Evaluation of Federated Database for Distributed Applications in e-Government

Carmelo Pino, Salvatore Ravidà and Santo Scibilia
Department of Electrical, Electronics and Computer Engineering, University of Catania,
V.le Andrea Doria, 6 95125 Catania, Italy
{cpino, sravida, sscibilia}@dieei.unict.it

***Abstract*—The evolution of the distributed database technologies represents a great opportunity in each sector of public administration to improve the efficiency of e-government, since the distributed technologies improve one of the basic need of the Public Administration, that is the interoperability between their different units, that is strictly related to the ability of retrieving data located on different archives. As known, the federated database approach could represent a solution to data integration, but the existing solutions often don't take into account real time constraints that are more and more required by the modern e-government scenario. For this reason, in this paper a performance evaluation of relevant technologies implementing the federated database solution is done and compared with the performance of a software architecture consisting of a cloud of web services we have developed to support data integration for the administrative units of a municipal administration. The results confirm the low time performance of the federated data solutions when the integration deals with a large amount of data, whereas the proposed customizable solution shows appreciable high performance.**

***Index Terms*— Federated Database, E-Government, Federated MySql, OpenLink Virtuoso, Performance evaluation.**

I. Introduction

The enormous growth of citizen's data in a public administration requires an adequate use of technologies in order to retrieve the information quickly [1].

There exist several e-government applications and services that can be categorized at different levels: Government to Government (G2G), Government to Business (G2B), Government to Citizens (G2C) and Government to Enterprise (G2E). Each category is related to specific set of applications and the common need is the communication between multiple sources.

Usually, the different municipal administrative units (AUs) are not located in the same building, but are distributed in different buildings on the territory. Typically, to accomplish their tasks, such AUs have to query not only the data stored on their servers but also the data of other departments stored on the remote servers. The need of accessing frequently remote data may cause several difficulties since these data are often coded in different formats and managed with different systems.

One of the main approaches to solve these problems is the federated database approach [2] that has been widely adopted to join information resident on different archives located on separate machines. Indeed, a wide range of technologies have been proposed in the literature to federate distributed data bases, e.g., [3], [4], [5], aiming at reducing the maintenance costs and at satisfying data coherence requisites; also, the use of the cloud technology was more and more proposed to meet security, privacy and scalability requisites, e.g., [6], [7], [8], [9], but the proposed federated architectures usually have not taken into account specifically the time requirements to provide interactive services.

Today, however, the time requirements can no longer be neglected even when the user query involves complex procedures due to the increasing need of providing fast and accurate information not only to the users who come to the information desks but also to mobile users [10]. This is true not only for the private sector, but also for the public one where public real-time information systems are more and more perceived by the citizens as an important aspect of the quality of the services offered by the public administration.

Of course, developing a federated data layer to provide real time responses for all the possible types of queries is not realistic, but it might be feasible if one assumes that AUs and citizens carry out almost always a limited set of queries.

The work presented in the paper started having this aim in mind: it is intended to create customizable solutions to allow AUs and citizens to obtain fast responses to their typical queries. In particular, three different solutions have been tested in a real e-government environment involving different administrative units of a municipal administration.

The first solution is based on the federated data base solution offered by MySQL (ftp.nchu.edu.tw/MySQL/tech-resources/articles/mysql-federated-storage.html), the second is based on the OpenLink Virtuoso [11], whereas the third is related to a novel software architecture specifically developed to federate data resources resident on different nodes of a cloud to improve the execution time of the distributed queries. Different tests were performed by executing different query types to evaluate the most suitable application scenario for each solution.

The remainder of the paper is as follows: in sect.2 the details about the three solutions for data federation are discussed, in sect.3 the tools used to perform the tests are described, as well as the typical use cases taken into account, whereas sect.4 discusses the experimental results.

