SOLAP_Frame: A Framework for SOLAP using Heterogeneous Data Sources

Tiago Eduardo da Silva
Federal Institute of Education, Science and Technology of Pernambuco, Palmares, Brazil
e-mail: tiagoes@gmail.com

Daniel Farias Batista Leite, Cláudio de Souza Baptista
Federal University of Campina Grande, Campina Grande, Paraíba, Brazil
e-mail: danielfarias@copin.ufcg.edu.br, baptista@computacao.ufcg.edu.br

***Abstract*—Business Intelligence (BI) and Geographic Information System (GIS) technologies have been used by several organizations aiming to improve the decision making process. In this context, the term Spatial OLAP (SOLAP) emerged for spatial multidimensional data sources. Nevertheless, it is very complex to integrate heterogeneous data sources into a SOLAP solution. This paper proposes a framework that enables the analysis of multidimensional spatial data from heterogeneous data sources. This goal is accomplished through an implementation of a SOLAP framework that contains interfaces and abstract classes that can be implemented and extended to support new data sources. To validate the proposed ideas, a case study was conducted.**

Keywords- Business Intellignce; GIS; SOLAP.

I. INTRODUCTION

With the increasing volume of data coming from a large variety of sources, there has been a considerable increase in investments on technologies capable of extracting information from these data and, consequently, help managers in the decision making process. *Business Intelligence* (BI) tools provide a historical, updated and predictive view of business operations of a company, enabling the identification of patterns, the availability of new functionalities and products, and improving the relationship with costumers. *On-line Analytical Processing* (OLAP) is one of the most used BI tools. An OLAP tool enables rapid exploration and analysis of data stored in multiple aggregation levels, according to the multidimensional approach. In this context, most companies are adopting BI tools in order to become more competitive in the marketplace [1].

In addition to this, most companies heavily deal with the spatial dimension in their datasets. Hence, it is important to investigate how to explore that dimension in order to improve the decision making process. However, traditional BI technologies do not take advantage of spatial data. On the other hand, Geographical Information Systems (GIS) were designed to work on georeferenced data using the Online Transaction Processing (OLTP) approach, and thereby prevents an efficient and deep data analysis.

More recently, corporations have demanded the integration of GIS and OLAP technologies to give rise to a new category of tools known as *Spatial Online Analytical Processing* (SOLAP). The integration of these technologies may happen through three distinct approaches: prioritizing the resources of GIS (GIS-dominant), overlapping visual and graphic resources of OLAP tools (OLAP-dominant), and the full integration approach (SOLAP) that aggregates the functionalities of GIS with graphs, tables and maps [2].

The need for integrating GIS and OLAP technologies has driven a set of new SOLAP academic solutions. Nonetheless, no consensus was reached as to the best way to achieve this integration. The solutions differ in several respects, particularly for the data model. Without a consensus on the data model, it is very hard to provide spatial cubes on the Web.

The lack of consensus for GIS - OLAP integration and standards for the provision of spatial and multidimensional data hinders the use of different data sources at the same time. Hence, the extraction of useful information to improve the decision making process at corporations is impaired.

This paper proposes a new framework that enables the analysis of spatial cubes coming from multiple and heterogeneous multidimensional data sources. Our proposed framework can be classified as an Enterprise Application Framework, as it is concerned with OLAP domain [3]. According to Sommerville, a framework is a software that can be extended to create a more specific application [4].

The main contributions of our research is the proposal of a SOLAP framework with interfaces and abstract classes that can be implemented and extended to support new data sources, making easy the integration of heterogeneous data cubes. Hence, we offer a reusable software to interoperate spatial data warehouses from heterogeneous data sources. To the best of our knowledge, this is the first study on this question.

Furthermore, in order to validate the proposed framework, we present a case study on the accountability analysis of the TCE-AC (Court of Accounts of the State of Acre – Brazil). In the case study, we extended our SOLAP framework to access cubes coming from two different sources: *Microsoft SQL Server Analysis Services* (SSAS) and *GeoMondrian*.

