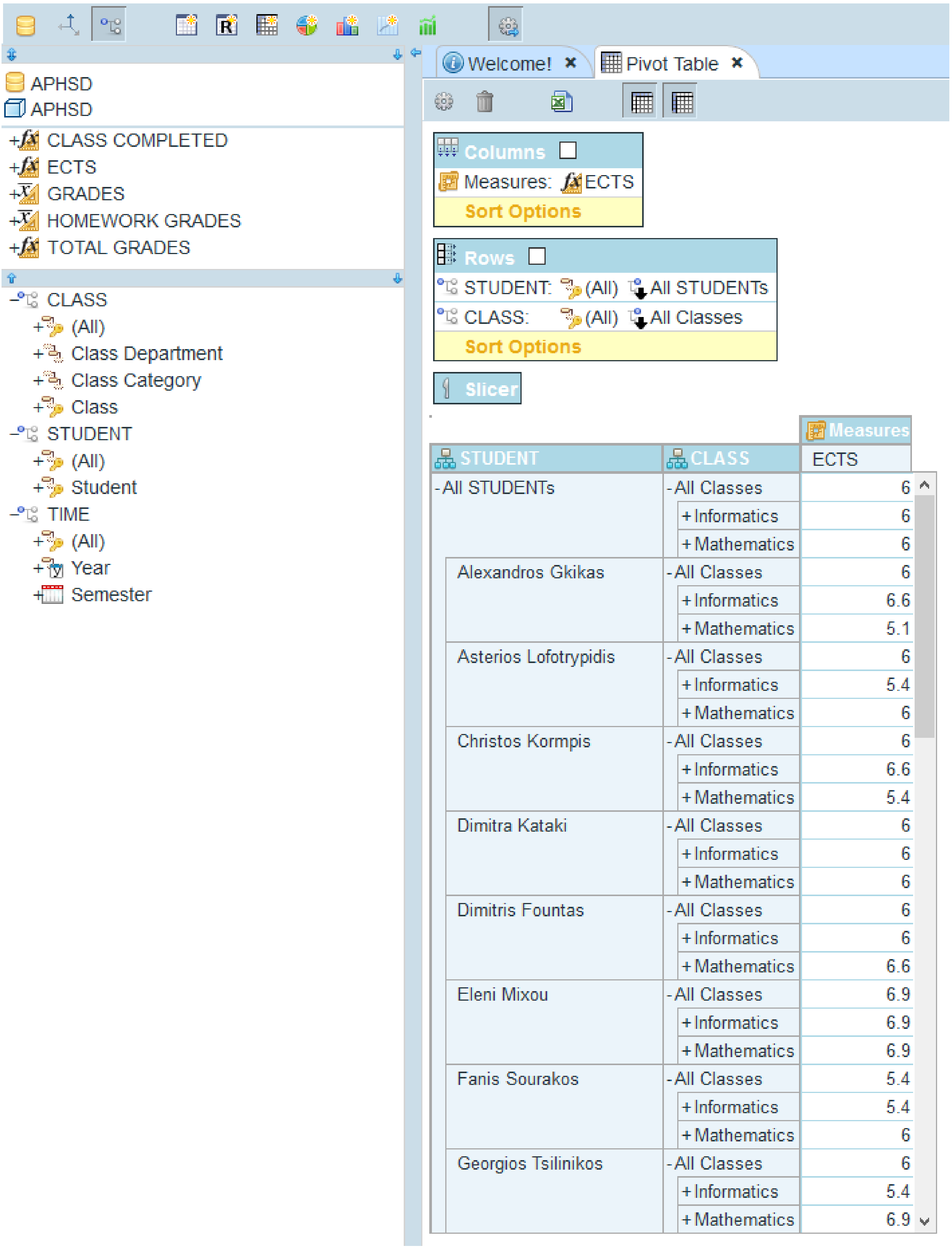
  In the following screenshot is a part of the schema XML file that was created by the Schema Workbench tool.



```xml
<Schema name="APHSD_1">
 <Dimension type="TimeDimension" visible="true" highCardinality="false" name="TIME">
  <Hierarchy visible="true" hasAll="true" allMemberName="All Years" primaryKey="SMST_ID">
   <Table name="TIME" schema="PUBLIC">
   </Table>
   <Level name="Year" visible="true" column="YEAR_ID" type="Integer" uniqueMembers="false" levelType="TimeYears" hideMemberIf="Never">
   </Level>
   <Level name="Semester" visible="true" column="SMST_NAME" ordinalColumn="SMST_ID" type="String" uniqueMembers="false" levelType="TimeHalfYear" hideMemberIf="Never">
   </Level>
  </Hierarchy>
 </Dimension>
 <Dimension type="StandardDimension" visible="true" highCardinality="false" name="STUDENT">
  <Hierarchy visible="true" hasAll="true" primaryKey="STD_ID">
   <Table name="STUDENT" schema="PUBLIC">
   </Table>
   <Level name="Student" visible="true" column="STD_NAME" type="String" uniqueMembers="true" levelType="Regular" hideMemberIf="Never">
   </Level>
  </Hierarchy>
 </Dimension>
```

Figure 5-5 APHSD_1.xml schema

After everything is set, the user can open the Xmondrian and begin the analysis by dragging and dropping measures and dimensions in the corresponding tiles.



APHSD
APHSD
+CLASS COMPLETED
+ECTS
+GRADES
+HOMEWORK GRADES
+TOTAL GRADES
−CLASS
+(All)
+Class Department
+Class Category
+Class
−STUDENT
+(All)
+Student
−TIME
+(All)
+Year
+Semester

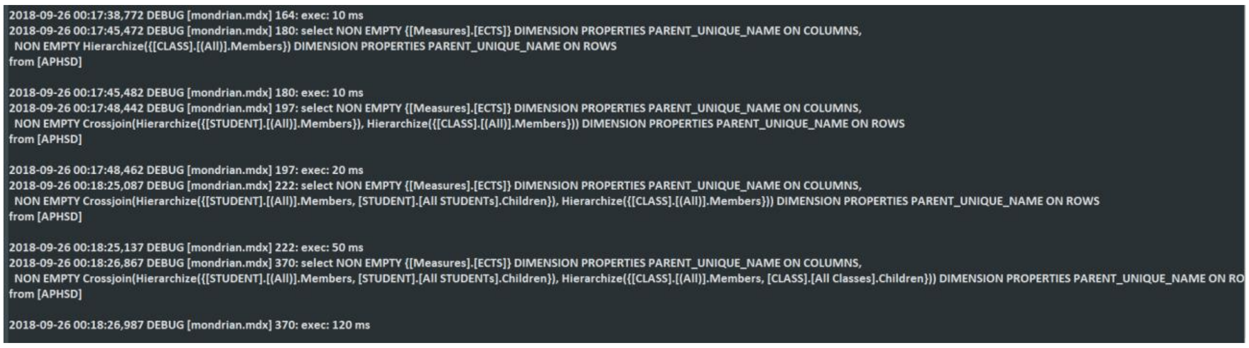
Columns: Measures: ECTS

Rows: STUDENT: (All) All STUDENTs; CLASS: (All) All Classes

		Measures
STUDENT	CLASS	ECTS
-All STUDENTs	-All Classes	6
	+Informatics	6
	+Mathematics	6
Alexandros Gkikas	-All Classes	6
	+Informatics	6.6
	+Mathematics	5.1
Asterios Lofotrypidis	-All Classes	6
	+Informatics	5.4
	+Mathematics	6
Christos Kormpis	-All Classes	6
	+Informatics	6.6
	+Mathematics	5.4
Dimitra Kataki	-All Classes	6
	+Informatics	6
	+Mathematics	6
Dimitris Fountas	-All Classes	6
	+Informatics	6
	+Mathematics	6.6
Eleni Mixou	-All Classes	6.9
	+Informatics	6.9
	+Mathematics	6.9
Fanis Sourakos	-All Classes	5.4
	+Informatics	5.4
	+Mathematics	6
Georgios Tsilinikos	-All Classes	6
	+Informatics	5.4
	+Mathematics	6.9