II. Federated Data Base Management systems: strengths and weaknesses

A federated database is a system in which several databases appear to function either as a single entity or cooperatively. Each database that belongs to the federated database is interconnected with the others through a network of computers. Every application that queries the data bases of the system will see the data as if they belong to a single virtual system. Therefore, a Federated Data Base Management System (FDBMS) represents an interface between the application software and the different DBMS resident on the different machines. It offers two types of applications: local and global in order to preserve the autonomy of the data sources and to allow the sharing of the information.
The use of a FDBMS in the domain of public administration is essential because each data storage of each AU contains references to other information located in the server of other AUs. This requires that each AU should retrieve information from multiple remote archives in the due time.
The main point to take into account to satisfy the time requirement in this type of scenario is related to the time spent by the application to query the virtual DB created as the union of the federated tables. In fact, the basic tables of this virtual DB are not indexed since they belong to old DBs, and consequently the execution of any complex query requested by the application running on an AU (e.g. the join of tables resident in different DB) involves the download of the entire remote tables and the execution of heavy operations on the server of the AU to extract the data.
This represents a great problem in terms of time performance especially when the remote tables contain a huge quantity of data or when the network communication is not too fast.
To evaluate the advantages of using our tool to federate data bases resident on a Cloud, in the following we recall first the two solutions used to implement a FDBMS for e-government applications, i.e., MySQL federated DB to federate SQL tables and OpenLink Virtuoso to federate triple data stores where data are coded by triples *subject-predicate-object* according to the standard RDF format [11].

A. Federated MySql DB

The Federated MySQL DB is a specific storage engine in MySQL, which was introduced with version 5.0.3 in 2005 to support the federation of the MySQL databases. This storage engine enables data resident on a local server to be accessed from a remote MySQL database without using replication or cluster technology. When using a federated table, queries on the local server are automatically executed on the remote (federated) tables. No remote data is downloaded from the remote tables to the local ones if the involved fields are indexed columns.

B. OpenLink Virtuoso

OpenLink Virtuoso was first developed as a row-wise transaction oriented to federate data coded in the standard RDF [12] format (RDBMS) using the same approach of the SQL tables federation. Virtuoso's core is a pioneering Virtual Database Engine; it was subsequently re-targeted as an RDF graph store with built-in SPARQL for distributed query [13]. The largest Virtuoso applications are in the RDF space, with terabytes of RDF triples that usually do not fit in RAM.
Let us note that Virtuoso, as well as similar RDF based approaches to data integration, represents the latest step towards the implementation of open and interoperable data systems, but for their success it is necessary to develop agreed data vocabularies and related properties, also called data ontology, such as the ones proposed in [14], [15], [16] to deal with the modeling of user activities, transport networks and public service. This is for further work.

C. Federating Web Services connected to a Cloud

This solution is sketched in fig.3. The user interactions with the federated database is supported by a front-end. Also in this case, if the query involves not-indexed fields, the front-end has to download the entire relevant tables from the remote server to the local machine where it will perform the requested query and return the results to the user. It is easy to understand that this mechanism can be considered a feasible solution when the data size is not too large and the network connections are fast enough. In all the other cases, this solution introduces a latency that cannot be accepted in real time applications.

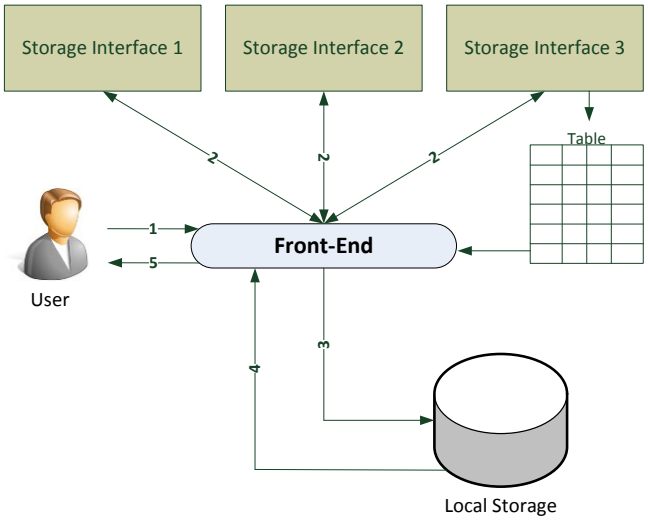

Figure 1: Example of a federated DB with not indexed fields: 1) the user performs the query, 2) the front-end interacts with the storage interface of each archive to obtain the entire tables, 3) the front end download the data, 4) the data are joined locally, and 5) the results are returned to the user.

