25th Meeting of the Wiesbaden Group on Business Registers
- International Roundtable on Business Survey Frames

Tokyo, 8 – 11 November 2016

Rr. Nefriana, Said Mirza Pahlevi, Irien Kamaratih
Badan Pusat Statistik-Statistics Indonesia
Session No. 5

Technology

Tuning Statistical Business Register System Matching Feature

1 Introduction

In November 2013 Indonesia SBR system development was started. It is a web based system that developed using PHP and backed by Microsoft SQL Server. One of the prominent features provided by the system is its matching function. The matching function is necessary in the SBR system because there is no single unique business ID in Indonesia, and hence the function is used to avoid duplication of businesses maintained in the SBR system. It is a feature to find top similar businesses in SBR database from businesses obtained from other sources, mostly from administrative data. In the matching activity, if similar businesses not found in the SBR database, then the business from administrative/new data will be added to the SBR database. Otherwise, it will be regarded as "matched" with a business in the SBR database. If the last happens, operator of the matching activity can replace the old business data with the new one, keep the old business data, or edit the old data. It depends on the quality of each business data. The procedure can be seen in Figure 1.

From 2013 until now, BPS has done two kind of matching activities. The first one was to match between Subject Matter Areas' (SMA) Medium-Large Enterprise (MLE) with Economic Census 2006 business list. The main data in this matching activity was the Economic Census 2006 business list. Therefore, for every business from SMA's MLE the system searched for top 25 to 30 similar businesses in Economic Census 2006 as a single source of truth for SBR. The second kind of matching activity was matching between data BPS provincial offices that had been uploaded, which had been obtained from local administrative data, with SBR data. Therefore, for every business from BPS provincial offices the system searched for top 25 similar businesses in the SBR data.

The problem about those matching activities known when there were complains about why there were more than one same businesses (though the data were slightly different) from the list of top 25 similar businesses. It was suspected that it happened because there was no initial matching activity between businesses in the Economic Census 2006 business list. Later, the suspicion proved true. However, it was known also that there was another issue; when previously operator matching a business, its same business did not appear in the top 25 similar businesses. Hence, the operator decided to add that business from provincial office as a new business in the SBR database while there was already the same business in the SBR database.

Beside that precision issue, SBR system was also complained about the performance. Operators said that they need to wait too long until several minutes to get the top similar businesses from searching results. Consequently, both precision and performance improvements need to be done.

