Send Orders for Reprints to reprints@benthamscience.ae

The Open Cybernetics & Systemics Journal, 2015, *9*, 37-43 37

Open Access

De-Duplication Scheduling Strategy in Real-Time Data Warehouse

Hui Liu[1], Jie Song[2,*], JinBoWu[2], and Yu-Bin Bao[3]

[1]*School of Materials & Metallurgy, Northeastern University, Shenyang 110819, China;* [2]*Software College, Northeastern University, Shenyang 110819, China;* [3]*School of Information Science and Engineering, Northeastern University, Shenyang 110819, China*

Abstract: Data quality of the data warehouse is crucial to decision-makers. Data duplication is considered one of the critical factors that affect the data quality. Therefore, data de-duplication is an essential process for data warehousing. Particularly, for a real-time data warehouse, it is necessary to ensure not only the data quality in real-time, but also the performance of the front-end queries and analysis. The scheduling strategy of de-duplication in a real-time data warehouse should be well studied. In this paper, we firstly investigate the three kinds of data de-duplication scheduling strategies named De-duplication Prior scheduling Strategy (DPS), Real-time scheduling Strategy (RS) and ETL Prior scheduling Strategy (EPS); then propose a new Time-Triggered scheduling Strategy (TTS) which belongs to EPS; finally evaluate the performance of the proposed scheduling strategy through experiments. This work is contributed to the efficient data cleaning and application of real-time data warehouse.

Keywords: Data warehouse, ETL, real-time, de-duplication, scheduling strategy.

1. INTRODUCTION

In this information age, efficient high-quality decision-making has become a powerful tool for enterprises to enhance their competitiveness. More and more enterprises have been building their own decision support systems and data warehouses. For a data warehouse, successful decision support depends largely on the data quality. However, guarantee of the data quality is not an easy task because there are many factors affecting the data quality; one of the critical factors is data duplication, which can not only lead to data redundancy, but also seriously affect the data analysis and decision making. Therefore, data de-duplication is an essential process for data warehousing.

In recent years with the strong demand for real-time decision-making, the real-time data warehouse, which means that any changes of the data sources can be automatically and immediately reflected in the data warehouse, came into being. However, real-time data warehouse also brings new challenges for the guarantee of data quality.

For traditional data warehouse, whose data update periodically (usually a day or a week), and generally using historical data to make decisions, having a relatively low real-time demand on data and de-duplication, it is relatively simple and easy to implement de-duplication scheduling strategy, only needing to trigger the de-duplication algorithm after ETL at non-working hours (*e.g.* at night). Whereas for real-time data warehouse, whose data updates are instantly reflected in the data warehouse (*i.e.* ETL is real-time), having a high real-time demand on data and de-duplication, it is difficult to find a de-duplication scheduling strategy to guarantee the data quality real-timely without affecting the performance of the front-end queries and analysis. What is worse, existing researches on data cleaning and de-duplication scheduling strategy are mostly based on traditional non-real-time data warehouse. Therefore, the challenges brought by real-time data warehouse, which can be summarized as the following two points, need to be addressed, and new de-duplication scheduling strategies under real-time environment to guarantee the data quality much more better and in real-time need to be proposed:

- **Increasing the burden of the data warehouse.** Real-time data warehouse, just as its name implies, updates frequently, so in order to guarantee the data quality, the de-duplication algorithm needs to be invoked frequently, thus increasing the burden of the data warehouse. Therefore, without a reasonable de-duplication scheduling strategy, the data warehouse may be in a constant busy state, seriously influencing the real-time decision and the query performance.
- **Causing small data duplication rate but large de-duplication cost.** ETL of real-time data warehouse occurs once the data update, so sometimes the data duplicate rate may be very small, at which it is not worthwhile to carry out the costly data de-duplication.

In this paper, according to the scheduling time, de-duplication scheduling strategies are divided into three types: De-duplication Prior scheduling Strategy (DPS), Real-time scheduling Strategy (RS) and ETL Prior scheduling Strategy (EPS). We will first describe the design ideas of all three scheduling strategies; then propose a new Time-Triggered scheduling Strategy (TTS) which belongs to EPS; finally evaluate the proposed approach qualitatively and quantitatively through several experiments.