Fig.2 shows how the mentioned mechanism has been modified to carry out a distributed query in the proposed architecture in the due time. In particular, on each data storage belonging to different Administrative Units is installed a Web Service (WS) that can be considered as a driver able to execute a single predefined query (from Q_1 to Q_n in fig.2).
The Front-End contains a set of Procedures (from P_1 to P_n) that should be configured by an expert designer. Each procedure P_i represents a well-known query Q_i that can be invoked by either a citizen or an employer. Each procedure P_i contains the instructions to execute the specific queries Q_i on the distant WSs to download from such WSs only the rows useful for the execution of the overall query.
This solution is flexible enough because it can be customized to the needs of every administrative area by only configuring the set of queries and related procedures.

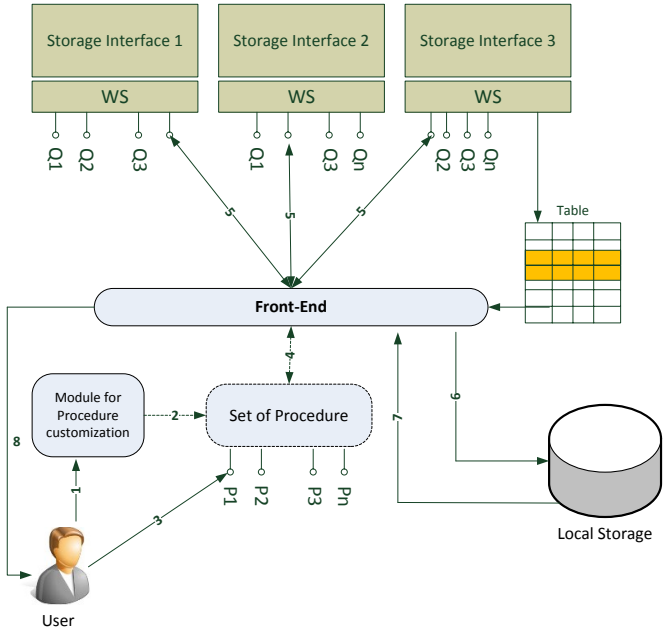

Figure 2: an example of how our solution may work in practice: 1) the user creates a set of specific procedure for its domain through a software module, 2) the module returns a set of procedures accessible through a set of interfaces, 3) the user performs a query by invoking a one of the available procedures, 4) the front-end receives the user request, 5) the front-end invoking a set of queries exposed by different WS that are mounted on each storage interfaces, 6) only the set of rows useful for the query issued by the user are sent from the front-end to server to which the user is connected and stored locally, 7) the data coming from different remote tables are joined on the local server, and 8) the results are returned to the user.

III. MATERIAL AND METHODS FOR PERFORMANCE EVALUATION OF RELEVANT USE CASES

The performance evaluation has been carried out using suitable monitoring tools able to analyze the data flow and response time for each query submitted to the system. For this reason, we have used:

- An ad-hoc software tool developed in C# for the interaction with the federated databases. This software tool allows us to carry out the federated queries issued by the users and to show the content of each table of the federated database. In particular the tool has been conceived to carry out queries that need to join the data of the tables stored in the databases of three administrative units of the Public Administration of the city of Catania: the registry office, the cadastral office and the garbage fee office.
- The WireShark (Network Protocol Analyzer) tool, that is a free and open-source packet analyzer for network troubleshooting and performance analysis.

The use cases taken into account deal with the main queries that need to extract data resident on the archives located in different AUs of the Public Administration to manage the payment of a local tax, i.e., the garbage collection tax. Fig.3 shows the two separate archives, i.e., the cadaster and garbage fee (TARSU), that are needed to compute the tax amount for each real estate owner. Let us note that these two storages are located in two different administration units and that the federated solution is certainly suitable to join data related to the two different archives.

