Title:
Human-powered Sorts and Joins

Summary:
In this paper, the authors proposed a novel query processing system Qurk to address the emerging opportunities in utilizing crowdsourcing in database management systems: firstly, currently there does not exist well-implemented crowd-worker-oriented baseline systems for simple database operations including sorting and joining datasets; secondly, the parameters in querying crowd workers (assignment amount, payment and cooperation of multiple workers) are still not well established. The Qurk system aims at an implementation of a workflow engine for the most important database operators as well as reduce the overall cost by optimization. Using batching and filtering techniques for joining and comparison-based and rating-based techniques for sorting, the system both achieves promising accuracy, alleviates the latency and reduces crowdsourcing cost for joining and sorting operations.

Comments:
For the joining problem, the authors designed a reasonable experiment and convinced that the accuracy of three joining implementations they proposed (simple, naïve and smart) are similar, and the reduction in time consumption and payment cost is significant. In feature filtering, the authors also gave reasonable concerns. For the sorting task, the authors deliberately designed a series of question for the workers to sort, ranging from "sort the squares according to size" to "sort the animals according to the degree to which it belongs to Saturn", each having different openness. By comparing the agreement between sorting and rating methods (tau) and worker agreement (kappa), the authors give a clear picture of the performance of the sorting algorithms.

The authors argued that the performance of crowdsourcing system significantly outperforms that of an average single worker, however this part lacks resources to back up. Personally, I think that the accuracy may actually vary a lot among different workers: for those who are familiar with the images of celebrities from IMDB, they may achieve a better performance than 78% stated in the article.

In the analysis of the results of the joining problem, the authors compared the decrease of accuracy with regard to the amount of assignment for each worker, but did not compare quantitatively the relation between the consequential increased workload, reduced costs and decreased accuracy in three types of batching.

Lastly, the authors may need psychological analysis on the performance of the different implementations to make their explanation more tenable: the naïve batching tend to have lower latency because half of the pair are the same across all the pairs within the batch.

Questions:
What is the point of comparing the performance of naïve and smart joining implementations if no batching is enabled at all? How is the process of identifying features done? (why decide gender, hair and skin? How are features extracted in the first place?)
Title:
Executing SQL over Encrypted Data in the Database-Service-Provider Model

Summary:
This paper stressed the problem of guaranteeing data privacy in the server side and proposed a database system that encrypts accepted and stored data on the server, enabling the server to process the data as encrypted state, leaving the decryption work on the client side, simultaneously minimalizing the necessary data transfer. This is implemented by a mapping and indexing function, and data are translated to an encrypted version (by identification and mapping) before stored and processed on the server. The authors also discussed in detail the mechanism of most SQL operations under this translation system. Experimental results validated the system and illustrated the performance of the system in relation to the scale of data.

Comments:
In the paper, the authors develop their ideas and elucidate the architecture of the system and the algorithms with a consistent example. The notations are developed in a clear way and helps understanding the model and mechanisms. The explanations in sections 3 and 4 are especially detailed – each database operation under the new architecture is elucidated in detail using their well-defined notations.

Given the mapping function, all the numerical values in the original table are replaced by “encrypted” values. The authors did not other data formats: what would the identification function and mapping function be if the values are strings or date format? How do the authors quantitatively evaluate the ambiguity induced with the mapping process, and the consequential increase in computational complexity?

For repeatability concerns, the authors did not specify how the identification function was generated in the experiment section. Also, the paper lacks sufficient discussion on the additional processing, memory and storage expenses under the new architecture. The author should offer a quantitative analysis of the difference between traditional non-encrypted data storage and the translated new data storage schema in terms of validity, data loss (if any) and computational expenses.

The comparison charts presented in section 6 are also specific to the given test dataset (TPC-H) rather than the system itself. Besides the computational complexity in relation to the data scale, the comparison between the new architecture and non-encrypted database should be also provided to ensure novelty and genuineness of the new architecture.

Question:
Does the etuple field contain all the information from all the other (regular) fields? If yes, could the other fields be omitted? If no, what is etuple field used for? How to prevent the server from decrypting the data using the inverse operator, if the cipher technique is known or is recognized by testing?