Figure 5-6 MDX front-end

In the backend MDX queries are being executed based on the “drag’n’drop” of the measures/dimension construct that the user performs.



```
2018-09-26 00:17:38,772 DEBUG [mondrian.mdx] 164: exec: 10 ms
2018-09-26 00:17:45,472 DEBUG [mondrian.mdx] 180: select NON EMPTY {[Measures].[ECTS]} DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON COLUMNS,
 NON EMPTY Hierarchize({[CLASS].[(All)].Members}) DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON ROWS
from [APHSD]

2018-09-26 00:17:45,482 DEBUG [mondrian.mdx] 180: exec: 10 ms
2018-09-26 00:17:48,442 DEBUG [mondrian.mdx] 197: select NON EMPTY {[Measures].[ECTS]} DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON COLUMNS,
 NON EMPTY Crossjoin(Hierarchize({[STUDENT].[(All)].Members}), Hierarchize({[CLASS].[(All)].Members})) DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON ROWS
from [APHSD]

2018-09-26 00:17:48,462 DEBUG [mondrian.mdx] 197: exec: 20 ms
2018-09-26 00:18:25,087 DEBUG [mondrian.mdx] 222: select NON EMPTY {[Measures].[ECTS]} DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON COLUMNS,
 NON EMPTY Crossjoin(Hierarchize({[STUDENT].[(All)].Members, [STUDENT].[All STUDENTs].Children}), Hierarchize({[CLASS].[(All)].Members})) DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON ROWS
from [APHSD]

2018-09-26 00:18:25,137 DEBUG [mondrian.mdx] 222: exec: 50 ms
2018-09-26 00:18:26,867 DEBUG [mondrian.mdx] 370: select NON EMPTY {[Measures].[ECTS]} DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON COLUMNS,
 NON EMPTY Crossjoin(Hierarchize({[STUDENT].[(All)].Members, [STUDENT].[All STUDENTs].Children}), Hierarchize({[CLASS].[(All)].Members, [CLASS].[All Classes].Children})) DIMENSION PROPERTIES PARENT_UNIQUE_NAME ON RO
from [APHSD]

2018-09-26 00:18:26,987 DEBUG [mondrian.mdx] 370: exec: 120 ms
```

Figure 5-7 MDX example back-end

The default view of Xmondrian is the pivot table. By dragging and dropping the measure GRADES in the measure tile, a single cell will appear with the value of the average grades of all students, in all classes for all semesters. The aggregation method is defined in the APHSD_1.xml file as average and thus all grades for all students, in all classes for all semesters, are aggregate as average.

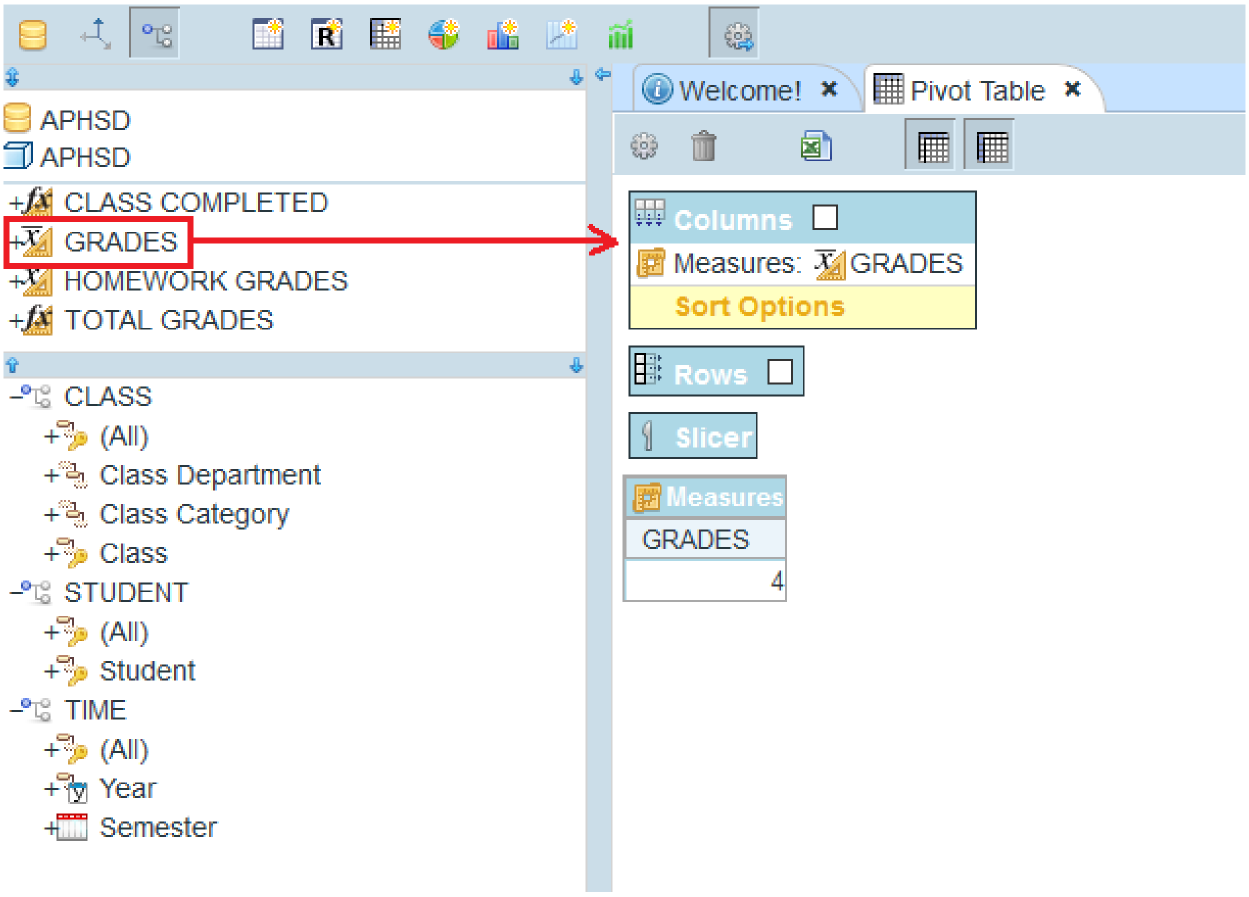
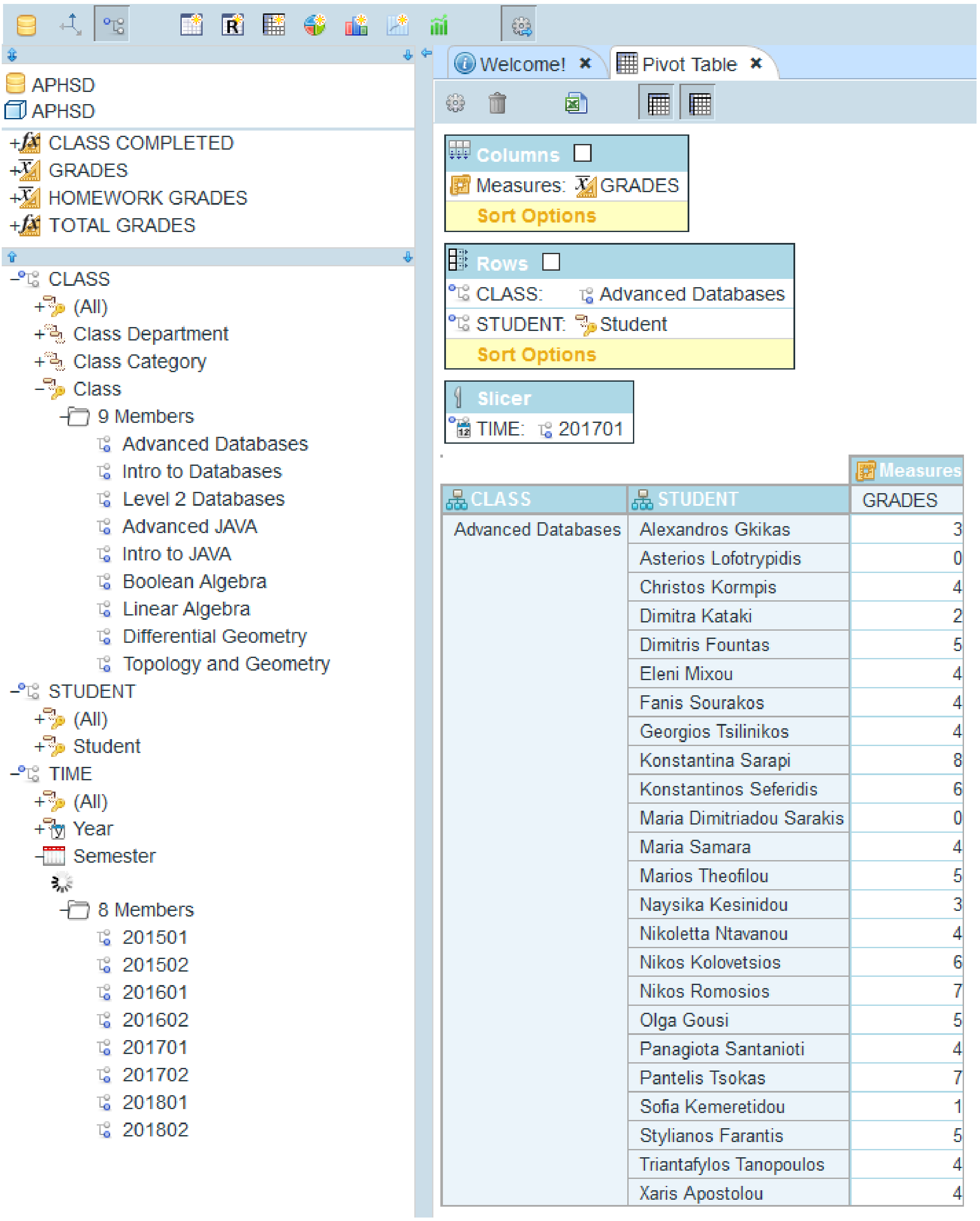