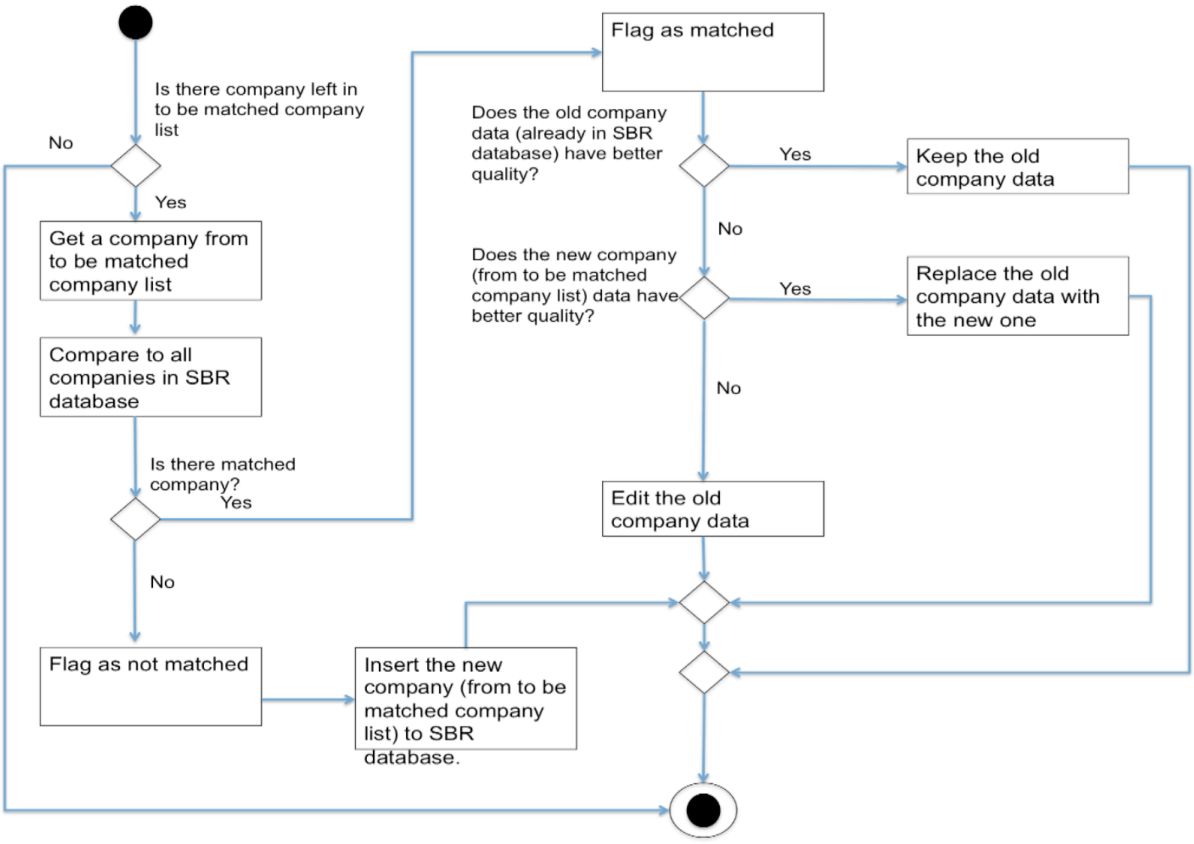

Figure 1: Matching Procedure

2. Tuning Database Architecture

Originally, a SBR guideline from UNECE suggests that all businesses reside in one single table with a column for statistical unit flag. Different from that guidelines, in BPS SBR an enterprise can also have enterprises children. Because of that reason, Indonesia SBR has different approach in the data model. Instead of using only a single table for all businesses with any kind of statistical unit, BPS SBR uses three tables with each table containing one kind of statistical unit plus one table for accommodating relationships between enterprises. This model can be seen in Figure 2.

For each business in New Data table, the system searches for similar businesses in Common tables (Common Establishment, Common Enterprise, and Common Enterprise Group). If there is matched businesses from the similar business list, it will be flagged as matched by operator after checking the contents of both data. If it is matched, user can keep the old data in SBR database, replace the data in Common tables with the new one (from New Data table), or can also edit the data in Common tables with the new data as references. If there is no matched business from Common tables, then simply the business from the New Data will be inserted to one of those Common tables (depends on the statistical unit type) and flagged as a new business.

To deal with that structure, to build SQL query this way is used: calculating smiliarity score for each business in each table and making union those businesses with their similarity scores from the three tables. After that, 25 top similar businesses are selected from that union. Unfortunately, FTS considers size of table for weighting the rank so that enterprise groups, which the number of businesses is the smallest, always get heavier weight. This make enterprise groups tend to be in the smaller ranks even though there are enterprises or establishments that have higher degree of similarity with the keywords. That also happens for enterprises, which the number of the businesses is smaller than the number of businesses in

establishment category. Enterprises always get heavier weight than the establishments. This make enterprises tend to be in the smaller ranks even though there are establishments that have higher degree of similarity with the keywords. To deal with this problem, simple manual weighting cannot be applied with hard coding because the FTS weighting always changes as the number of the businesses in a table changing. Moreover, it is said in its official website of Microsoft that for FTS, "The actual values are unimportant and typically differ each time the query is run. The rank value does not hold any significance across queries." This means that simple weighting with proportion to the size of each table cannot be used either.

Our experiment was to integrate those three Common tables. The new architecture can be seen in Figure 3. A new table was created with business id and other variables that are used for FTS indexing. Meanwhile, the original table were still used for relating feature. Triggers were used to update the new table content when there was change in those three tables. With this new architecture, FTS was only played in one single table. By setting so, it was expected that the FTS weight can be same for all type of statistical unit that businesses have.