The rest of this paper is organized as follows. Section II discusses related work on SOLAP. Section III addresses the architecture of the proposed framework. Section IV presents a case study involving the Court of Accounts of the State of Acre – Brazil. Finally, Section V highlights the conclusions and further work to be undertaken.

 ISBN: 978-1-61208-469-5

II. RELATED WORK

Spatial OLAP has been a very active research area for a long time. Surveys on SOLAP can be found in [5][6][7]. Salehi et al. propose a formal model for spatial datacubes [8]. Aguila et al. address a conceptual model for SOLAP [9]. Baltzer focuses on spatial multidimensional querying [10]. Glorio and Trujillo highlight the optimization of spatial queries [11]. Tahar Ziouel et al. propose an approach for cartographic generalization of SOLAP applications [12].

Several SOLAP tools have been developed over the last years in many contexts. Rivest et al. propose a generic SOLAP tool, called JMap Spatial OLAP, comprising two modules: an administrative one, allowing for the setup of the connection with a multidimensional spatial database; and a visualization module, allowing for the interactive exploration of data through charts and maps [2]. The proposed tools are based on Relational OLAP (ROLAP) architecture and support the three types of spatial dimensions: geometric, non-geometric and mixed.

Bimonte et al. developed the GeWOlap SOLAP tool, highlighting the synchronization of different forms of data visualization. Their architecture comprises three layers: data, SOLAP server and client layers (user interface) [13].

Escribano et al. proposed a tool called Piet, integrating GIS and OLAP technologies and executing the precomputation of the map layers [14]. The Piet architecture also comprises three layers: data, SOLAP server and client layers.

Another challenge faced in SOLAP solutions is the issue of aggregation performance when queries involve considerable amounts of spatial data. Jiyuan Li et al. combine SOLAP approach with the Map-Reduce model for processing large amounts of data in parallel [15]. Saida Aissi et al. propose a multidimensional query recommendation system aiming to help users to retrieve relevant information through SOLAP, improving the data exploitation process [16].

The integration of the GIS and OLAP technologies was also explored through data communication techniques, with the objective of enabling their interoperability. Silva et al. proposed a Web Service called GMLA WS, which combines XML for Analysis (XMLA), Geography Markup Language (GML) and Web Feature Service (WFS) [17].

Dubé et al. presented an XML format to supply and exchange SOLAP cubes through web services [18]. The proposed XML format does not depend on the OLAP/GIS tool and represents all the necessary data (facts and members) and metadata (scheme), besides supporting spatial dimensions and members. The advantage of exchanging data through Web Services is that the communication is not limited to traditional client-server platforms, but also supports ubiquitous mobile computing environments.

These solutions differ in the data model and there is no standard for provision and analysis of spatial data. In this perspective, the Open Geospatial Consortium (OGC) published in 2012 a report (white paper) containing an evaluation of the ways that the OGC standards (e.g., WMS - Web Map Service, WFS, WPS - Web Processing Service, etc.) could be extended, in order to promote the use of geospatial information and the interoperability of GeoBI applications. However, neither extension nor standard for this purpose has been published by OGC yet.

We observed that the key features to compare SOLAP solutions are: (I) to enable the creation of queries through a visual query language; (II) to provide an integrated view of both multi-dimensional and spatial data; (III) to support spatial operators allowing for more comprehensive analysis; (IV) to give access to various multidimensional data sources (cube servers); (V) to enable geocoding data to provide spatial analysis in pure OLAP sources; (VI) to use open technologies to reduce costs; (VII) to be extensible so that new features can be added; and (VIII) to enable data visualization through maps, tables and graphs. Table I presents a comparison among the related work in which the cells that contain an X mean that a given solution implements a given feature and those that contain a – mean that a given solution does not implement a particular feature.