1874-110X/15

Rest of this paper is organized as follows. Following the introduction, related works are briefly introduced in Section 2. Then, we illustrate three de-duplication scheduling strategies in Section 3. Section 4 proposes TTS. Section 5 evaluates the proposed TTS through several experiments. Finally, we summarize our work and present our future works in Section 6.

2. RELATED WORKS

Duplicate detection is the approach that identifies multiple representations of a same real-world object. It is a crucial task in data cleaning, and it has applies in many scenarios such as data integration, customer relationship management, and personal information management [1]. The duplicate problem has been studied extensively under various names, such as merge/purge [2], record linkage [3], entity resolution [4], or reference reconciliation [5], too names but a few [6]. There are some researches on real-time de-duplication recently [7]. Proposed the CBLOCK which learns hash functions automatically from attribute domains and a labeled dataset consisting of duplicates; [8] proposed an object-based de-duplication framework and an efficient object index mechanism to speed up the searching facility to identify duplicate objects; [9] proposed a buffer cache de-duplication technique for query dispatch in the field of replicated databases which has a well-known problem that different buffer caches share some identical data; [10] propose three algorithms for the problem of string edit distance with duplication and contraction operations, which improve the time complexity of previous algorithms for this problem. To our best knowledge, few works focus on the scheduling strategy of de-duplication.

Researching on real-time data warehouse (RTDWH) is the hot topic in data warehouse area. Now days, there are many new concepts, such as Zero Latency Data Warehousing (ZLDWH) [11], Active Data Warehousing (ADWH) [12, 13], are both real-time data warehouses. In RTDWH area, real-time ETL and triggering approach is proposed and studied, but few works focus on data de-duplication scheduling. Most works which focus on the real-time data refreshment are close to our research [14-16]. Bouzeghoub *et al.* discussed in [15] an approach for modeling the data refreshment processes. They distinguish three types of data refreshments that are triggered by a timer or data conditions in the data warehouse system [16] showed a design and implementation for data refreshments from various sources as a workflow. Qu, *et al.* [17] proposed a balanced scheduling algorithm QUTS in a web database and a two-level scheduling structure. Bateni and *et al.* [18] proposed an update scheduling algorithm in RTDWH. Through adjusting the execution order of updates, it can improve data freshness of RTDWH. In out paper, the de-duplication scheduling is related to the ETL scheduling in the proposed Real-time scheduling Strategy (RS), and in ETL Prior scheduling Strategy (EPS), the ETL scheduling is also considered.

3. SCHEDULING STRATEGY

According to the invoking time of the data de-duplication algorithm, de-duplication scheduling strategies can be categorized into three kinds: De-duplication Prior scheduling Strategy (DPS), Real-time scheduling Strategy (RS) and ETL Prior scheduling Strategy (EPS). Each scheduling strategy has a different scheduling time and target data sources. This section will elaborate on the design ideas of these three scheduling strategies. To begin with, we designed a real-time data warehouse architecture. Different from the traditional data warehouse, in order to achieve the real-time capabilities, a temporary storage area for the changing data and an incremental data warehouse area are designed. In addition, the de-duplication algorithm used in this research also can be found in our previous study [19].

3.1. De-duplication Prior Scheduling Strategy

De-duplication Prior scheduling Strategy (DPS) refers to that the de-duplication algorithm is invoked before ETL. The advantage of this scheduling strategy is that data quality has been ensured before ETL, so there is no need to carry out the de-duplication again when the real-time data warehouse updates. However, this scheduling strategy has the following technical difficulties and disadvantages.

- Before ETL, data are distributed across various data sources which are distributed and heterogeneous, so the de-duplication algorithm has to be executed in various data sources, incremental data warehouse and data warehouse. This requires us to use the distributed de-duplication algorithm, such as the MapReduce programming model. However, distributed de-duplication algorithm is much more difficult to implement and has a high running cost.
- The trigger time of ETL depends on the de-duplication scheduling strategy, making it difficult to guarantee real-time capabilities. DPS causes the decision delay of real-time data warehouse.