For the tuning architecture experiment, we got 400 sample businesses from Common Establishment table. Then, we also added those 400 same businesses to Common Enterprise Table and Common Enterprise Group table. For Common Establishment table, then we added "AA" after the name of each business from those 400 businesses. For Common Enterprise table, we added "BB" after the name of each business from those 400 businesses. Finally for Common Enterprise Group table, we added "CC" after the name of each business from those 400 businesses. With this way, we have very similar businesses among those three tables but different enough so we and the system can differ them. The similarity scores between the cloned businesses will be very close, but if the table weighting plays, the cloned business in Common Enterprise Group table will tend to get additional similarity scores. This is because Common Enterprise Group table has the smallest business number. After that, we run the matching queries 1200 times (400 businesses that have been tripled). We did that twice for each business; before the database tuned and after the database tuned. Because the keywords were derived from the same tables where top 25 similar businesses obtained, ideally those keywords businesses appeared in the first rank of the top 25 similar businesses. We recorded the rank of the keyword companies from 1 to 26 (we recorded the rank as 26^{th} if we did not find the keyword company in the first 25^{th} rank) and also the time required to run the query for each company.

We concluded that the ranking results were improved. Before being tuned, 515 out of 1200 businesses were not in the first rank. After the database architecture being tuned, all of those businesses were in the first rank. We can say that the average distance to the first rank was improved from 0.93 to be 0.00. The average time of running the query was also improved. Before the table being tuned, the average time to run the query was 7.18 seconds and after being tuned it was 5.78 seconds.