This paper presents a SOLAP framework, known as SOLAP_Frame that enables the connection to multiple and heterogeneous data sources. The framework was developed using open technologies. Furthermore, the proposed framework presents an integrated visualization of multidimensional and spatial data, allowing for the creation of queries by means of a visual specification language with support to spatial operators and data visualization through maps, tables and charts. Finally, SOLAP_Frame also enables the geocodification of the data and is extensible, providing support for addition of new functionalities, such as new operators or data visualization methods. To the best of our knowledge, this is the first work to propose a SOLAP framework that is able to interoperate with new heterogeneous data sources.

TABLE I. RELATED WORK COMPARISON

Solutions	Features							
	I	*II*	*III*	*IV*	*V*	*VI*	*VII*	*VIII*
JMAP [2]	X	X	-	-	-	X	X	X
GeoWOLAP [11]	X	X	-	-	-	-	-	-
Piet [12]	-	-	-	-	-	-	-	-
SOLAP_Frame	X	X	X	X	X	X	X	X

III. SOLAP_FRAME: A FRAMEWORK FOR SPATIAL ANALYSIS USING HETEORGENEOUS DATA SOURCES

This section presents the SOLAP_Frame architecture. In the next subsections, we describe the architecture and the extension points of the proposed SOLAP framework.

A. Architecture

The framework architecture comprises three layers: client, application and data layers, as shown in Figure 1.

The client layer comprises a set of graphical Web interfaces in which the user can connect to a

multidimensional spatial cube, geocode members, compose queries by means of a visual specification and visualize the data.

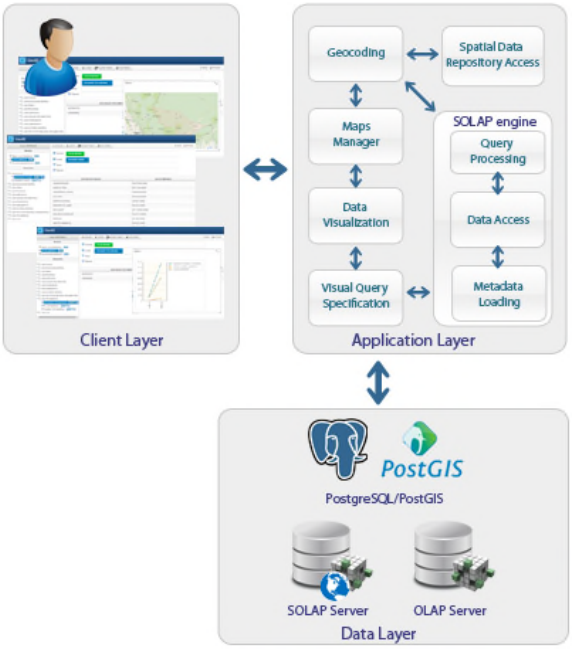

Figure 1. The SOLAP_Frame architecture.

The data layer comprises the multidimensional data sources to be analyzed and the spatial data repository. The framework is capable of accessing several multidimensional servers (cube servers), employing different technologies and manufacturers. The framework also supports the geocoding of cube members, enabling the spatial analysis of non spatial OLAP cubes. The application data repository is stored in the *PostgreSQL* DBMS, with the PostGIS spatial extension. The spatial data resulting from the geocoding of members of the cube are stored in this repository, characterizing a Data Warehouse federated approach. Any spatial DBMS can be used for this purpose, by simply extending the proposed solution through its extension points.

The application layer is responsible for the implementation of the whole application logic. This layer has six modules: visual query specification, data visualization, map manager, spatial data repository access and the SOLAP engine. We highlight the SOLAP engine as the main module of this layer, providing communication between the application and the multidimensional servers (OLAP or SOLAP) attached to the data layer.

