Distr.
GENERAL

WP.19
6 May 2011

ENGLISH ONLY

UNITED NATIONS ECONOMIC COMMISSION FOR EUROPE (UNECE)
CONFERENCE OF EUROPEAN STATISTICIANS

EUROPEAN COMMISSION
STATISTICAL OFFICE OF THE EUROPEAN UNION (EUROSTAT)

ORGANISATION FOR ECONOMIC COOPERATION AND DEVELOPMENT (OECD)
STATISTICS DIRECTORATE

Meeting on the Management of Statistical Information Systems (MSIS 2011)
(Luxembourg, 23-25 May 2011)

Topic (iii): Innovation and related issues

Monitoring the acquisition process by web widgets

Invited Paper

Prepared by Leonardo Tininini and Antonino Virgillito,
Italian Institute of Statistics (ISTAT), Italy

I. Introduction

1. The web is increasingly becoming a fundamental means for supporting several phases of the statistical data life cycle, particularly data acquisition and dissemination (Sindoni & Tininini, 2008), and several products are currently available, even as open source software (Barcaroli *et al.*, 2008), to support statistical offices in many common activities. However, little interest has been devoted so far to use the web for publicly monitoring the acquisition process itself, thus providing an effective feedback to respondents, and more generally to users.

2. In this paper we illustrate the main features of a collection of so-called *web widgets*, that were developed for monitoring the 2010 Italian Agriculture Census acquisition process. A widget is a small web component that can be freely embedded into any web page (e.g. a blog), without requiring specific technical knowledge (Patruno, 2010). Widgets can be composed and organized in personalized dashboards according to the user's specific needs and preferences. The widgets described in this paper display some key data about the Census acquisition process, e.g. the number of Census web questionnaires compiled by enterprises, as well as the hourly distribution of web questionnaire compilations in the last 24 hours.

3. Data are refreshed at specified time intervals, by providing a nearly-real-time overview of the main indicators summarizing the acquisition process. In this way it is the acquisition process itself to become the focus of the statistical analysis and the main data can be freely and constantly accessed by both Census operators and conventional web users, thus obtaining a high degree of timeliness and transparency on the overall process.

4. In order to avoid the acquisition system's overloading, a sophisticated mechanism was developed, based on decoupling and periodically synchronizing the data accessed by the widget requests from the source data, continuously updated by the acquisition system. Such mechanism is closely related to the techniques of

materialized views and view-based query rewriting, commonly used in the context of data warehousing to improve the performances of OLAP systems. In this paper we describe how such techniques were adapted to our context of interest and introduced into both the application and data layers, resulting in a good trade-off between scalability and freshness of the provided information.

5. The paper is organized as follows. Section II shows the main functional characteristics of the web widgets developed for the 2010 Italian Agriculture Census, i.e. the data they provide as well as their various visualizations. In Section III the techniques of materialized views and view-based query rewriting are briefly illustrated to explain how a good trade-off between performances and freshness of the provided data can be achieved. The system's architecture is analyzed in Section IV, highlighting the design choices made in order to achieve high scalability. Section V concludes the paper.

II. The web widgets

6. The main objective underlying the development of the web components described in this paper was to provide a compact, effective and timely overview of the 2010 Italian Agriculture Census acquisition process. Obviously, this does not necessarily require the introduction of widgets and could have been achieved by developing a collection of conventional, dynamically generated web pages. Such dynamic pages could have been linked from the main "static" Census website and indeed, this is also one of the possible ways to access the web widgets that will be described in the following.

7. However, what makes widgets particularly appealing (but also very challenging for several aspects) is that they can be freely embedded into *any* web page (e.g. become part of a blogger's page) without requiring specific technical knowledge. In other words, they can become one of the components of any web page, in particular be freely composed and organized in personalized *dashboards*, according to the user's specific needs and preferences. As we discuss in Section IV, this feature poses a serious scalability challenge, because the widget designer does not know in advance the number and kind of pages that will embed her widget, as well as the number of times such pages (and hence, indirectly, the widget(s) embedded therein) will be accessed by Internet users. In the same section we illustrate how the internal design of widget was devised to specifically account for this issue.