Figure 5-8 Default view of Xmondrian

From now on the user can manipulate the pivot table by rearranging the dimensions and creating different view depending on the scope that the data need to be observed.

That said, the data can be arranged in ways that can be interpreted differently and providing the user multiple scopes for the analysis.

For example, the view can be arranged to represent the grades of the class “Advanced Databases” for all Students for the 1st semester of 2017. That way, it’s easier for the user to understand how was the students` performance in this particular class for that semester. This could lead to several observations/decisions like the Students performed poorly this year and thus we need to revisit the final exams and identify what led to this situation.



APHSD
APHSD
+ CLASS COMPLETED
+ GRADES
+ HOMEWORK GRADES
+ TOTAL GRADES

- CLASS
  + (All)
  + Class Department
  + Class Category
  - Class
    - 9 Members
      Advanced Databases
      Intro to Databases
      Level 2 Databases
      Advanced JAVA
      Intro to JAVA
      Boolean Algebra
      Linear Algebra
      Differential Geometry
      Topology and Geometry
- STUDENT
  + (All)
  + Student
- TIME
  + (All)
  + Year
  - Semester
    - 8 Members
      201501
      201502
      201601
      201602
      201701
      201702
      201801
      201802

Welcome! | Pivot Table

Columns
Measures: GRADES
Sort Options

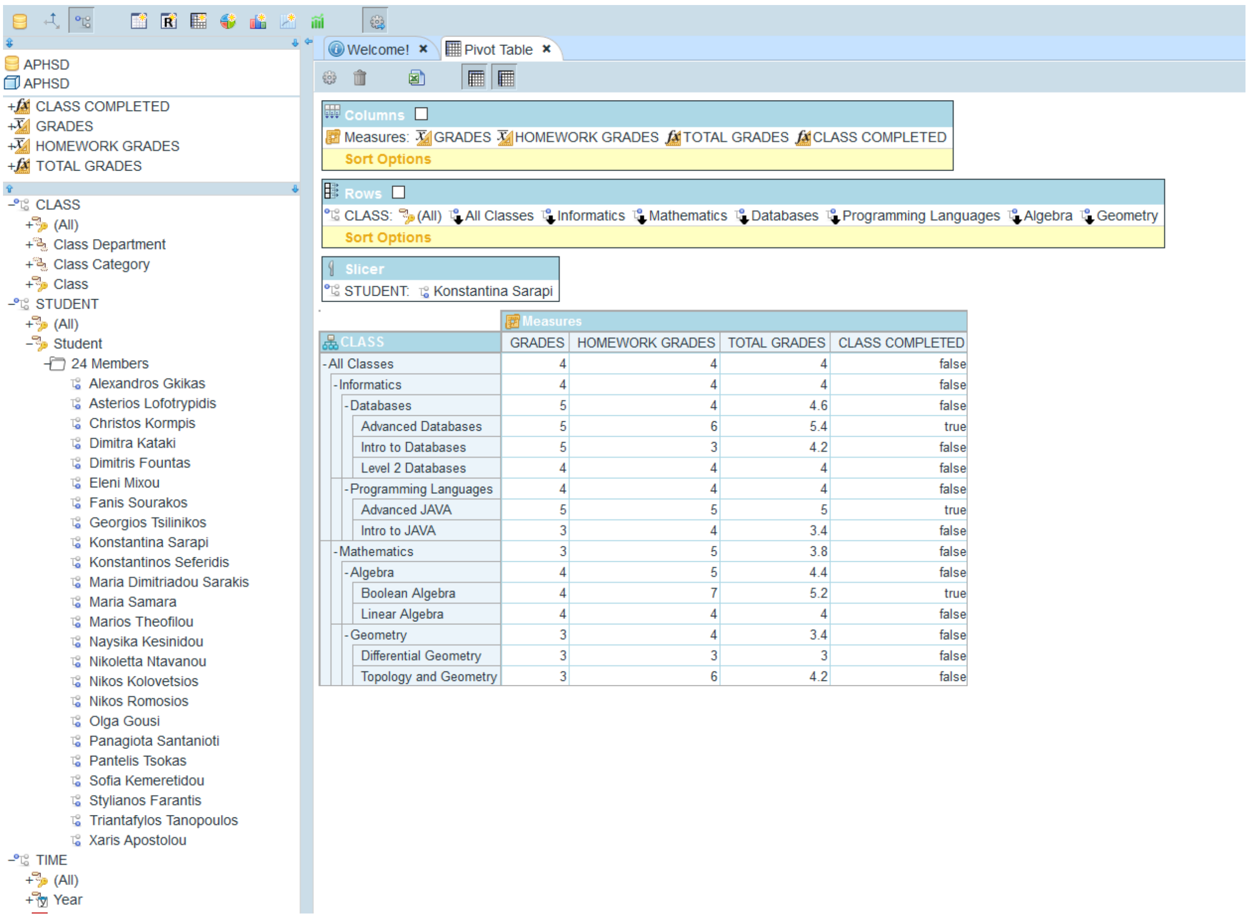
Rows
CLASS: Advanced Databases
STUDENT: Student
Sort Options