The visual query specification module controls the query execution and results visualization, turning the interactions between users and the graphical interface into objects that compose the query visual specification. After receiving the result of a visual query, the visual specification module forwards the result with its markup to the data visualization module so that the data can be transformed and presented in the specified format. Depending on the markup type, data may have to be transformed, for example, grouped to compose the map layers or graph axes. After being transformed, data are forwarded to the most appropriate component of the interface for visualization (e.g., tables, maps, charts, text and caption).

The map management module is responsible for displaying data in maps. As such, this module receives, from the data visualization module, a set of spatial and numerical data, query results, and the markup. The repository access module, in turn, was implemented to retrieve the metadata and the data from the spatial tables stored in the spatial data repository. The metadata and the data on the spatial tables are used by the geocoding module, which is accessed using the Java Database Connectivity (JDBC) driver for the PostgreSQL database management system (DBMS) with the PostGIS spatial extension.

The SOLAP engine comprises of three sub-modules: data access, metadata loading and query processing. This engine is responsible for: implementing the connection to a given multidimensional data source; loading of metadata from cubes to be analyzed; translating the visual specification to the destination query language; submitting the translated query; and receiving the result data.

The implementation of the SOLAP engine depends on the manufacturer of the SOLAP server to be accessed.

The data access module enables the connection to the multidimensional data source and the choice of the cube to be analyzed. This module is also in charge of executing queries in the language of the accessed technology and returning the results of these queries. To accomplish the data access module, it is necessary to have information on the connection properties that match both source and cube properties. The source properties state where and how to connect to the multidimensional data source, while the cube properties address which cube belonging to a given source should be accessed. The data access module knows how to handle heterogeneous sources.

The metadata loading and the query processing modules of the SOLAP engine interact with the data access module, which is specialized, that is, its implementation depends on the adopted technology.

The metadata loading module is responsible for retrieving cube metadata. In order to connect to a multidimensional data source, the *ConnectionProperties* and *DataSource* objects must be informed. The metadata coming from this connection will be turned into Cube objects, which will be loaded into memory for subsequent use by other modules of the proposed framework.

The query processing module is responsible for translating the visual queries to queries in the target technology native language; executing them using the data access module; and returning the query. In order to retrieve the data, besides the connection properties, a visual query is passed as parameter to the processing query module.

B. SOLAP_Frame: Extension Points

SOLAP_Frame contains extension points that enable to connect it to heterogeneous data sources. In this section, we provide details of the communication interfaces.

 ISBN: 978-1-61208-469-5

B.1 SOLAP Engine Facade

The main extension point of the proposed framework is the implementation of the SOLAP engine facade. This facade is responsible for the communication between the proposed solution and the multidimensional data source. The message exchanges between the solution and the SOLAP engine consist of requesting metadata and data from a cube. The facade standardizes this message exchange. To request metadata from a cube to the SOLAP engine, the facade contains the *loadCube* method that receives as parameters the connection properties of both the data source and cube and returns an object that represents the cube.

To request data from a cube to the SOLAP engine, the facade contains three methods: *processQuery*, *getLevelMembers* and *filterLevelMembers*. The *processQuery* method receives as parameter an object that models the query, which is part of the visual specification defined by the user. This visual query should be translated into the query language of the source technology; and then executed. The query result must be modeled in a return object called *VisualQueryResult*.

The *getLevelMembers* method receives as parameter an object that represents a hierarchical level. This level is used to retrieve the members of the cube. Finally, the *filterLevelMembers* method receives as parameter, besides the level, a filter that can be either conventional or spatial. This filter will be used to select the members to be retrieved.

The solution will automatically identify the implementation of the front end by means of the Contexts and *Dependency Injection services* (CDI) present in the Java Enterprise Edition platform, and will register it for use. A name and a type must be associated with the SOLAP engine in order to be presented to the user. The type is used by the BI engine manager keeping the mapping between types and implementations available in the solution.