8. The widgets described in this paper display some key data about the Census acquisition process, e.g. the number of Census web questionnaires compiled by enterprises, as well as the hourly distribution of web questionnaire compilations in the last 24 hours. Data are displayed in various visualization formats, each providing a different perspective of the aggregates summarizing the aggregation process, particularly according to the territorial and temporal dimension. More precisely, the different widgets display data: (a) in the conventional tabular format, to provide the key numerical data describing the process in a compact form; (b) by means of a thematic map, to stress how the analyzed phenomena are geographically distributed; (c) on a bar-chart, to better emphasize the hourly distribution of questionnaire acquisition.

A. The "General picture" widget

9. The "General picture" widget (see Figure 1) shows the numeric values of 5 main aggregates, namely: (i) the number of operators; (ii) the number of interviews completed by operators and recorded in the Census Management System; (iii) the number of questionnaires (fully) compiled by enterprises by using the web; (iv) the number of questionnaires returned (compiled) by operators by the web; (v) the number of questionnaires under compilation (i.e. only partially compiled) on the web by enterprises.

10. As for all widgets the refresh rate can be tuned according to the time required to perform the update of all data and to the related additional workload this update requires to the database (which is obviously the same DB, continuously accessed by the web questionnaire application, as well as by the Census Management System). We discuss the technical issues involved in the tuning of the update frequency in Sections III and IV.

Figure 1: the "General picture" widget

11. This widget accepts an additional parameter to select the territory of interest, namely the whole Italian territory (this is the default setting) or, alternatively, a specific region by specifying its Istat territorial ID in the embedding HTML code. In the current version there is no explicit navigational feature (e.g. specific links in the widget to switch from one region to a different one), but the support to simple territorial "roll-up" and "drill-down" functionalities are planned for future releases.

B. The "Regional thematic map" widget

12. The data in the "Regional thematic map" widget is organized in 5 tabs, each corresponding to a different Italian map, where each region's colour is faded in proportion to the value of the specific aggregate for that region. The aggregates are the same as those in the "General picture" widget and the user can switch from a certain aggregate to a different one by selecting the corresponding tab. The maps are built using the freely-available Google Interactive Chart Tools API (specifically, the Geo Map component).

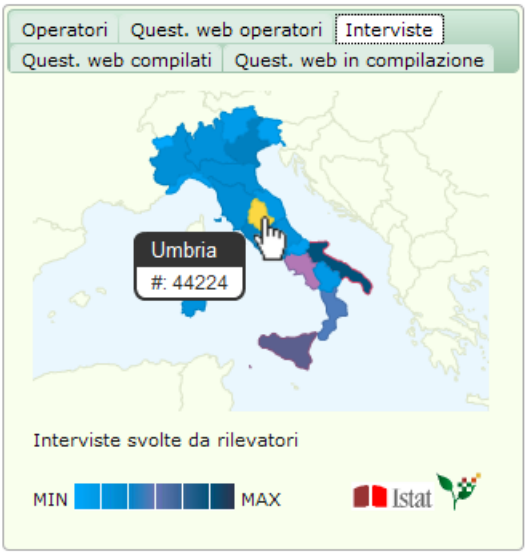

Figure 2: the "Regional thematic map" widget

13. The aggregates are represented on the map using classes of values. However, the exact aggregate values, along with the name of each region, can be obtained by rolling the mouse cursor over the region on the map. Figure 2 shows the data and name obtained by rolling the mouse cursor over the Umbria region.