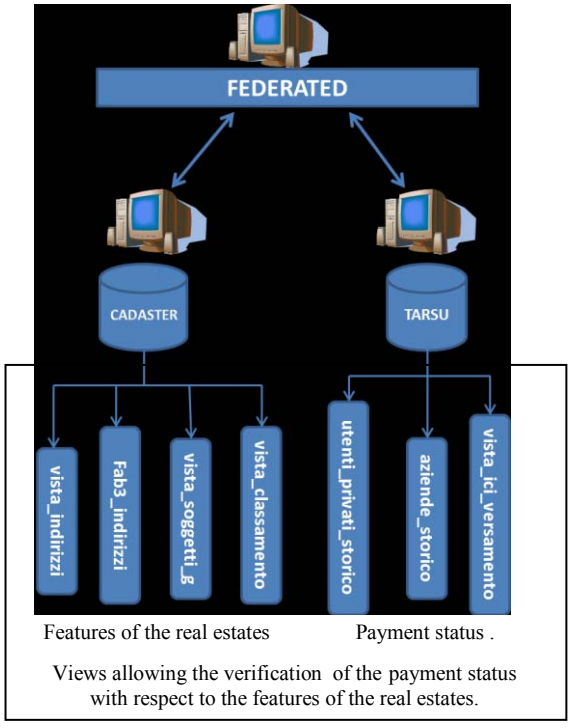

Figure 3: Archives involved in the garbage collection tax computation and typical views that allow proprietary owners and employers to verify the payment status.

IV. PERFORMANCE EVALUATION

To evaluate the performance of the previously outlined solutions, the system has been subjected to a set of queries, and statistical information has been collected during the queries execution phase .using the software Wireshark. Three filters has been applied to monitor the network traffic: a filter on the MySQL databases to isolate all the relevant MySQL traffic, a filter on the IP destination address to monitor the information sent to the remote servers, and a filter on the IP source address to monitor the information received by the local server from the remote database.By the use of Wireshark it was possible to visualize a graphical representation of the network traffic. The performance evaluation has been carried out considering some statistical information, in details: number of transmitted or received packets, time between the send and receive of the first and the last packet, and number of transmitted or received bytes.

The executed queries are of increasing difficulty: 1) Query on a single table, 2) Query on a single table with "WHERE" clause, 3) Queries on two tables, 4) Query on two tables of the same database with the clause "WHERE", and 5) Query on two tables of different databases with "WHERE" clause.

The tests performed to measure the performance of the Federated MySQL DB and OpenLink Virtuoso are summarized in different graphs. Each graph shows in the x-axis the time in seconds and in the y-axis the number of transmitted packets. The execution time is given by the time interval in which the packet traffic differs from zero.

The red graph represents the volume of packets sent from the local server to the WSs located on the remote servers. The blue graph deals with the packets received from the remote servers. The black graph, drawn for the first, second and fifth case, represents the MySQL traffic.

The red and the blue graphs are more or less the same for both the Federated MySQL DB and OpenLink Virtuoso solutions. This is quite understandable since the data of both such solutions are stored on SQL tables and the extra time needed by the OpenLink Virtuoso with respect to the Federated MySQL DB to pass from the query expressed in the standard RDF vocabulary to the one expressed using the columns of the inner SQL database is quite negligible.

The black graphs in the first two experiments (see figg.5, 6) are concentrated in the initial phase of the query execution; this demonstrates that most of the execution time deals with the transmission of the records fields of the relevant tables from the remote servers to the local one. This is true for all the analyzed queries (see figg.5, 6, 7, 8) except for the last one (fig.9), where the query was based on the indexed fields of the remote tables. In this case, the remote server extracts from the tables only the relevant records, thus sending to the local server only the relevant data and not the entire tables. The little black graph shown in fig.9 confirms the low quantity of data extracted from the remote SQL tables to respond to the user query based on indexed fields. This determines a low query execution time that is mainly due to the transmission of such few data from the remote servers to the local one.

