Learning through play: Iterative query-seeking through partial query specification

CS 598 Project Proposal : Doris Jung-Lin Lee

Motivation:

Formulating ad-hoc database queries is a challenging problem for analysts. Prior work in query by example and query synthesis address this by inferring candidate queries based on input and output data examples [1],[2]. DataPlay has identified that *local syntactical query* and lack of *non-answers* of SQL have made it challenging to iteratively construct quantified queries. They have used a query tree to enable tweaking and auto-correction of a user's intended query [3],[4].

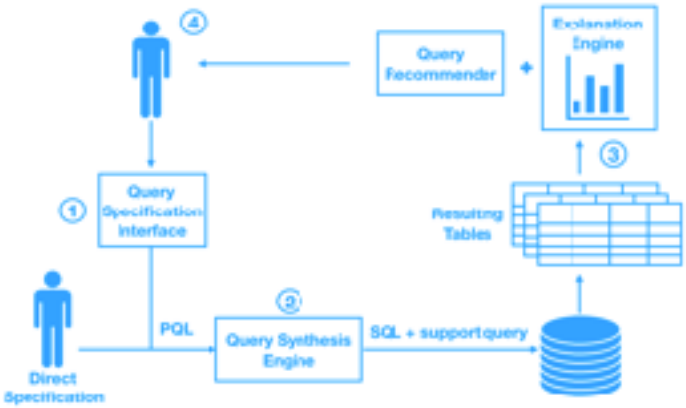
However, the more pressing challenge for both novice and expert analysts is often coming up with the right question to ask. Most of these prior approaches are intent-to-query mapping mechanisms that assume users have a question in mind. They focus on resolving the "language barrier" to help novices unfamiliar with SQL issue queries to the database. Users are often not sure about what types of data operations they should perform on their data or what query they want to find. Moreover, while existing database query interface are capable of synthesizing SQL queries based on high-level specifications, the space of queries that could be issued with these interface are often limited by the set of form fields and interactions envisioned by tool-designers.

Problem:

In this project, we take a principled approach to this problem by designing a querying language that is tolerant to ambiguity, called *partial query language* (PQL). PQL is a querying language that takes in an underspecified query during exploratory data analysis and the system automatically issues the corresponding SQL queries, as well as "support queries" for generating explanation that can provoke further investigation. These underspecified query can include requests for overviews, uncertain attributes selections, or view suggestions within a range query. The goal of PQL is to simultaneously enable flexible querying as well as gain better understanding of the dataset to facilitate users to discover the interesting questions he would like to ask.

System:

Fig. 1: Schematic of a envisioned data playground operating on PQL. Each boxed component is a user-defined "sandbox" that could be customized for individual use cases.



1. A query specification interface that captures this ambiguity is used to specify PQL . The user could by-pass the specification interface and specify in PQL directly if they have something more clear in mind.
2. Through the query synthesis engine, PQL is translated into one or multiple SQL statements along with several "support queries". The support queries are queries used to supply information to the query recommendation and explanation engine, which provides support information that enable the user to refine his question.
3. The explanation engine serves to inform the users about the result of his specified PQL. These explanations can include histograms of data distributions or overview statistics computed based on the resulting tables. The explanation engine could be optionally coupled with an approximate query processing engine to provide feedback for refinements at interactive speeds. The query recommender searches through semantically local queries to the PQL issued.
4. The user receives recommendations and explanations regarding his query results and makes refinements to his PQL query.

Project Plan:

As a first step to building the data playground envisioned in Fig. 1, my project this semester will focus on the design specification of PQL and a proof-of-concept implementation of the query synthesizer, recommender and explanation engine.

References:

[1] Li, H., Chan, C.-Y., & Maier, D. (2015). Query from examples: An iterative, data-driven approach to query construction. Proceedings of the VLDB Endowment, 8(13), 2158–2169. http://doi.org/10.14778/2831360.2831369

[2] Wang, C., Cheung, A., & Bodik, R. (2017). Synthesizing highly expressive SQL queries from input-output examples. Proceedings of the 38th ACM SIGPLAN Conference on Programming Language Design and Implementation - PLDI 2017, 452–466. http://doi.org/10.1145/3062341.3062365

[3] Abouzied, A., & Hellerstein, J. M. (2012). Playful Query Specification with DataPlay. Proceedings of the 38th International Conference on Very Large Data Bases, 1938–1941. http://doi.org/10.14778/2367502.2367542

[4] Abouzied, A., Hellerstein, J., & Silberschatz, A. (2012). Dataplay: interactive tweaking and example-driven correction of graphical database queries. Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology, 207–217. http://doi.org/10.1145/2380116.2380144