Slicer
TIME: 201701

		Measures
CLASS	STUDENT	GRADES
Advanced Databases	Alexandros Gkikas	3
	Asterios Lofotrypidis	0
	Christos Kormpis	4
	Dimitra Kataki	2
	Dimitris Fountas	5
	Eleni Mixou	4
	Fanis Sourakos	4
	Georgios Tsilinikos	4
	Konstantina Sarapi	8
	Konstantinos Seferidis	6
	Maria Dimitriadou Sarakis	0
	Maria Samara	4
	Marios Theofilou	5
	Naysika Kesinidou	3
	Nikoletta Ntavanou	4
	Nikos Kolovetsios	6
	Nikos Romosios	7
	Olga Gousi	5
	Panagiota Santanioti	4
	Pantelis Tsokas	7
	Sofia Kemeretidou	1
	Stylianos Farantis	5
	Triantafylos Tanopoulos	4
	Xaris Apostolou	4

Figure 5-9 The grades of all Students in Class "Advanced Databases" for the first Semester of 2017

On top of that, multiple measures can be arranged one after the other and provide either comparison between two or more measures, or a more high-level view of the data. In our specific example, the measures GRADES, HOMEWORK GRADES, TOTAL GRADES, CLASS COMPLETED can be placed together. Another significant feature here is the “slicing”. The view represents the exam, homework and total grades along with the indicator that a class has been completed, but not for all students. The cube is “slice” and only the data for “Konstantina Sarapi” student is shown. This is one of the aforementioned OLAP operations, the “Slice’n’Dice”.



APHSD
APHSD
+ CLASS COMPLETED
+ GRADES
+ HOMEWORK GRADES
+ TOTAL GRADES

- CLASS
  + (All)
  + Class Department
  + Class Category
  + Class
- STUDENT
  + (All)
  - Student
    - 24 Members
      Alexandros Gkikas
      Asterios Lofotrypidis
      Christos Kormpis
      Dimitra Kataki
      Dimitris Fountas
      Eleni Mixou
      Fanis Sourakos
      Georgios Tsilinikos
      Konstantina Sarapi
      Konstantinos Seferidis
      Maria Dimitriadou Sarakis
      Maria Samara
      Marios Theofilou
      Naysika Kesinidou
      Nikoletta Ntavanou
      Nikos Kolovetsios
      Nikos Romosios
      Olga Gousi
      Panagiota Santanioti
      Pantelis Tsokas
      Sofia Kemeretidou
      Stylianos Farantis
      Triantafylos Tanopoulos
      Xaris Apostolou
- TIME
  + (All)
  + Year

Columns
Measures: GRADES HOMEWORK GRADES TOTAL GRADES CLASS COMPLETED

Rows
CLASS: (All) All Classes Informatics Mathematics Databases Programming Languages Algebra Geometry

Slicer
STUDENT: Konstantina Sarapi

	Measures			
CLASS	GRADES	HOMEWORK GRADES	TOTAL GRADES	CLASS COMPLETED
-All Classes	4	4	4	false
-Informatics	4	4	4	false
-Databases	5	4	4.6	false
Advanced Databases	5	6	5.4	true
Intro to Databases	5	3	4.2	false
Level 2 Databases	4	4	4	false
-Programming Languages	4	4	4	false
Advanced JAVA	5	5	5	true
Intro to JAVA	3	4	3.4	false
-Mathematics	3	5	3.8	false
-Algebra	4	5	4.4	false
Boolean Algebra	4	7	5.2	true
Linear Algebra	4	4	4	false
-Geometry	3	4	3.4	false
Differential Geometry	3	3	3	false
Topology and Geometry	3	6	4.2	false

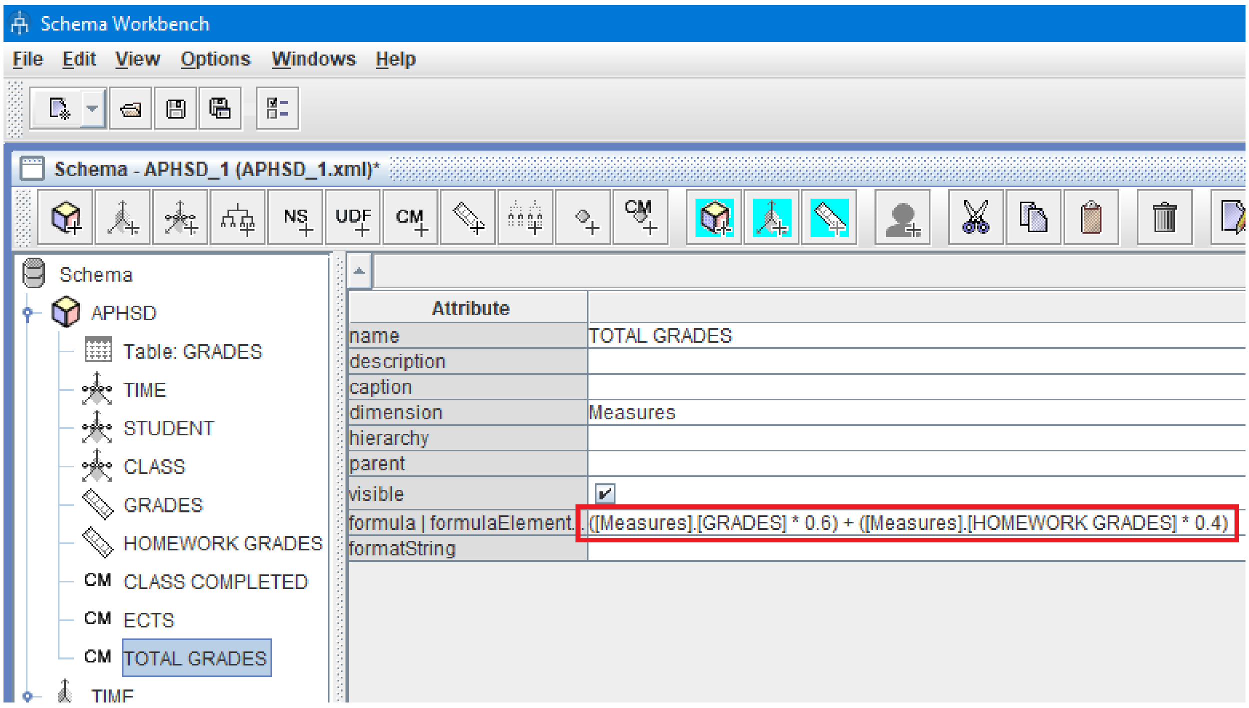
Figure 5-10 GRADES (Exam, Homework, Total) CLASS COMPLETED sliced for one Student