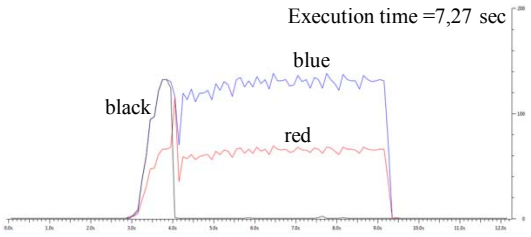

Figure 5: Simple query on single table.

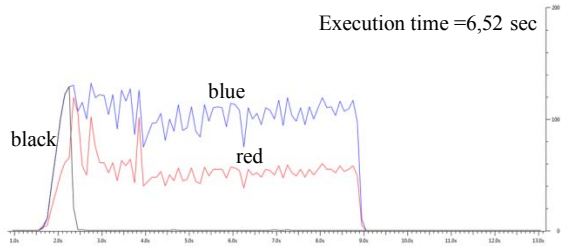

Figure 6: Query on single table with WHERE clause

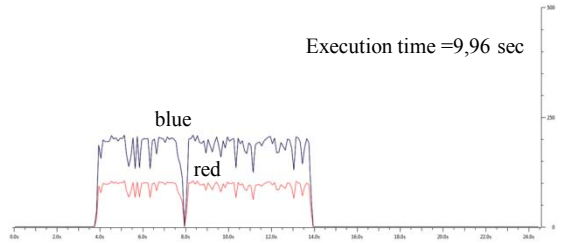

Fig. 7: Query on two tables.

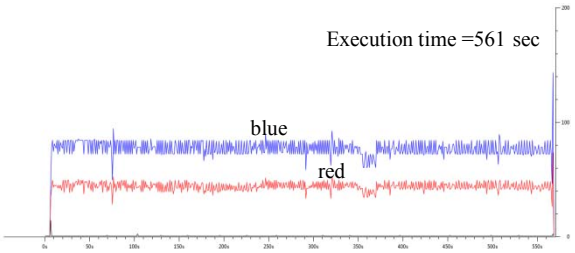

Figure 8: Query on two table with WHERE clause without indexed fields.

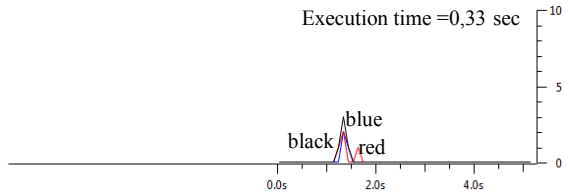

Figure 9: Query on two table of two different databases with WHERE clause and indexed fields.

The obtained results are not surprising since it is known that the use of not indexed fields determines high execution time. However, they are relevant since they show that in real applications such delay is not compatible with the time performance needed to support interactive services that are featured by response times of a couple of seconds. On the other hands, it is not possible to redesign the databases of the

public administration, and consequently the indexing scheme cannot be modified to support real time distributed query. For this reason, we develop the alternative way to federate databases outlined in the previous section, i.e., the one of implementing on each remote server a software package able to extract for each query only the relevant data for responding to the query even if such relevant data don't correspond to indexed columns of the tables. This reduces the number of type of queries that can be issued but it is suitable to respond in real time to the queries of a wide set of use cases.
In table1 we report the performance of our solution to carry out the five types of queries analyzed from fig4 to fig.9. Such performance compared to the performance of the conventional federated solutions demonstrates that our solution outperforms the conventional approaches especially at the increase of the amount of data stored in the relevant data tables. Let us note that in the fifth case our solution has the same performance of the conventional approaches since in this case it is possible to execute the query issued by the user using indexed fields.

TABLE I - Comparison of the query execution time. Federated MySQL DB - OpenLink Virtuoso versus our solution.