B.2 Connection properties interface

The communication process requires that the user provides the connection properties. This information will be used every time the SOLAP engine needs to communicate with the data source. Thus, another extension point in our framework is the implementation of this user interface.

The connection properties depend on the technology to be used. Hence, the parameters that must be provided vary according to the technology.

The front end for the SOLAP engine contains a method called *getLoaderPopup*, which returns an object called *LoaderPopup*, which, in turn, contains the necessary information for the exhibition of the component. This object is used by the interface that lists all the available engines in the solution. The *LoaderPopup* object is formed by another object called *LoaderBean*, that needs to be implemented. The *LoaderBean* is the controller responsible for preparing the component for exhibition and for enabling access to the properties of connections created by the user. Figure 2 presents a class diagram for the *LoaderPopup* and *LoaderBean* object.

B.3 SOLAP engine for XMLA server

In order to provide access to several heterogeneous multidimensional data sources, we also developed a SOLAP engine for servers that provide their data through the XMLA protocol. To enable the XMLA engine to access a specific technology, it is necessary to implement the abstract classes described in the following. In the implementation of the SOLAP engine for XMLA, we used the XMLA driver supplied by the Open Java API for OLAP (olap4j), which is also an open specification for the construction of OLAP applications based on the JDBC specification. Once connected to the data source, the user chooses the cube that will be analyzed. After that, an alias is assigned to the cube. This alias will be used to identify the cube in the system.

After the selection of the cube, its metadata will be loaded. For this, it is necessary to convert the metadata from the native format into the target one. The metadata loading is carried out by the *Olap4jXMLACubeMetadataDAO* class, and the abstract class *AbstractOlap4jXMLACubeConverter* implements the basic methods necessary for the conversion of the cubes from the native format to the format used in the solution.

The methods of the *AbstractOlap4jXMLACubeConverter* abstract class are spatially related and depend on the technology used by the XMLA server. This is due to the fact that XMLA does not specify a standard format for the transportation of spatial data. Furthermore, the Multidimensional Expressions (MDX) query language, used by the XMLA server, does not specify spatial functions. Figure 2 presents a class diagram for the XMLA engine.

To load the data, the *AbstractOlap4jXMLAQueryDAO* class supplies the basic functionalities necessary for the correct operation of the solution. However, it is necessary to implement the method responsible for translating MDX filters into the language for the chosen technology. The abstract method is necessary due to the fact that the solution has filters that use spatial functions. Since these functions are not standardized for MDX, they vary according to the technology used.

IV. CASE STUDY APPLIED TO THE COURT OF ACCOUNTS OF THE STATE OF ACRE - BRAZIL

In order to evaluate the SOLAP_Frame, we ran a case study on public accountability of the Court of Accounts of the State of Acre – Brazil (TCE-AC). The aim of this case study was to help in the decision making process related to the definition of more efficient management strategies to achieve an effective control of public spending.

To run this case study, the framework was extended to connect to two multidimensional data sources: *SQL Server Analysis Services* and *GeoMondrian*, of which the first one provides access to conventional data, and the second one provides access to spatial data. Three cubes are available for the analysis: Commitment, Liquidation and Payment. The Commitment cube uses the opensource *GeoMondrian* server, while the other ones utilize the Microsoft SSAS.

 ISBN: 978-1-61208-469-5

Figure 2. Class Diagram for XMLA Engine.

The fact tables to be analyzed are represented by the measures: Commitment values, Liquidation values and Payment values. Besides the measures modeled in the *Spatial Data Warehouse* (SDW), the cubes have two additional measures: number and mean of the values.

In this context, we highlight two queries aiming at validating the proposed framework with respect to the simultaneous access to several multidimensional data sources. Moreover, the queries proposed enable to address the functionalities related to the spatial analysis of multidimensional data. The two queries are:

- "Display a thematic map with the sum of the Commitment values for each city in the State of Acre – Brazil"; and
- "What is the sum of the liquidated values in the neighbor cities of Rio Branco city, concerning the functions Administration, Agriculture and Legislative?"