Measures GRADES and HOMEWORK GRADES are consider as "interfaced" measures, which means that their values are fetched from the database and more specific from the table GRADES which contains the values of both exam grades and homework grades (refer to Figure 5-1 Grades Star Schema).

On the other hand, measures TOTAL GRADES and CLASS COMPLETED are considered as "calculated" measure that use the values of the "interfaced" measures to perform a calculation.

More specifically, TOTAL GRADES is the sum of the GRADES times the exam "weight" plus the HOMEWORK GRADES times the homework "weight".

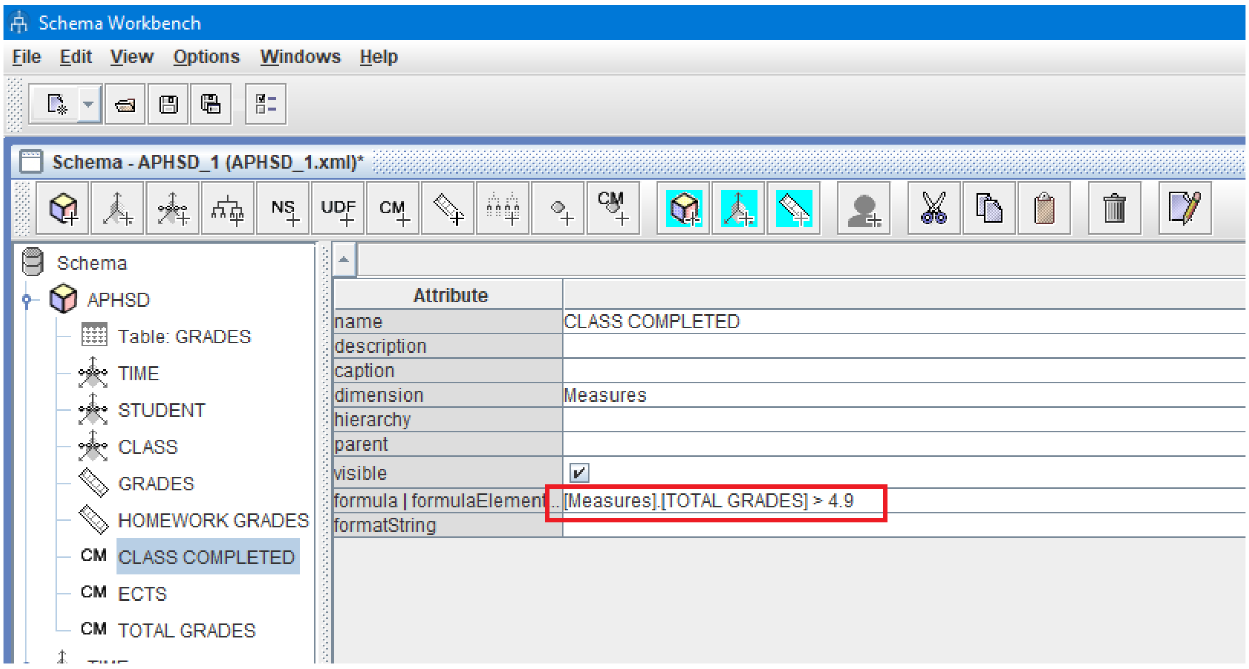
(TOTAL GRADES = GRADES * 0.6 + HOMEWORK GRADES * 0.4)



Attribute	
name	TOTAL GRADES
description	
caption	
dimension	Measures
hierarchy	
parent	
visible	✓
formula | formulaElement	([Measures].[GRADES] * 0.6) + ([Measures].[HOMEWORK GRADES] * 0.4)
formatString	

Figure 5-11 TOTAL GRADES calculation

Similarly, the CLASS COMPLETED is a Boolean calculated measure which is true in case the total grades are greater than 4.9.



Attribute	
name	CLASS COMPLETED
description	
caption	
dimension	Measures
hierarchy	
parent	
visible	✓
formula | formulaElement	[Measures].[TOTAL GRADES] > 4.9
formatString	

Figure 5-12 CLASS COMPLETED Calculation

Another remarkable feature of Xmondrian is the timeseries chart. In that chart, the behavior of an element (e.g. a student) can be observed through time and lead to interesting findings. In the following timeseries charts are analyzed and compared the grades of 4 students in 3 classes.

It can be observed that for the first student #3 (i.e "Konstantinos Seferidis) the performance in the exams follows a descending course in 2017 and improves in 2018 whereas the performance on exams of student #4 (i.e. "Naysika Kesinidou") shows a significant improvement after the first semester of 2017.