Query	Federated solution		Our Solution	
	Exec. Time (sec)	Total transmitted byte	Exec. Time (sec)	Total transmitted byte
1	7,27 s	11MB	1,0	100KB
2	6,52 s	11 MB	1.2	115KB
3	9,96 s	27,6 MB	1.64	200KB
4	561s	60 MB	2.68	500KB
5	0,33	4 KB	0,33	4KB

6 CONCLUSIONS

The main problem of the federated data base approach is that the queries are carried out with respect to the virtual DB created as the union of the federated tables. The execution of complex query that contains not indexed fields involves the download of the entire relevant tables (see row 4 of table 1). This reduces greatly the time performance especially when the tables consist of a huge amount of data or when the network communication is not fast enough.
The solution based on our framework show how it's possible to improve the time performance using a customized solution. In this case only a set of data (rows) really useful to produce the query results, are downloaded from the remote servers. The customizable solution needs an appropriate knowledge of the database schema for each AU to obtain the specific set of procedures for querying the distributed databases.
In the domain of public administration where the needs of real-time execution is more and more increasing, it is recommended to use our solution instead of the federated approaches since it is not possible to redesign the indexing scheme of the data bases.
The tests were performed connecting the servers hosting the databases in a local cloud. In the near future we plan to measure the proposed solution over the same metropolitan cloud interconnecting the administrative units involved in the selected application scenario. Also, the main server to manage the queries coming from the citizens PCs or mobiles will be installed over a cloud machine to provide the citizens with the needed interactive information services even in case such services involve complex queries of distributed databases.

REFERENCES

1. Zakareya Ebrahim, Zahir Irani, E-government adoption: architecture and barriers. Business Process Mng. Journal, 2005.
2. Sheth, A., Larson J.A., Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases, ACM Computing Surveys, vol 22(9), 1990
3. Stone P.D., et al., Query Execution and Maintenance Costs in a Dynamic Distributed Federated Database, www.usukitacs.com/sites/default/files/P5_pstone_query_execution_maintenance_costs_federated_db.pdf, 2012
4. Costanzo A., Faro A., Giordano D., Venticinque M., Wi-City: A federated architecture of metropolitan databases to support mobile users in real time, International Conference on Computer and Information Science, ICCIS 2012, A Conference of World Engineering, Science and Technology Congress, ESTCON 2012, Kuala Lumpur, 2012
5. Bent G., et al., A Dynamic Distributed Federated Database, Proc. 2nd Conf. Int. Technology Alliance 2008.
6. Smitha, K., Thomas T., Chitharanjan K., Cloud Based E-Governance System: A Survey, Procedia Eng., Vol.38, 2012
7. Ramgovind, S. Eloff M., Smith, E., The management of security in Cloud computing, Information Security for South Africa (ISSA), vol., no., pp.1-7, 2-4 Aug. 2010
8. Mukherjee, K.; Sahoo, G.; , A novel methodology for security and privacy of cloud computing and its use in e-Governance, World Congress on Information and Communication Technologies (WICT), 2012
9. Tripathi, A., Parihar, B., E-Governance challenges and cloud benefits," IEEE Int. Conf. on Computer Science and Automation Engineering (CSAE), 2011
10. Costanzo A., Faro A., Giordano D., Wi-City: Living, deciding and planning using mobiles in intelligent cities, 3rd Int. Conf. on Pervasive Embedded Computing and Communication Systems, PECCS, Barcelona; Spain, 2013
11. Openlink Virtuoso: Universal server platform for the real-time enterprise, http://virtuoso.openlinksw.com
12. Powers,S. Practical RDF, O'Reilly Media, 2003
13. Quilits B., Leser U., Querying distributed RDF data sources with SPARQL, Proceedings of the 5th European semantic web conference on The semantic web, ESWC'08, 2008
14. Zhai, J., Jiang, J., Yu, Y. and Li, J.: Ontology-based Integrated Information Platform for Digital City, Proc. Wireless Comm., Networking and Mobile Comp., WiCOM '08, 2008
15. Faro, A., Giordano D., Musarra, A. 2003: Ontology Based Mobility Information System. Proc. of IEEE Systems, Men and Cybernetics , vol.5, 4288-4293, Washington, IEEE, 2003
16. Costanzo A., Faro A., Giordano D., Spampinato C.,, Implementing Ubiquitous Information services with ontologies: Methodology and case study, IEEE Proc. Federated Conference on Computer Science and Information Systems, FedCSIS Wroclaw, Poland, 2012