In order to solve the first query, the Commitment cube was used. The Commitment Value measure was added to the tab "Columns", while the spatial hierarchy District Name was added to the tab "Layers". The caption, created for the Commitment Value measure, was used to visualize the data on the map. The District Name level was used as a label for the geometries. Figure 3 presents the result of the query.

The second query demonstrates the use of the spatial filters available in the framework. In this example, the cube Liquidation was used; the Liquidation Value metric was added to the tab "Columns", the hierarchy Function Description was added to the tab "Rows", and the hierarchy District Name, to the tab "Layers". The members of the Function Description level were filtered. A geographic filter was added to the districts, and the spatial operator *Touches* was used to filter neighbor cities of Rio Branco municipality. To visualize the data, we used a caption in spatial panels. Figure 4 presents the result of the query.

V. CONCLUSIONS AND FUTURE WORK

Currently, there is a demand for exploring spatial data sources to improve the decision making process. However, no consensus has been reached yet regarding the best way to accomplish this integration. This lack of standard makes it hard to analyze spatial cubes from heterogeneous multidimensional data sources.

From the survey of the state of the art, it was possible to compare the strengths and weaknesses of the main existing solutions and conclude that these solutions do not provide analysis of spatial cubes from heterogeneous multidimensional data sources.

This paper presented a framework that executes spatial data analysis from heterogeneous multidimensional data sources. The framework provides interfaces and abstract classes that can be extended to incorporate new multidimensional data sources. Reusability is a main issue addressed by our framework concerning the implementation of SOLAP tools. Furthermore, by using geocoding operation, it is possible to have a SOLAP tool over a traditional OLAP one, which emphasizes the flexibility of the proposed framework.

To validate the proposed ideas, we conducted a case study in a Court of Accounts, in which where Microsoft SSAS and GeoMondrian cube servers were accessed through extensions of the proposed framework. This case study has enabled a concrete evaluation of the SOLAP functionality.

The spatial cube model of the proposed framework proved to be efficient, allowing the spatial analysis of cubes from multiple heterogeneous sources.

As a future work, the framework can be explored in various expansion points in order to make it more robust. For example, the incorporation of new SOLAP operators and support for faster data.