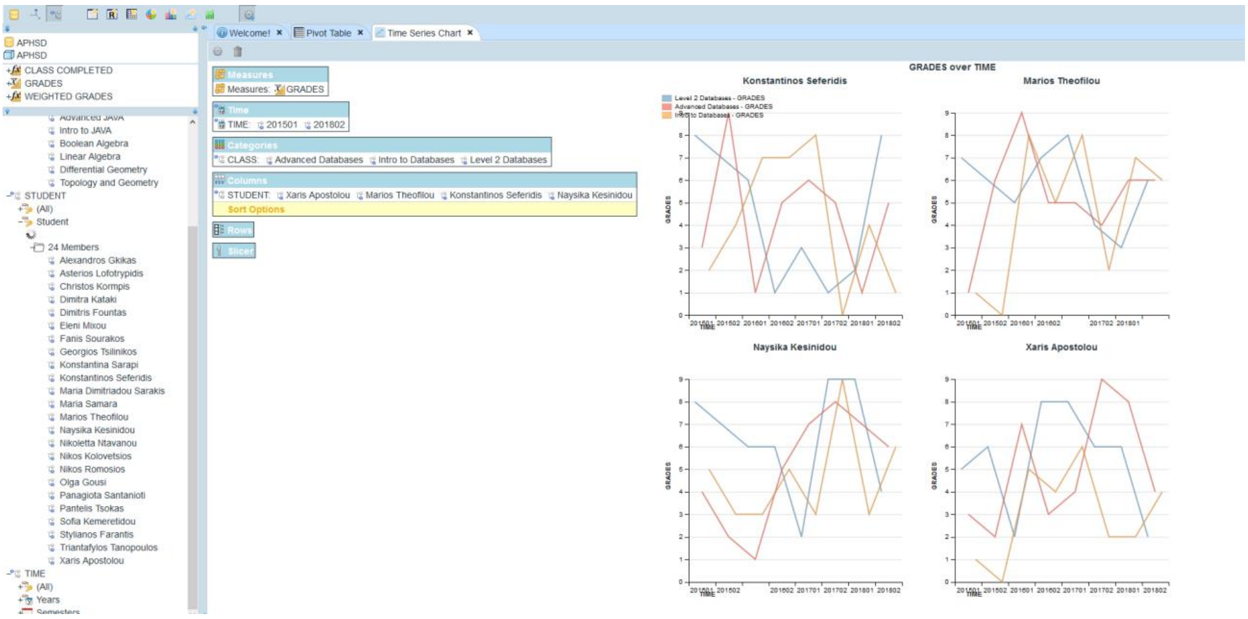

Measures: GRADES
TIME: 201501 201802
CLASS: Advanced Databases Intro to Databases Level 2 Databases
STUDENT: Xaris Apostolou Marios Theofilou Konstantinos Seferidis Naysika Kesinidou

Figure 5-13 Timeseries chart - Grades of 4 students in 3 classes

Another case could be the analysis of the grades of all students for each Class category from 2015 to 2018 as a timeseries chart.

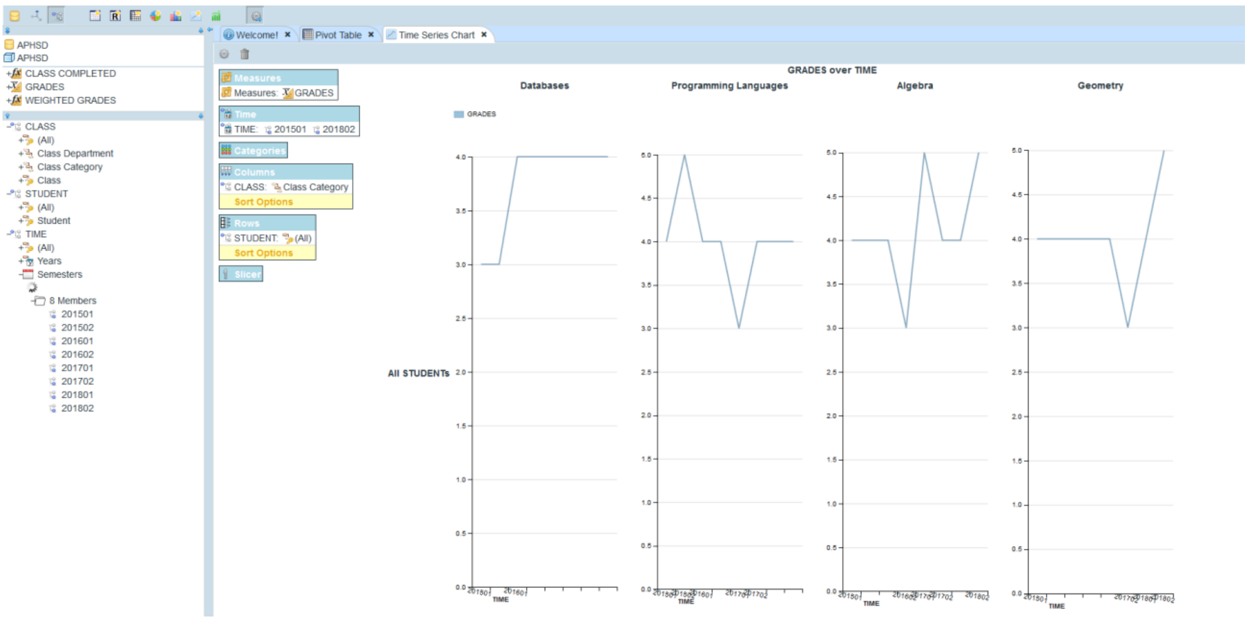


Figure 5-14 Grades of all 4 class categories over time

Note: The timeseries charts is a quite useful tool especially when it comes to Forecasting systems in Retail industry. By using timeseries, the historical sales of products can be studied and analyzed not only from the users but from the various forecasting algorithms as well which use that kind of information to predict the sales` demand for a certain period of time in the future.

In general, Xmondrian provides a basic toolset yet quite capable to perform OLAP analysis in large volumes of data. Since it’s easy to deploy, after it’s connected to the database and the schema is defined and configured, it is ready to be used and provide solutions and assist decision making in various environments.

5.2 OLAP in Retail - Oracle RPAS

Retail Predictive Application Server is the platform that uses OLAP analysis to develop schemas (or as Oracle calls them “solutions”) that are used to automate, optimize or replace specific business processes of retailers.

RPAS can store the data either in datastores based on Oracle’s Berkley DB which a software library that offers a high-performance embedded database.

The main difference with Pentaho’s Mondrian is that it does not require a connection with a database to configure the solution blueprint. The values of the measures are loaded in csv files which structure and format are defined in solution’s XML.

The solution blueprints are created in Oracle's schema creation tool, the Configuration Tools. It provides an easy and structured way to create the solution's blueprint. In other words, it provides a user-friendly environment to create and manipulate the XML files that are required from RPAS to represent the data and the ways that OLAP analysis will take place.

The following screenshot shows the user interface of the Configuration Tools.

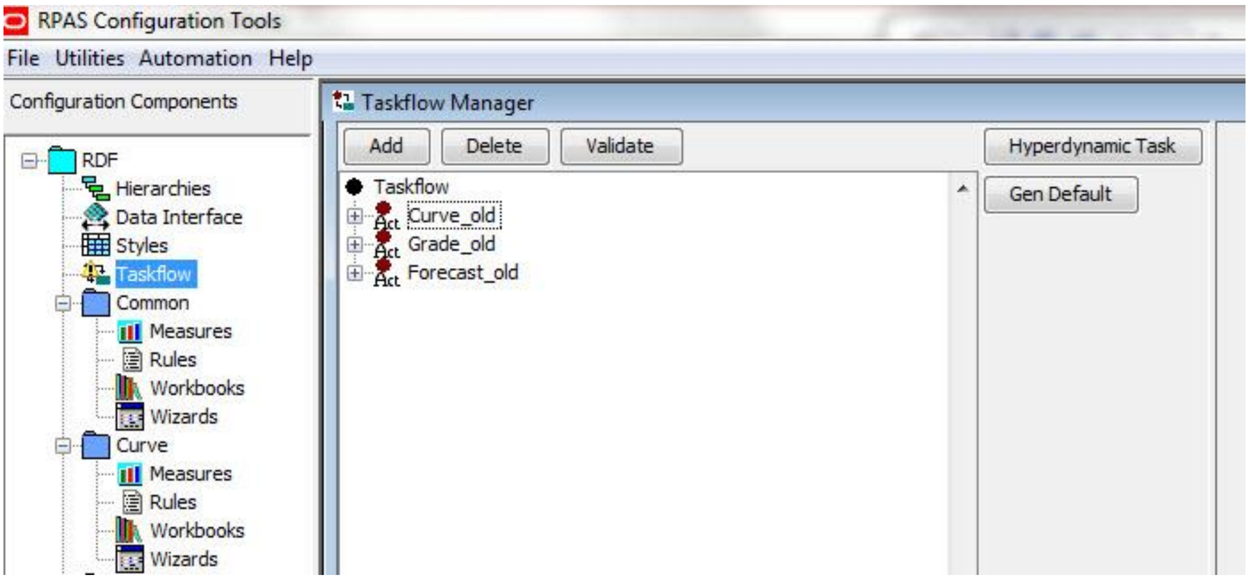


Figure 5-15 RPAS Configuration Tools.