3.2. Real-time Scheduling Strategy

Real-time scheduling Strategy (RS) refers to that the de-duplication algorithm is invoked immediately after the ETL, de-duplication algorithm has to be executed in incremental data warehouse and data warehouse. RS makes the greatest effort to ensure the real-time de-duplication so that the users can make decisions by the latest de-duplicated data (see Fig. **1**).

Fig. (1). Real-time scheduling strategy.

In Fig. (**1**), t_1, t_2, …,t_n represent the trigger time of the real-time ETL. This paper focuses on the problem of de-duplication scheduling strategy, so briefly representing the time period of real-time ETL as a time point does not affect the research results of real-time de-duplication. Corresponding to t_1, t_2, …,t_n, T_1, T_2, …, T_n represent the real-time de-duplications after ETLs. According to this scheduling strategy, de-duplication can be ensured to be farthest real-time. However, ETL of the real-time data warehouse happens frequently, so does the de-duplication. In the worst case, two adjacent de-duplications have a too small time interval, thus making the real-time data warehouse much busy with ETLs and de-duplications all the time (see Fig. **2**).

Fig. (2). Busyness of ETL and de-duplication.

The shaded area in Fig. (**2**) represents that the next ETL begins before the previous de-duplication completes, thus triggering another de-duplication. Two de-duplications run concurrently, seriously consuming the system resources and heavily influencing the system's query performance. What's more, RS can't guarantee the integrity of the data to be processed by the de-duplication. ETL of the real-time data warehouse occurs in real-time, so any small changes will be instantly reflected in the data warehouse. As a consequence, the data in sub-table may not be imported to the data warehouse yet, so the data is still incomplete as far as multi-level de-duplication is concerned. Here, multi-level de-duplication refers to that whose duplication detection considers the duplicates of both the records in current table and all referred records in the sub-tables [19]. For example, two records in *A* (see Fig. **3**) are duplicated only in condition of their similarity is greater than a certain threshold, and their referred records set in table *B* are duplicated too; similarly, whether two records in B are duplicated is also determined by their similarity and their corresponding referred records set in *C* and *D*. The detection is performed level by the level till all the tables in the hierarchy have been detected. Contrary to multi-level de-duplication, single-level de-duplication only detects duplicates in a single table (*A* in Fig. **3**) without other sub-tables (*B*, *C*, *D* in Fig. **3**), which is not enough. For example, in a medical database, where *patients* table is referred by *blood-sampling* table, it is hard to tell whether two patients are duplicated without detecting their *blood-samplings*.

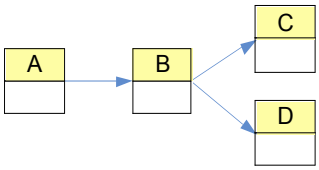

Fig. (3). Example of hierarchical tables.

3.3. ETL Prior Scheduling Strategy

ETL Prior scheduling Strategy (EPS) refers to that the de-duplication algorithm will not be invoked until the data needing to be de-duplicated accumulate to a certain amount, de-duplication algorithm has to be executed in incremental data warehouse and data warehouse. The basic idea of EPS is to find an appropriate time point on which the de-duplication algorithm is executed (see Fig. **4**).

Fig. (4). ETL prior scheduling strategy.

In Fig. (**4**), the de-duplication algorithm is not executed after the ETL at time t_1, t_2, …,t_n. At time t, when the data accumulate to a certain amount, the de-duplication algorithm is executed on all the updated data by ETL before time t with a time cost of $T_{1\sim n}$. For EPS, the appropriate time point t is the critical; the integrity of the data, the usage of the resources, the ratio of de-duplication time and idle time, and the amount of the accumulated duplication need to be considered. Section IVwill focus on EPS. If the ETL trigger time is regularly related with the increasing of the data amount in the data warehouse, Time-Triggered scheduling Strategy (TTS), which is discussed in section IV, can be adopted. Otherwise, Event-Triggered scheduling Strategy (ETS) can be adopted, the critical events are defined and the de-duplication algorithm is triggered when these events happen. The occurrence frequency of the events may be regular, such as "every time on the hour", or completely random, or presents complex regularities, or depends on the specific business logic.