 ISBN: 978-1-61208-469-5

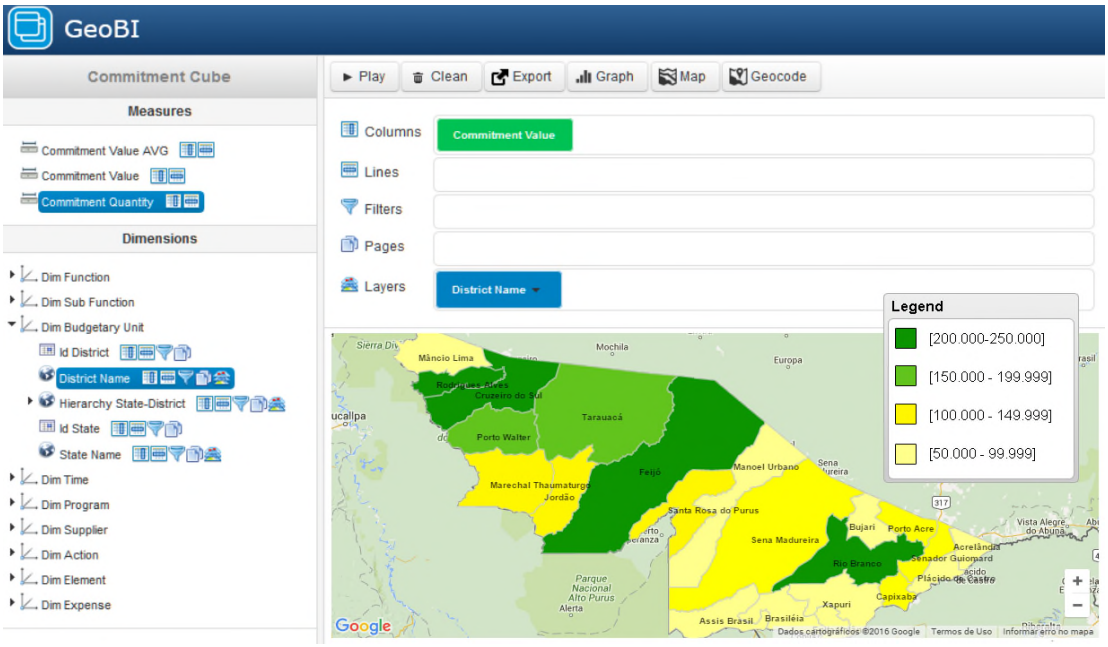

Figure 3. Query 1: result.

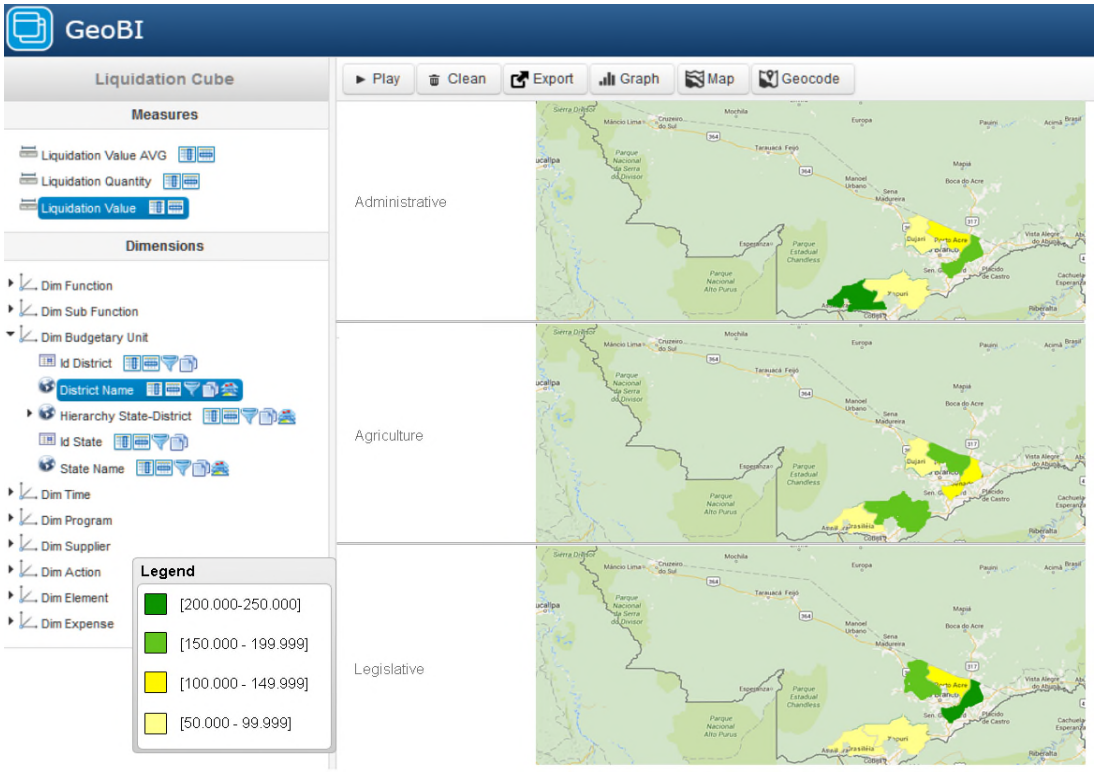

Figure 4. Query 2: result.

ACKNOWLEDGEMENTS

The authors thank the Court of Accounts of the State of Acre – Brazil (TCE-AC), the Brazilian ElectricityRegulatory Agency (ANEEL) and the National Council for Scientific and Technological Development (CNPq) for funding this research.