C. The "Latest news histogram" widget

14. The "Latest news histogram" widget (Figure 3) shows the time distribution of the questionnaires compiled per-hour over the last 24 hours. The histogram shows the total number of questionnaires compiled on the web by both enterprises and operators. The fraction corresponding to enterprises is depicted in darker green, the one corresponding to operators in lighter green. As for the thematic map widget more details can be obtained by rolling the mouse cursor over some sections of the widget. More precisely, the roll-over on each bar displays the details of the two component values (enterprises vs. operators)

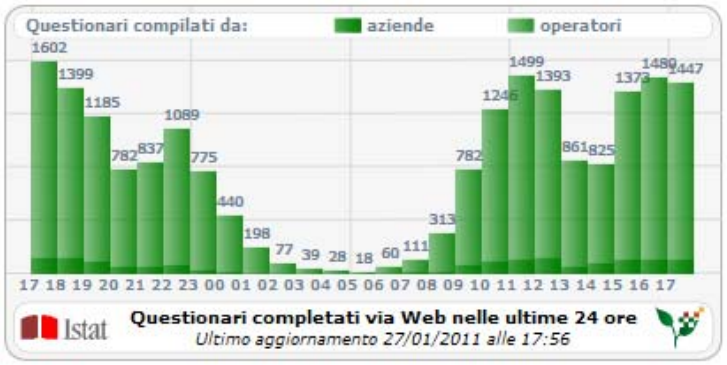

Figure 3: the "Latest news histogram" widget

15. Instead, by rolling the mouse over one of the two legend components on the upper part of the widget displays two different histograms, corresponding to questionnaires compiled only by enterprises or operators, respectively. Figure 4 shows the histogram displayed when rolling over the enterprises ("aziende") legend component (note that bars are automatically re-scaled to improve readability).

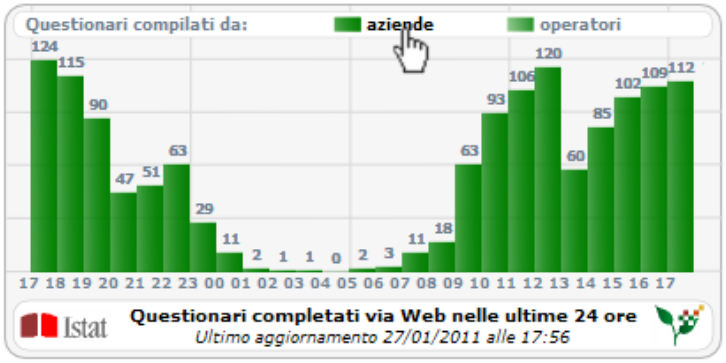

Figure 4: rolling over to show the enterprise-only histogram

III. Enhancing performances by materializing data

16. The data displayed by the widgets are typically obtained by applying simple aggregations (like counts, sums or ratios thereof) on the elementary data, which are constantly and concurrently updated by the acquisition process. For the Italian Agriculture Census this implies in general, as suggested by some values shown in Figure 1, counting several hundreds of thousands of records, while they are being concurrently updated by a few thousands of acquisition operators.

17. Aggregate queries, above all when executed on large tables while they are being updated, can significantly affect the database performances, as they necessarily require the access to large amounts of data, often combined with operations of sorting or similar, when a "grouping" is also required (e.g. SELECT-GROUP-BY queries). In other words, the queries required by the widgets could have resulted in a substantial performance degradation for both the operators and enterprises working on the Census Management and Acquisition System and the users monitoring the acquisition process through the widgets. This potential degradation has been avoided by adapting two techniques, commonly known as *view materialization* and *query rewriting*, widely used in data warehousing environments.

18. Informally speaking, a materialized view is the combination of a query definition (i.e. the SQL expression of the query) together with the result (i.e. the set of records) obtained by running that query on a given database at a given point in time. The result is typically "refreshed" (recomputed) at predefined time intervals (e.g. every week-end, every night, or more frequently, if necessary), and the refresh can be achieved either by restarting the computation from scratch (complete refresh) or by computing the new

values, combining the previous results with the changes occurred since the last refresh time (incremental refresh). Incremental refresh is only possible for particular classes of aggregations (e.g. COUNT and SUM), is generally more complex and can be convenient only if the percentage of updates is small with respect to the total number of records to be aggregated. Materialized views can be stored either on permanent storage devices (i.e. the server disks) or in a cache (i.e. in the server RAM).