4. TIME-TRIGGERED SCHEDULING STRATEGY

Time-Triggered scheduling Strategy (TTS) belongs to the ETL Prior scheduling Strategy (EPS), the basic idea of TTS is to figure out an appropriate time interval T through analyzing the various factors of the real-time data warehouse, and then execute the de-duplication algorithm in every time interval T. TTS is simple and effective approach, whose key issue is to determine the time interval T and adjust it dynamically.

Generally, the execution time of the de-duplication algorithm [19] has an approximately linear relationship with both the data amount and the duplicate rate. Let the amount of the data to be de-duplicated be m, the duplicate rate be r and the execution time be$t$, then t can be expressed as a function of m and r: $t = \omega(m, r)$.

$$\left.\begin{aligned}\frac{\partial\omega(m,r)}{\partial m} &= X_1 r + Y_1 \\ \frac{\partial\omega(m,r)}{\partial r} &= X_2 m + Y_2\end{aligned}\right\} \Rightarrow \omega(m,r) = Am + Bmr + Cr + D \quad (1)$$

$$\therefore t = \omega(m,r) = Am + Bmr + Cr + D$$

In formula (1), A, B, C, $D$are positive coefficients related with specific de-duplication requirements. The verification of this formula can be found in the experiments in section V.

Assumption: For the de-duplication on the data updated by ETL during a certain time, the more times the de-duplication algorithm is executed, the more time it will cost.

Deduction: During time period T, let the amount of the data updated by ETL be m_T, data duplicate rate be r_T. Assume that the de-duplication algorithm is executed n times, the amount of the data processed each time are m_1, m_2, …, m_n, and the corresponding duplicate rate are r_1, r_2, …, r_n, then:

$$\sum_{i=1}^{n} t_i = \sum_{i=1}^{n} \omega(m_i, r_i) = \sum_{i=1}^{n} (Am_i + Bm_i r_i + Cr_i + D)$$

$$= A\sum_{i=1}^{n} m_i + B\sum_{i=1}^{n} m_i r_i + C\sum_{i=1}^{n} r_i + nD$$

$$t_{1\sim n} = \omega(m_T, r_T) = Am_T + Bm_T r_T + Cr_T + D$$

$$m_T = \sum_{i=1}^{n} m_i \qquad r_T = (\sum_{i=1}^{n} r_i m_i) / \sum_{i=1}^{n} m_i$$

For the parameters and variables in the above formula are greater than or equal to zero, it can be simply deduced that-

when $n = 1$, $t_{1\sim n}=\sum_{i=1}^{n} t_i$, and when $n> 1$, $t_{1\sim n}<\sum_{i=1}^{n} t_i$. Now, the above assumption has been proved right. The de-duplication time during time period T is defined as $\sum_{i=1}^{n} t_i$, and idle time as $T'=T-\sum_{i=1}^{n} t_i$. Thus, obviously, when$n = 1$, *i.e.* the de-duplication is executed only once, the time cost is the shortest and on this occasion, $t = Am_T+Bm_Tr_T + Cr_T+ D$. However, the less the times the de-duplication algorithm is executed, the worse the real-time performance of the data warehouse is. Therefore, to make a better tradeoff between the de-duplication time and the real-time performance of the data warehouse, it's necessary to quantify their evaluation indicators.

Definition 1: De-duplication Cycle: In the de-duplication scheduling strategy for the real-time data warehouse, define de-duplication cycle as the time interval between the start times of two adjacent de-duplications.

As shown in Fig. (**5**), T_1 and T_2 are two adjacent de-duplications, whose start times are t_1, t_2, and end times are t_1', t_2'. Then, the de-duplication cycle $T = t_2 - t_1$.

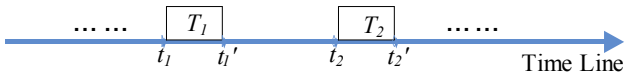

Fig. (5). De-duplication cycle.

