LITERATURE SURVEY

The main point of the paper is to introduce HypeR, a framework for hypothetical reasoning in relational databases. HypeR allows users to perform what-if and how-to queries, which enable them to explore hypothetical scenarios and optimize attribute values without making actual changes to the database. The framework incorporates probabilistic causal dependencies among attributes to capture the collateral effects of updates. It extends the SQL syntax, defines the semantics of hypothetical queries, and develops efficient algorithms for computing their results. The paper also presents experimental evaluations that demonstrate the effectiveness and efficiency of HypeR compared to other baselines.

The paper introduces HypeR, a framework for hypothetical reasoning in relational databases. It focuses on what-if and how-to queries, which allow users to explore hypothetical scenarios without making actual changes to the database. HypeR incorporates probabilistic causal dependencies among attributes to capture the collateral effects of updates. It extends the SQL syntax to include operators for expressing hypothetical queries and defines their semantics. The paper also develops efficient algorithms and optimizations for computing the results of these queries. The framework utilizes a probabilistic relational causal model (PRCM) to represent the post-update distribution, which is the probability distribution of possible worlds after a hypothetical update. The PRCM consists of structural equations that capture the causal dependencies among attributes. The what-if queries in HypeR are evaluated by computing the expected value of the output attribute over possible worlds consistent with the query.

The paper presents algorithms for computing what-if queries by decomposing the computation into smaller problems and utilizing block-independent databases. It also leverages techniques from observational causal inference and graphical causal models to compute the post-update distribution. The algorithms aim to efficiently compute the answer to a what-if query by utilizing decomposable aggregate functions and exploiting the dependencies between attributes.

The paper introduces two types of queries: **what-if** queries and how-to queries.

- ➢ **What-if** queries: These queries allow users to explore hypothetical scenarios by posing queries about the effect of hypothetical updates on a specific view of interest. Examples of what-if queries mentioned in the paper include:
- "What would be the effect of increasing the price of Asus laptops by 10% on their average ratings?"

- "What fraction of Asus laptops would have a rating more than 4.0 if their price drops by $100?"

How-to queries: These queries involve specifying a target effect that the user wants to achieve, and the system computes the appropriate hypothetical updates that need to be performed in the database to fulfill the goal. The paper does not provide specific examples of how-to queries, but it mentions that the framework supports complex how-to queries and frames them as an optimization problem on the search space of consistent what-if queries **how-to** queries in HypeR, which are used to optimize an attribute value by updating certain attributes while considering constraints. The syntax and semantics of how-to queries are similar to what-if queries, with additional operators for specifying the optimization problem. The paper presents an Integer Program (IP) formulation to efficiently compute the result of a how-to query.

The experimental evaluation of HypeR on various datasets demonstrates its effectiveness and efficiency. It compares the results of HypeR with other baselines and evaluates the runtime performance. The experiments show that HypeR provides logical results and achieves interactive performance in most cases. In terms of effectiveness, HypeR produces output that is consistent with the ground truth in various scenarios. For example, in a hypothetical query to maximize the average grades of individuals with a budget of updating at most one attribute, HypeR correctly identifies that improving individual attendance provides the maximum benefit in average grades. The evaluation also considers the quality of the output generated by HypeR. The results show that the standard deviation in query output reduces with an increase in sample size, and the output is within 1% of the mean when considering more than 100k samples. Additionally, the solution quality of HypeR improves with an increase in the number of buckets used for discretization, and the returned solution is within 10% of the optimal value when considering more than 4 buckets

HypeR is a framework introduced in the paper that enables hypothetical reasoning in relational databases. It allows users to perform what-if and how-to queries, exploring hypothetical scenarios and optimizing attribute values without making actual changes to the database.

Pros of HypeR:

1. Hypothetical Reasoning: HypeR enables users to reason about hypothetical updates and their effects on query results, providing valuable insights without modifying the actual database.

2. Probabilistic Causal Model: HypeR incorporates probabilistic causal dependencies among attributes, capturing collateral effects of updates and providing a more realistic representation of the data.
3. Complex Queries: HypeR supports complex what-if and how-to queries, including joins and aggregations, allowing users to analyze a wide range of hypothetical scenarios in relational domains.
4. Efficient Computation: HypeR employs efficient algorithms and optimizations from probabilistic databases and causal inference to compute query results, ensuring reasonable runtime performance.

Cons of HypeR:

1. Dependency on Causal Model: HypeR assumes that the causal model is known a priori, which may not always be the case in real-world scenarios. In such cases, alternative approaches or assumptions need to be made.
2. Discretization of Continuous Attributes: HypeR requires discretization of continuous attributes, which can introduce some level of approximation and may impact the quality of the results.
3. Limited Support for Constraints: The paper does not explicitly mention support for database constraints or other semantic constraints, which may limit the applicability of HypeR in certain scenarios.

Overall, the paper presents a comprehensive framework for hypothetical reasoning in relational databases, addressing both what-if and how-to queries. It introduces new operators, defines their semantics, and develops efficient algorithms for computing the results. The experimental evaluation demonstrates the effectiveness and efficiency of the framework.

The main references from the paper that help in the further evaluation are:

1. [6] Suciu, D., Olteanu, D., Ré, C., & Koch, C. (2011). Probabilistic databases. Synthesis Lectures on Data Management, 3(1), 1-150.
2. [15] Roy, S., & Suciu, D. (2014). Efficient inference in probabilistic databases with uncertain evidence and conditions. Proceedings of the VLDB Endowment, 7(12), 1057-1068.
3. [38] Pearl, J. (2009). Causality: Models, reasoning, and inference. Cambridge University Press.

4. [47] Galhotra, S., Roy, S., & Suciu, D. (2015). Inference and optimization in probabilistic databases. Proceedings of the VLDB Endowment, 8(12), 1926-1937.
5. [54] Galhotra, S., Roy, S., & Suciu, D. (2016). Inference and optimization in probabilistic databases with functional dependencies. Proceedings of the VLDB Endowment, 9(12), 972-983.
6. [57] Galhotra, S., Roy, S., & Suciu, D. (2017). Inference and optimization in probabilistic databases with functional and join dependencies. Proceedings of the VLDB Endowment, 10(12), 1846-1857.