Source: https://docs.oracle.com/cd/E12475_01/rdf/pdf/1412/html/ig/rdf-ig-apx-patch_rdf.htm

RDF: The name of the project that may be consisted of multiple solution. Here, RDF stands for Retail Demand Forecasting which is an Oracle's application which uses historical sales data to predict the sales demand for a specific -user defined- forecast horizon.

Hierarchies: The section in which the hierarchies are defined

Data Interface: The user defines the measures that will be loaded in the system through either csv or fixed-width files. In addition, it is defined the intersection in which the data will be loaded.

Styles: The various styles of measures, wizards etc can be configured here

Taskflow: In this section, the user can configure the order of the workbooks in the home page of Fusion Client.

Common: This is the name of one of the solution of a project. Here are configured all the measures that will be used, the calculations between the rules and the workbooks in which the data analysis will be performed.

Measures: All the measures of the solution are configured here where the user sets the name, the label, the aggregation/spreading type and other measure properties.

Rules: Here are defined any rules between the measures. A measure can be either loaded in the system or calculated using other measures. The rules that are defined here are for the measures that need to be calculated in real-time when a workbook is open. Measures that are calculated in batch scripts only require a “load” rule in order to fetch the values of this measure from the database to the workbook. On the other hand, “writable” measures need a “commit” rule to send the value that was set by the user in the database.

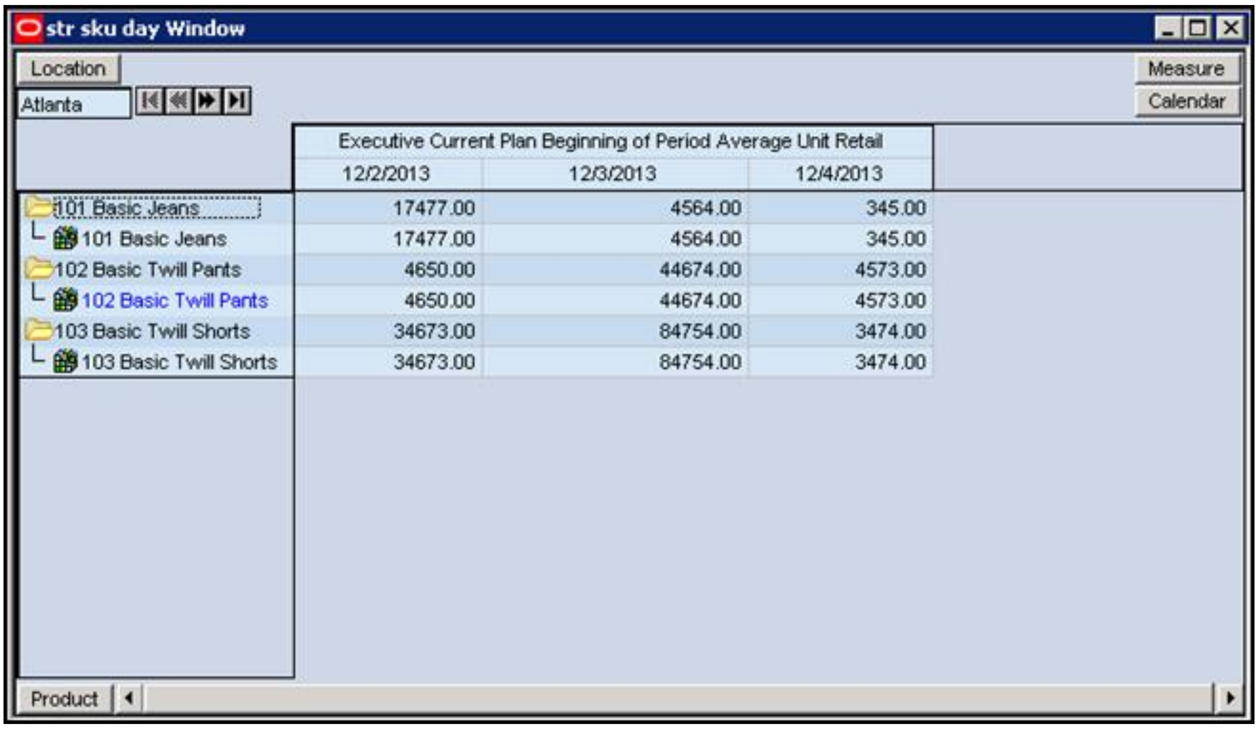
Workbooks: A workbook can be considered as a smaller instance of a cube that is used to fetch a small portion of the data for performance reasons. The workbooks in which the measures will be present are configured in this section. Each workbook contains worksheets in which the measures are added. Worksheets are actually the user interface in which the users are working and analyzing the data.

Wizards: The data that will be fetched when opening a workbook can be setup here. The user has the ability to chose which positions of the hierarchies will be loaded in the workbooks instead of the whole cube.

After the solutions’ blueprint is ready, the configuration is deployed, the interfaced measures are loaded and the data is ready to be used for analysis by the RPAS client.

Oracle sells RPAS in two different “clients”.

Classic Client – which requires local installation of the platform in order to use it.