Definition 2: De-duplication Busy Degree: During a de-duplication cycle T, define de-duplication busy degree as the ratio between the de-duplication time (T_i) and the total time (T).

Definition 3: Cumulative Duplication: Define cumulative duplication as the duplicate data accumulated during a de-duplication cycle T.

Assume that the data loading speed and the data duplicate rate are functions of time, denoted as $v(t)$ and $r(t)$, then the de-duplication busy degree $f(T)$ can be calculated as follows:

$$f(T)=\frac{A\int_0^T v(t)dt +B\int_0^T v(t)r(t)dt + C\int_0^T r(t)dt + D}{T} \quad (2)$$

De-duplication busy degree represents the time ratio of de-duplication time during a de-duplication cycle T; the larger the $f(T)$ is, the more time the system spends on de-duplication, *i.e.* the more resources the system consumes. Therefore, the *de-duplication busy degree* should be decreased as much as possible.

During a de-duplication cycle T, *cumulative duplicationg*(T) can be calculated as follows:

$$g(T)=\int_0^T v(t)r(t)dt \quad (3)$$

The longer the updated data remains duplicated in the data warehouse, the larger the *cumulative duplication* is; in other words, *cumulative duplication* reflects the real-time performance of the de-duplication algorithm. Therefore, to ensure the real-time performance of the de-duplication algorithm, the *cumulative duplication* should be decreased as much as possible. As mentioned previously, the key issue of the time-based scheduling strategy is to determine the de-duplication cycle T, which has influences on both the *de-duplication busy degree* and the *cumulative duplication*. The best case is to find a T satisfying that both $f(T)$ and $g(T)$ achieve the minimum values. If the mathematical expressions of $v(t)$ and $r(t)$ have been determined, the optimal de-duplication cycle T can be deduced by mathematical methods.

When $v(t)=0$, $r(t)=0$. Under this situation, whatever the value of T is, $g(T)$ is always equal to zero (the minimum value); when T approaches to infinity, $f(T)$ approaches to zero (because $f(T) = T_i /T$), which is to say, under the EPS, de-duplication will not happen if ETL doesn't happen.

For a real-time data warehouse, assume that ETL occurs approximately continuously and in real-time; the speed of ETL is related with data sources, data warehouse and the hardware systems, so it can be approximately assumed to be a constant value v. In addition, there are many reasons causing data duplication, and most of them are objective, so $r(t)$ is supposed to be a random function. However, during an appropriate long period of time, the data duplication rate can be substituted by a constant value R, where $R=\frac{1}{T}\int_0^T r(t)dt$, and R is independent with v. Therefore, formula (2) and (3) can be further deduced as follows:

$$\begin{aligned} f(T) &= Av + BvR + \frac{CR + D}{T} = \alpha T^{-1} + \beta \\ g(T) &= vRT = \gamma T \end{aligned} \quad (4)$$

In formula (4), $\alpha= CR+ D$, $\beta = Av+ BvR$, $\gamma=vR$. What can be seen from formula (4) is that $f(T)$ and $g(T)$ cannot achieve the minimum values at the same time, because they are separately monotone decreasing function and monotone increasing function of T on the domain of $T>0$. To address this problem, this paper determines a weight value ω, and takes $h(T)=f(T)+\omega g(T)$ as the final evaluation indicator of the TTS. When $h(T)$ achieves the minimum value, the de-duplication scheduling strategywith a de-duplication cycle T is considered the optimal:

$$\begin{aligned} h(T) &= \alpha T^{-1} + \beta + \omega\gamma T \\ h(T)' &= -\alpha T^{-2} + \omega\gamma = 0 \\ \Rightarrow T &= \sqrt{\frac{\alpha}{\omega\gamma}} = \sqrt{\frac{CR + D}{\omega vR}} \end{aligned} \quad (5)$$