19. The problem of evaluating the answer to a query by using some materialized (pre-computed) views has been extensively studied in the literature (Halevy, 2001). In the specific context of aggregate queries this is commonly known as (aggregate) query rewriting (Cohen, Nutt & Sagiv, 2006; Grumbach & Tininini, 2003). The term stems from the fact that the original aggregate query (referring to the elementary data) is rewritten into an equivalent one (referring to the views), much faster to be computed. The rewritten query generally combines and/or further aggregates the materialized views (e.g. by summing the regional aggregates to compute a national aggregate). Obviously the price to pay is that the data returned to users are not as "fresh" as they would be, by running queries directly on the elementary data.

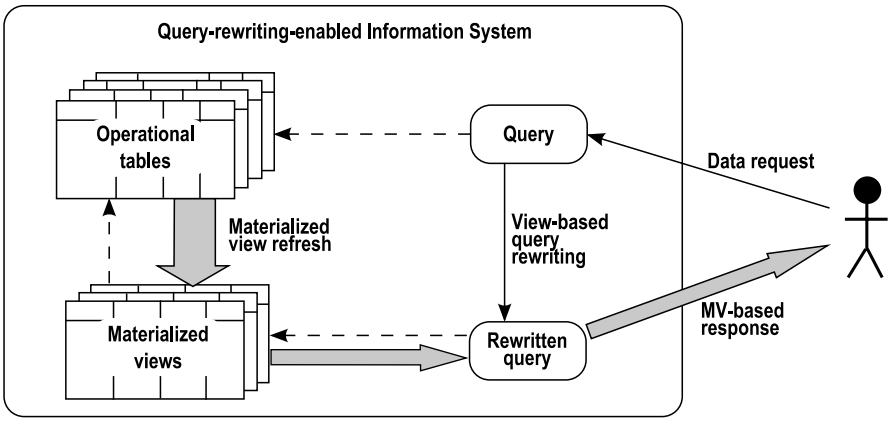

Figure 5: Materialized views and query rewriting

20. The process of materialization and query rewriting is outlined in Figure 5. The overall system is constituted by both operational tables (with elementary data) and materialized views, which are defined in terms of the operational tables and periodically refreshed. When the user expresses a data request by formulating a query (which refers to the operational tables) the rewriting engine transforms it into an equivalent rewritten query which refers to the materialized views instead of the operational tables. The rewritten query is then executed on the materialized views (generally obtaining a much faster execution plan) and the results returned to the user.

IV. The widgets' software architecture

21. The widgets are implemented within a single Java web application, directly accessing the Census database. One of the main features of the widgets is the possibility for any end-user to freely incorporate them into her web site. Potential scalability problems may arise since the number of users that will choose to incorporate the widget into their site is not under the control of the designer and consequently there is no way of estimating the additional Internet traffic due to the widgets and, more importantly, the resulting additional load on the database.

22. The risk is to introduce bursts of unpredictable query load that might interfere with the database access made by operators during their work, possibly slowing down the overall acquisition activity. Thus, the main criteria driving the design of the application was to conceive a solution that could always result in a minimum (and bounded) overhead on the database against unpredictable request traffic.

23. Figure 6 shows the schema of the general high-level design of the web application. The basic criterion introduced in order to achieve high scalability is to decouple the access to the widgets made by the

user from the access to the database made by the widgets. The design is centered on a specific component (referred in the Figure as Update Job), whose execution is scheduled to automatically run every minute. This component periodically computes the aggregates values and stores them into the application memory.