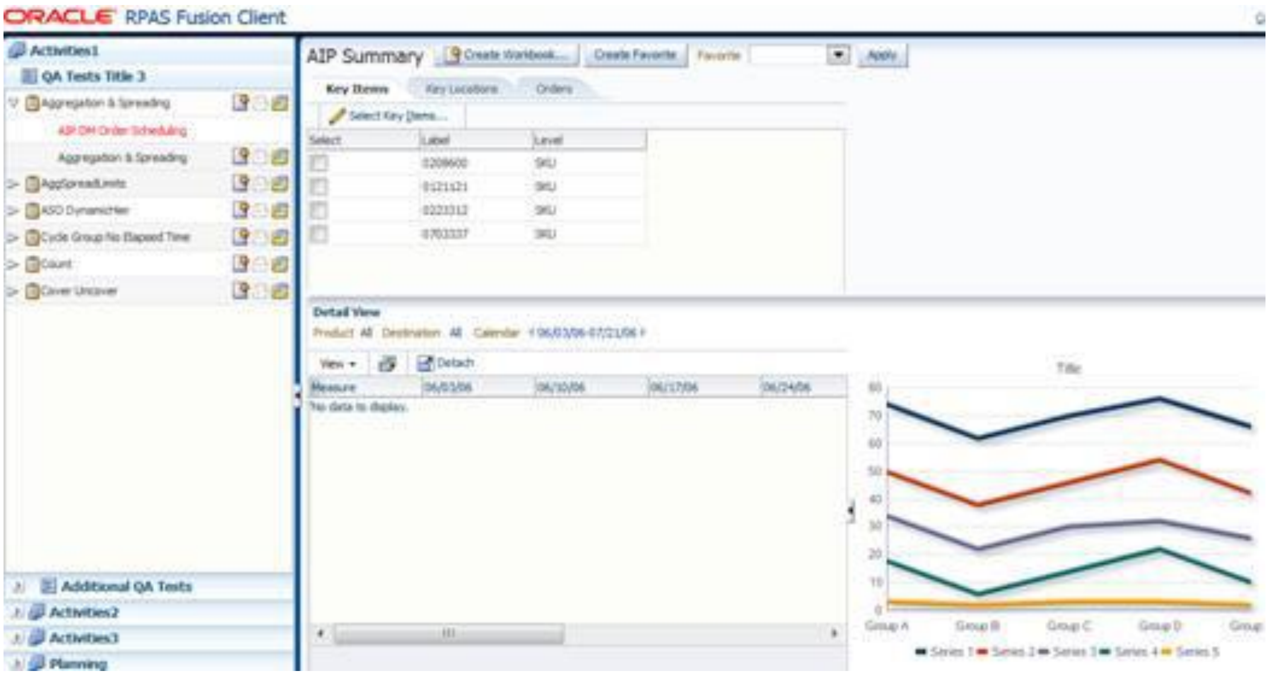


	Executive Current Plan Beginning of Period Average Unit Retail		
	12/2/2013	12/3/2013	12/4/2013
101 Basic Jeans	17477.00	4564.00	345.00
101 Basic Jeans	17477.00	4564.00	345.00
102 Basic Twill Pants	4650.00	44674.00	4573.00
102 Basic Twill Pants	4650.00	44674.00	4573.00
103 Basic Twill Shorts	34673.00	84754.00	3474.00
103 Basic Twill Shorts	34673.00	84754.00	3474.00

Figure 5-16 Example of Classic Client.

Source:
https://docs.oracle.com/cd/E12478_01/rpas/pdf/150/html/classic_client_user_guide/dynamic_position_maintenance_(dynamic_add).htm

Fusion Client – which uses the cloud and doesn't require local installation. Weblogic and Fusion Client are installed in a remote server and the users can login to RPAS through browsers like Mozilla Firefox or Google Chrome.



Select	Label	Level
☐	0208600	SKU
☐	0121121	SKU
☐	0223312	SKU
☐	0703337	SKU

Product All Destination All Calendar ‹ 06/03/06-07/21/06 ›

Measure	06/03/06	06/10/06	06/17/06	06/24/06
No data to display.				

Figure 5-17 Example of Fusion Client.

Source:https://docs.oracle.com/cd/E12478_01/rpas/pdf/150/html/fusion_client_user_guide/gettingstarted.htm

6 Conclusions

Despite the fact that OLAP analysis is present for quite a long time, it’s benefits and capabilities are still used when it comes to analyze large volumes of data. Decision support, timeseries analysis, configurable dashboards and the other capabilities of OLAP engines can be used in various fields and not only retail.

In cases where the data follows exponential growth, OLAP Engines can still provide solutions for the users to satisfy their need to gain deeper understanding of the various working processes which leads to more accurate and efficient decision making.

7 Improvement Proposal

In the scope of this thesis, only a small portion of Mondrian OLAP Engine’s capabilities has been presented and dummy data was used.

In University of Macedonia, the open-source Mondrian OLAP Engine could be integrated with the existing systems such as the databases where the data of the Students, Classes etc is stored and a more sophisticated schema can be created to integrate the various databases that may exist.

The requirements for such project could be:

- Mondrian can connect at multiple databases at the same time and thus the integration with the other databases could be a seemingly easy task
- Joint session between the involved members to define and design the needs
- A schema XML to interpret the aforementioned design of the desired analysis which can be created in Mondrian Schema Workbench tool
- Documentation of the whole process in an auditable and “repeatable” way

A system like Mondrian will allow Professors and Administrative officers to perform analysis in various aspects of the academical everyday life, from analyzing the grades of the active students to even analyze data and statistics of the graduated students and how the university degrees were depicted in their professional career.

This could be achieved by integrating Xmondrian as an addon of the already implemented system of the university, CoMPUs.

This document could be also used as a “manual” to provide the required information of how Xmondrian can be deployed and used.

8 Web Pages Visited

http://rpbouman.blogspot.com/2016/03/need-mondrian-war-checkout-xmondrian.html

http://hsqldb.org/index.html

http://hsqldb.org/doc/util-guide/dbm-chapt.html

https://orangeslate.com/2006/11/21/hsqldb-a-lightweighted-relational-database/

http://tomcat.apache.org/

https://erdplus.com/#/standalone

https://mondrian.pentaho.com/documentation/workbench.php

https://tomcat.apache.org/tomcat-9.0-doc/index.html

https://docs.microsoft.com/en-us/sql/analysis-services/multidimensional-models/mdx/mdx-query-fundamentals-analysis-services?view=sql-server-2017

9 Table of Figures

Figure 2-1 Example of an OLAP Cube 6
Figure 2-2 Example of Hierarchies in Oracle's RPAS Configuration Tools........ 7
Figure 2-3 Example of hierarchical relationship........ 8
Figure 2-4 Example of the different aggregation types 9
Figure 2-5 Example of different Spreading Types........ 10
Figure 2-6 Data flow from OLTP to OLAP. 11
Figure 2-7 Differences between OLTP and OLAP. *Source: http://datawarehouse4u.info/OLTP-vs-OLAP.html*........ 12
Figure 2-8 MOLAP Architecture........ 13
Figure 2-9 ROLAP Architecture........ 14
Figure 2-10 Differences between MOLAP and ROLAP. 15
Figure 2-11 HOLAP Architecture........ 16
Figure 3-1 Apache Tomcat where Xmondrian was deployed........ 17
Figure 3-2 Mondrian Schema Workbench 18
Figure 3-3 Database Connection........ 18
Figure 3-4 HSQL Database Manager........ 20
Figure 3-5 MDX Query example........ 20
Figure 3-6 Drag'n'drop functionality of Xavier 22
Figure 4-1 Sales per store in January........ 23
Figure 4-2 Average Sales per country........ 24
Figure 4-3 Sum of sales per Product class 25
Figure 5-1 Grades Star Schema........ 26
Figure 5-2 HSQLDB Manager 27
Figure 5-3 datasources.xml........ 27
Figure 5-4 HSQLDB database file 28
Figure 5-5 APHSD_1.xml schema........ 28
Figure 5-6 MDX front-end........ 29
Figure 5-7 MDX example back-end........ 30
Figure 5-8 Default view of Xmondrian........ 30
Figure 5-9 The grades of all Students in Class "Advanced Databases" for the first Semester of 2017........ 32
Figure 5-10 GRADES (Exam, Homework, Total) CLASS COMPLETED sliced for one Student 33

Figure 5-11 TOTAL GRADES calculation 34
Figure 5-12 CLASS COMPLETED Calculation 34
Figure 5-13 Timeseries chart - Grades of 4 students in 3 classes 35
Figure 5-14 Grades of all 4 class categories over time 36
Figure 5-15 RPAS Configuration Tools 37
Figure 5-16 Example of Classic Client. 38
Figure 5-17 Example of Fusion Client 39