It can be proved that when$T=\sqrt{\frac{CR + D}{\omega vR}}$, $h(T)$ achieves the minimum value. The above deduction can be applied in most cases. Take the loading speed as an example, for a regular data loading, we can find the function expression of $v(t)$ at a specific time period (such as morning, afternoon or the evening), and then determine the optimal de-duplication cycle for each time period; for other irregular data loading, we can also reduce the loading speed to several approximate

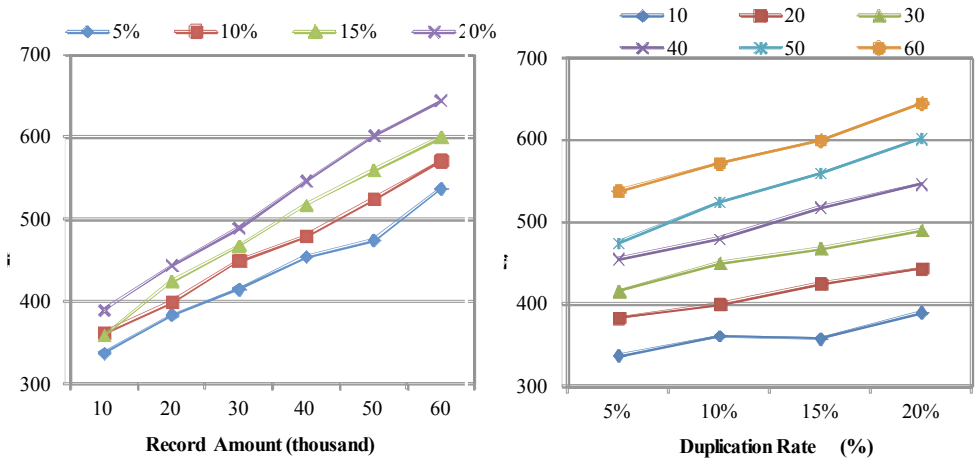

Fig. (6). De-duplication execution time under different data amounts and duplication rates.

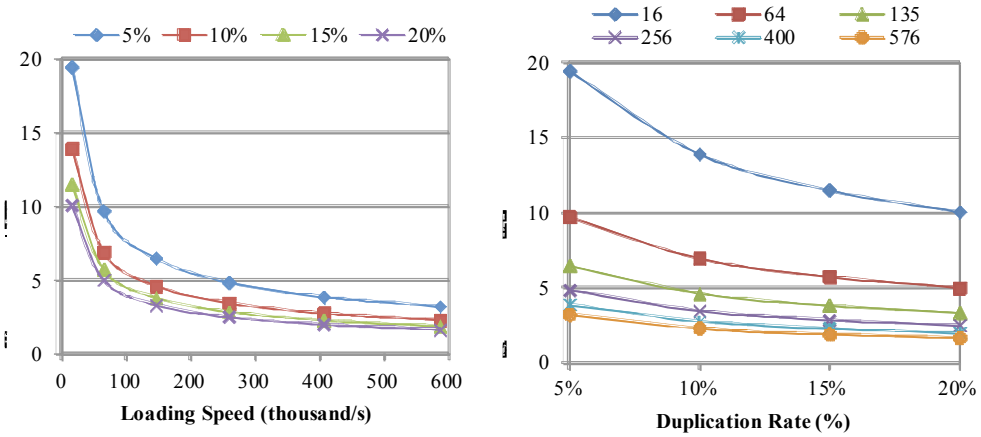

Fig. (7). The optimal de-duplication cycle under different loading speeds and duplication rates.

constant values, and then determine the optimal de-duplication cycle for each corresponding time period and adjust the de-duplication cycle dynamically according to the current loading speed.

5. EVALUATION

In the experiments, we use the 2.8GHz Pentium 4 Server, Windows Server 2003 platform and Oralce10g database. In order to control the data duplication rate, the experiments use the simulated data generated by programs. For example, to obtain a dataset with a duplication rate of 10%, we will first generate 95% non-duplicated records, and then randomly select 5% from the generated data to insert into the dataset.