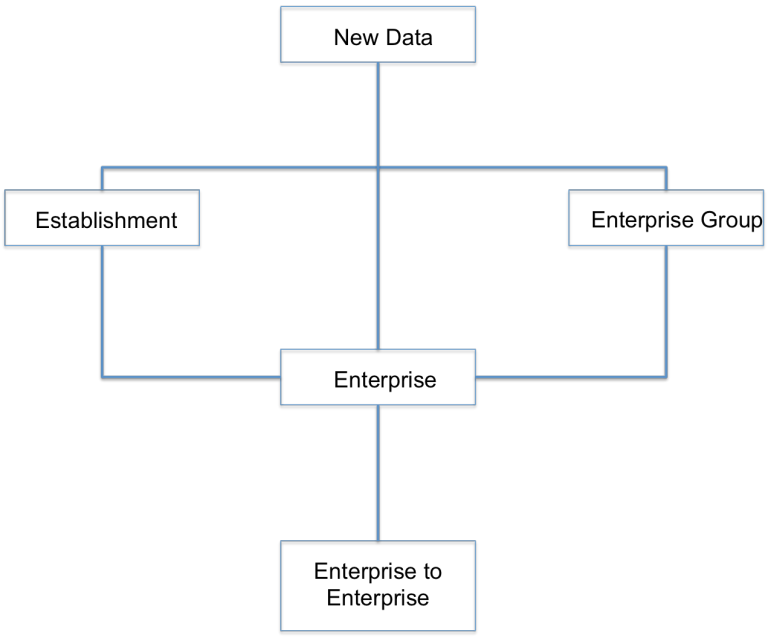

Figure 2. SBR Database Architecture Before Being Tuned

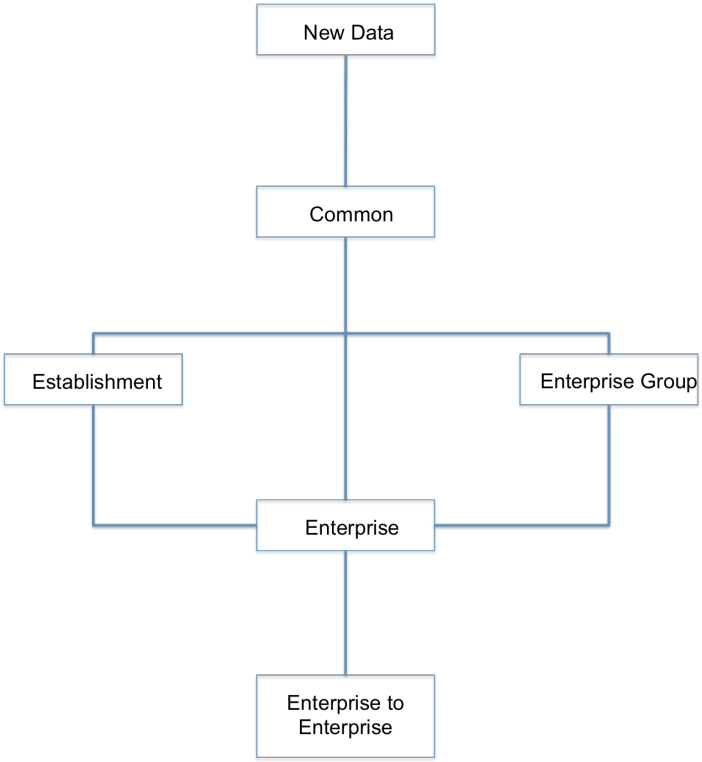

Figure 3. SBR Database Architecture After Being Tuned

3. Removing Stop Words

Parameters that were used to determine the similarity between businesses are business's name, business's commercial name, address, regency code, province code, and country code. For business's name, business's commercial name, and address there can be seen some stop words. A research has been conducted to know the effect of removing those stop words for business' name and adress including kind of accommodation like hotel and villa, kind of businesses, and "street" word (in Indonesian, "street" can be translated as "jalan" which commonly shortened as "JL"). Also certain symbols like hyphen and back slash were taken into account on this experiment.

For the stop words experiment, we got 400 businesses that having those stop words as random samples. We run the matching query with those 400 businesses as the keywords twice; without the stop words removed and with the stop words removed. Same with the previous experiment, because the keywords were derived from the same table where top 25 similar businesses obtained, ideally those keywords businesses appeared in the first rank of the top 25 similar businesses. We recorded the rank of the keyword companies from 1 to 26 (we recorded the rank as 26^{th} if we did not find the keyword company in the first 25^{th} rank) and also the time required to run the query for each company.

From 400 times experiment were done, we found that 22 of the businesses were not in the first rank of query result. After the stopwords were removed and the query was run again, only 14 businesses not in the first rank. We can say that the average distance to the first rank was improved from 1.22 to be 1.12. Beside that, the time performance was also improved. From 400 times experiments the average query time decreased from 11.25 seconds to be 9.59 seconds.

4. Tuning Procedures

SBR system from the beginning of the development have been using nonstored procedures. It was suspected that this was the reason why loading time for matching feature was too long. To know if it was really the reason, that nonstored procedure of matching feature was translated to a stored procedure.

For the stop words experiment, we also got 400 businesses as random samples. We run the matching query with those 400 businesses as the keywords twice; without the procedure tuned and with the procedure tuned. From 400 times experiment were done, we found that the performance was improved after the database tuned from 7.37 seconds to be 3.42 seconds while the ranks for all sample companies were exactly the same between before and after being tuned.

5. Conclusion

There were three efforts that have been done to improve the precision/quality of the searching of top 25 similar businesses and also the performance of the matching feature. From the experiments, it can be concluded that tuning database architecture and removing stop words improve the precision/quality of the searching of top 25 similar businesses and also decrease the processing time. It was also found that tuning query procedure improved query time performance. With these experiments results, we have applied the stop word removal in the SBR system and will use the tuning database architecture and tuning query procedure in the near future.

6. References

[1] United Nations Economic Commission for Europe (UNECE). (2015). *Guidelines on Statistical Business Registers*. New York and Geneva.

[2] Microsoft. *How Search Query Results Are Ranked (Full-Text Search)*. Accessed from https://technet.microsoft.com/en-us/library/ms142524(v=sql.105).aspx.