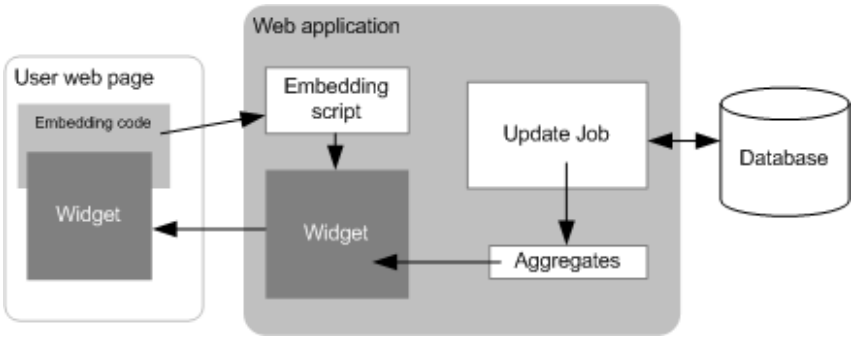

Figure 6: the overall high-level architecture of the web application

24. Each widget is implemented as a separate JSP page. The process to embed the widgets into a generic web page is structured into three phases. The embedding starts form a single row of predefined HTML code that has to be included in the page (Embedding code). An example of Embedding code is shown in Figure 7.

```
<script src="https://censimentoagricoltura.istat.it/CensWidget/caTable.js"
type="text/javascript"> </script>
```

Figure 7: an example of widget embedding code

25. It consists in a simple call to a JavaScript fragment (Embedding script) that is stored on the server and is downloaded and executed on the client browser. The execution of the Embedding script creates a frame within which the JSP page rendering the widget is actually loaded. Through this technique, the widgets can be incorporated into virtually any web site or blog regardless of the underlying technology and without any programming knowledge, at the same time maintaining maximum browser compatibility and a fixed and predictable screen space.

26. From this description we can see that at every user request a widget is rendered only using pre-computed information. In particular, the only element in the application accessing the database, at predetermined intervals, is the Update Job. Then, the additional load on the database introduced by the widgets can be precisely controlled by tuning the query interval. More importantly, the load is independent from the number of accesses to the application, so high levels of scalability can be achieved.

27. Similarly from what written above for query rewriting, the obvious drawback to this solution is that the information provided is not updated at query time but only refreshed periodically. However, we deem that the refresh interval set in our implementation (one minute) is adequate for the kind of monitoring information we provide and the benefits obtained on the scalability side largely compensate the drawbacks.

28. Furthermore, setting a refresh interval which is too short can have a negative impact on database performance, especially if the queries (as in this case) require the calculation of aggregate values on large amounts of data. In the case of the Italian Agriculture Census the additional workload is relatively low, hence the data could be refreshed every minute without significantly affecting the overall system performances. However, in the case of the Population Census, where the amount of data to be processed is expected to be up to 2 orders of magnitude larger, more conservative rates will probably be required, e.g. one refresh every 5-10 minutes or even every hour.

V. Conclusion and future works

29. In this paper we described the design of a set of web widgets for user-side monitoring of the Agriculture Census activity. The proposed design can be considered as a general pattern for building embeddable components for the monitoring of an acquisition/production process, where a high level of scalability can be reached at the only price of a delay in the information update. This delay strictly depends on the efficiency of the query execution, motivating the application of optimization techniques based on materialized views. In summary, we showed how combined application- and database-level optimizations allowed us to achieve a good trade-off between scalability and freshness of the provided information. We envision a future application of these ideas within the more severe testbed represented by the forthcoming Population Census.

VI. References

Barcaroli, G., Bergamasco, S., Jouvenal, M., Pieraccini, G. & Tininini, L. (2008): Generalised software for statistical cooperation. Contributi ISTAT.

Cohen, S., Nutt, W. & Sagiv Y. (2006): Rewriting queries with arbitrary aggregation functions using views. ACM Trans. Database Syst. 31(2): 672-715.

Grumbach, S. & Tininini, L. (2003): On the content of materialized aggregate views. J. Comput. Syst. Sci. 66(1): 133-168.

Halevy, A.Y. (2001): Answering queries using views: A survey. VLDB J. 10(4): 270-294.

Patruno, V. (2010): Data Sharing. Proceedings of the International Marketing and Output DataBase Conference – IMAODBC.

Sindoni, G. & Tininini, L. (2008): Statistical Dissemination Systems and the Web. In: Handbook of Research on Public Information Technology, Information Science Reference, 578-591.