Fig. (**6**) shows the performance curves of the de-duplication algorithm under different data amounts and duplication rates (r = 5 %, 10%, 15%, 20 %). It can be seen from Fig. (**6**) that de-duplication execution time has an approximately linear relationship with both the data amount and the duplication rate. Now, formula 1 has been proved reasonable, and the parameters in the formula can also be calculated as: $A = 3$, $B = 12$, $C = 148$, $D = 300$.

Then, we calculated the optimal de-duplication cycle according to formula 5 under the conditions of $R = r$= 5%, 10%, 15%, 20%, v= 16, 64, 135, 256, 400, 576 (thousand/s), and ω=1 (see Fig. **7**).

Fig. (**7**) shows the optimal de-duplication cycle curves under different loading speeds and duplication rates. It can be seen from Fig. (**7**) that optimal de-duplication cycle decreases with the increasing of loading speed and duplication rate. And the influence on the optimal de-duplication cycle is obvious only when the loading speed increases from 16 to 256.

Finally, we summarized the three scheduling strategies as follows. DPS is difficult to implement. RScan ensure a high real-time performance and is simple and easy to implement, but for the real-time data warehouse system, this strategy can easily keep the system busy with de-duplication most of the time, thus heavily affecting the front-end query and analysis performance, and the original data in the data warehouse are involved in many times of de-duplication, resulting in a longer total de-duplication time. EPS is relatively complex, it can solve the problems the other two scheduling strategies cannot, and because the de-duplication doesn't occur until the updated data accumulate to a certain amount, the original data in the data warehouse are involved in less times of de-duplication, shortening the total de-duplication time. EPS can be further divided into two categories: Time-Triggered scheduling Strategy (TTS) and Event-Triggered scheduling Strategy (ETS). The former is mainly adopted in the real-time data warehouse whose data update rate is relatively smooth and easy to predict; this approach is relatively easy to implement with low resources consumption, because the

Table 1. Comparison results among strategies.

	Real-time Scheduling Strategy	ETL Prior Scheduling Strategy	
		TTS	ETS
Real-time	Best-effort guarantee of the real-time de-duplication	De-duplication lags behind ETL	Same as the left
Totalde-duplication Time	Original data in the data warehouse involved in many times of de-duplication; longer total de-duplication time	Original data in the data warehouse involved in a few times of de-duplication, shorter total de-duplication time	Same as the left
Scheduling system cost	Less	Less	More
Algorithm complexity	Simple	Difficult	Simple
Implementation complexity	Simple	Simple	Difficult
Is Periodic de-duplication	No	Yes	No
Trigger approach	Triggered on the coming of the data	Triggered until a certain time	Triggered under a certain condition
Application fields	Real-time system of Sparse de-duplication	Real-time system of both sparse and dense de-duplication with a smooth and predictable data update	Real-time system of both sparse and dense de-duplication with a irregular and unpredictable data update

computation only includes the calculation of the de-duplication cycle in advance. The latter is mainly adopted in the real-time data warehouse whose data update rate is irregular and unable to predict; this approach consumes much system resources, increasing the system's overheads to a certain degree, because this approach has to calculate the evaluation indicator in real-time. Detailed comparison between these strategies is listed in Table **1**.

CONCLUSION

This paper first pointed out the difficulties and challenges of the data quality assurance in real-time data warehouse; then proposed and elaborated three de-duplication scheduling strategies: De-duplication Prior scheduling Strategy (DPS), Real-time scheduling Strategy (RS) and ETL Prior scheduling Strategy (EPS); finally presented a Time-Triggered scheduling Strategy (TTS). Experiments have shown that the TTS proposed can be well used in real-time data warehouse: it can reduce the data de-duplication time while ensuring the real-time performance. All of these works contribute to the efficient cleaning of the data in the data warehouse. Future works include duplicate detection on key-value data storage, on scientific data stored in XML or other formats, or in a distributed environment.

CONFLICT OF INTEREST

We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

ACKNOWLEDGEMENTS

This research was funded by a grant (No. 61173028) from the National Natural Science Foundation of China, and a grant (No. N130417001) from the Fundamental Research Funds for the Central Universities of China, and a grant (No.201403314) from the Science Foundation of Liaoning Province.