REFERENCES

[1] S. Chaudhuri, U. Dayal, and V. Narasayya, "An overview of business intelligence technology," Communications of the ACM, vol. 54, no. 8, 2011, pp. 88-98.

[2] S. Rivest, Y. Bédard, M. Proulx, F. Hubert, and J. Pastor, "SOLAP technology: Merging business intelligence with geospatial technology for interactive spatio-temporal exploration and analysos of data," ISPRS P&RS, vol. 60, no. 1, 2005, pp. 17-33.

[3] M. E. Fayad and D. C. Schmidt, "Object-oriented application frameworks", Communications of the ACM, vol. 40, no. 10, 1997, pp. 32-40.

[4] I. Sommerville, Software Engineering, 10th edition, 2015, Pearson.

[5] L. Gómez, B. Kuijipers, B. Moelans, and A. Vaisman, "A Survey of Spatio-Temporal Data Warehousing," JDWM, vol. 5, no. 3, 2009, pp. 28-55.

[6] S. Bimonte, A. Tchounikine, M. Miquel, and F. Pinet, "When Spatial Analysis Meets OLAP Multidimensional Model and Operator," JDWM, vol. 6, no. 4, 2010, pp. 33-60.-

[7] G. Viswanathan and M. Scheneider, "On the requirements for user-centric spatial data warehousing and SOLAP," in Proceedings of the 16th DASFAA, 2011, pp.144-155.

[8] M. Salehi, Y. Bédard, and S. Rivest, "A formal Conceptual Model and Definition Framework for Spatial Datacubes," Geomatica, vol. 64, no 3, 2010, pp. 313-326.

[9] P. Aguila, R. Fidalgo, and A. Mota, "Towards a more straightforward and more expressive metamodel for SDW modeling," in DOLAP 2011, pp. 31-36.

[10] O. Baltzer, "Computacional Methods for Spatial OLAP," Ph.D. thesis, 2011.

[11] O. Glorio and J. Trujillo, "Designing Data Warehouses for Geographic OLAP Querying by Using MDA," in Proceedings of the international Conference on Computational Science and its Applications, 2009, pp. 305-519.

[12] T. Ziouel, K. A. Derbal, and K. Boukhalfa. "SOLAP On-the-Fly Generalization Approach Based on Spatial Hierarchical Structures," CIIA 2015, pp. 279-290.

[13] S. Bimonte, A. Tchounikine, and M. Miquel, "Spatial OLAP: Open Issues and a Web Based Prototype," in M. Wachowicz & L. Bodum (Eds.), Proceedings of the 10th AGILE International Conference on Geographic Information Science, 2007.

[14] A. Escribano, L. Gomez, B. Kujipers, and A. Vaisman, "Piet: a GIS-OLAP implementation," in Proceedings of the ACM 1oth international workshop on Data Warehousing and OLAP, 2007, pp. 73-80.

[15] J. Li, L. Meng, F. Z. Wang, W. Zhang, and Y. Chai, "A Map-Reduce-enabled SOLAP cube for large-scale remotely sensed data aggregation," Computers & Geosciences, vol. 70, 2014, pp. 110-199.

[16] S. Aissi, M. S. Gouider, T. Sboui, and L. B. Said, "Personalized recommendation of SOLAP queries: theoretical framework and experimental evaluation," SAC 2015, pp. 1008-1014.

[17] J. Silva, V. Times, and A. Salgado, "An Open Source and Web Based Framework for Geographical and Multidimensional Processing," SAC 2006, pp. 63-67.

[18] E. Dubé, T. Badard, and Y. Bedard, "XML enconding and Web Services for Spatial OLAP data cube exchange: an SOA approach," CIT, vol. 17, no 4, 2009, pp. 347-358.

 ISBN: 978-1-61208-469-5