REFERENCES

[1] H. Galhardas, D. Florescu, D. Shasha, E. Simon, and C. Saita, "Declarative Data Cleaning: Language, Model, and Algorithms", In: *Proc. of International Conf. on Very Large Databases*, pp. 371-380, 2001.

[2] M. Hernandez, and S. Stolfo, "The Merge/Purge Problem for Large Databases", In: *Proc. of the ACM SIGMOD*, pp. 127-138, May 1995.

[3] I. P. Felligi, and A. B. Sunter, "A theory for record linkage", *Journal of the American Statistical Society*, no. 64, pp. 1183-1210, 1969.

[4] I. Bhattacharya and L. Getoor, "Relational Clustering for Multi-type Entity Resolution", In: *Proc. of Workshop on Multi-Relational Data Mining (MRDM)*, 2005.

[5] X. Dong, A. Halevy, and J. Madhavan, "Reference Reconciliation in Complex Information Spaces", In: *Proc. of SIGMOD*. pp. 85-96, 2005.

[6] A. K. Elmagarmid, P. G. Ipeirotis, and V. S. Verykios. "Duplicate record detection: A survey", *IEEE TKDE,* 2007.

[7] A. D. Sarma, A. Jain, A. Machanavajjhala, and P. Bohannon, "*CBLOCK: An Automatic Blocking Mechanism for Large-Scale De-duplication Tasks*", CoRR abs/1111.3689, 2011.

[8] F. Yan, and Y. Tan, "A method of object-based de-duplication," *Journal of networking Netw*, vol. 6, no. 12, pp. 1705-1712, 2011.

[9] T. Yamamuro, Y. Suga, N. Kotani, T. Hitaka, and M. Yamamuro, "Buffer cache de-duplication for query dispatch in replicated databases," *DASFAA*, pp. 352-366, 2011.

[10] T. Pinhas, D. Tsur, S. Zakov, and M. Ziv-Ukelson, "Edit distance with duplications and contractions revisited," *CPM*, pp. 441-454, 2011.

[11] M. Nguyen, and A. Tjoa, "Zero-Latency Data Warehousing (ZLDWH): the State-of-the-art and Experimental Implementation Approaches," In: *Proc. of the 4th IEEE Int'l Conf. on Computer Science, Research, Innovation, and Vision for the Future*, 2006, pp. 166-175.

[12] S. Brobst, "*Active Data Warehousing - A New Breed of Decision Support, Keynote Speech at the Intl. Workshop on Very Large Data Warehouse*," Aix-en-Provence, France, Sept. 2002.

[13] T. Thalhammer, M. Schrefl, and M. Mohania, "Active Data Warehouses: Complementing OLAP with Analysis Rules," *Data & Knowledge Engineering*", vol. 39, no. 3, pp. 241-269, 2001.

[14] J. Schiefer, J. Jeng, and R. M. Bruckner, "*Managing Continuous Data Integration Flows*", CAiSE Workshops, 2003.

[15] M. Bouzeghoub, E. Fabret, and M. Matulovic-Broque, "Modeling the Data Warehouse Refreshment Process as a Workflow Application" In: *Proc. of DMDW'99, Heidelberg, Germany*, 1999.

[16] Y. Zhuge, H. Garcia-Molina, and J. L. Wiener, "Consistency Algorithms for Multi-Source Warehouse View Maintenance", *Distributed and Parallel Databases, Kluwer,* vol. 6, no. 1, pp. 7-40, 1998.

[17] H. Qu, A. Labrinidis, and D. Mosse, "UNIT: User-centric transaction management in web-database systems", *ICDE* pp. 1-10, 2006.

[18] M. Bateni, L. Golab, M. Hajiaghayi, and H. Karloff, "Scheduling to minimize staleness and stretch in real-time data warehouses", *SPAA*, pp. 29-38, 2009.

[19] J. Song, Y. Bao, and G. Yu, "A multilevel and domain-independent duplicate detection model for scientific database," *WAIM*, pp. 729-741, 2010.

Received: September 22, 2014 Revised: November 30, 2014 Accepted: December 02, 2014