
# Explaining Differentially Private Query Results With DPXPlain

Tingyu Wang
Duke University
tw296@cs.duke.edu

Yuchao Tao
Duke University
yctao@cs.duke.edu

Amir Gilad*
Hebrew University
amirg@cs.huji.ac.il

Ashwin Machanavajjhala
Duke University
ashwin@cs.duke.edu

Sudeepa Roy
Duke University
sudeepa@cs.duke.edu

## ABSTRACT

Employing Differential Privacy (DP), the state-of-the-art privacy standard, to answer aggregate database queries poses new challenges for users to understand the trends and anomalies observed in the query results: Is the unexpected answer due to the data itself, or is it due to the extra noise that must be added to preserve DP? We propose to demonstrate DPXPlain, the first system for explaining group-by aggregate query answers with DP. DPXPlain allows users to compare values of two groups and receive a validity check, and further provides an explanation table with an interactive visualization, containing the approximately 'top-k' explanation predicates along with their relative influences and ranks in the form of confidence intervals, while guaranteeing DP in all steps.

**PVLDB Reference Format:**
Tingyu Wang, Yuchao Tao, Amir Gilad, Ashwin Machanavajjhala, and Sudeepa Roy. Explaining Differentially Private Query Results With DPXPlain. PVLDB, 16(12): 3962 - 3965, 2023.
doi:10.14778/3611540.3611596

**PVLDB Artifact Availability:**
The source code, data, and/or other artifacts have been made available at https://github.com/yuchaotao/Private-Explanation-System.

## 1 INTRODUCTION

*Differential Privacy (DP)* [6] is the gold standard for protecting privacy in query processing and is critically important for sensitive data analysis. It has been widely adopted by organizations like the U.S. Census Bureau and companies like Google, Microsoft, and Apple. The core idea behind DP is that a query answer on the original database cannot be distinguished from the same query answer on a slightly different database. This is usually achieved by adding random noise to the query answer to create a small distortion in the answer. Recent works have made significant advances in the usability of DP, allowing for complex query support [2, 9, 12], and employing DP in different settings.

Automatically generating meaningful *explanations* for query answers in response to questions asked by users is an important step in data analysis that can significantly reduce human efforts. Explanations help users validate query results, understand trends and anomalies, and make decisions about next steps regarding data processing and analysis, thereby facilitating data-driven decision making. Several approaches for explaining aggregate and non-aggregate query answers have been proposed in database research, including intervention [10, 13], Shapley values, counterbalance, (augmented) provenance, responsibility, and entropy (details in [11]).

One major gap that remains wide open is to provide explanations for analyzing query answers from sensitive data under DP. Several new challenges arise from this need. First, in DP, the (aggregate) query answers are distorted due to the noise that must be added for preserving privacy, so the explanations need to separate the contributions of the noise from the data. Second, even after removing the effect of noise, new techniques have to be developed to provide explanations based on the sensitive data and measure their effects. Third, the system needs to ensure that the returned explanations, scores, and ranks still have high accuracy while being private.

Therefore, we propose to demonstrate DPXPlain[1], a novel system that can augment aggregate queries with explanations while satisfying DP. Our approach consists of three phases of explanations and employs the notion of interventions. We next illustrate the steps of DPXPlain through a detailed example.

Example 1.1. *Consider the* `Adult` *(a subset of Census) dataset [3] with 48,842 tuples. We consider the following attributes:* `age`, `workclass`, `education`, `marital-status`, `occupation`, `relationship`, `race`, `sex`, `native-country`, and `high-income`, *where* `high-income` *is a binary attribute indicating whether the income of a person is above 50K or not; some relevant columns are illustrated in Figure 1a. In Phase-1, the user submits a query and gets the results as shown in Figure 1b. This query is asking the fraction of people with high income in each* `marital-status` *group. As Figure 1b shows, the framework returns the answer with two columns:* `group` *and* `Priv-answer`. *Here* `group` *corresponds to the group-by attribute* `marital-status`. *However, since the data is private, instead of seeing the actual aggregate values* `avg-high-income`, *the user sees a perturbed answer* `Priv-answer` *for each group as output by some DP mechanism with a given privacy budget (here computed by the Gaussian mechanism with privacy budget* $\rho = 0.1$ *[1]). The third column* `True-answer` *shown in grey (hidden for users) in Figure 1b shows the true aggregated output for each group. In Phase-2, the user selects two groups to compare their aggregate values and asks for explanations. However, unlike standard explanation frameworks [8, 10, 13] where the answers to a query are correct and hence the question asked by the user is also correct, in the DP setting, the answers that the users see are perturbed. Therefore, the user question and the direction of comparison may not be valid. Hence our system first tests the validity of the question. Consider*

*Work done during postdoc at Duke University.


Proceedings of the VLDB Endowment, Vol. 16, No. 12 ISSN 2150-8097.
doi:10.14778/3611540.3611596

[1]The research paper that presents the complete technical details behind DPXPlain has been published in PVLDB Vol 16(1), 2022, and will be presented at VLDB 2023 [11].

| marital-status | occupation | ... | education | high-income |
|---|---|---|---|---|
| Never-married | Machine-op-inspct | ... | 11th | 0 |
| Married-civ-spouse | Farming-fishing | ... | HS-grad | 0 |
| Married-civ-spouse | Machine-op-inspct | ... | Some-college | 1 |
| ... | ... | ... | ... | ... |

**(a) Example of the `Adult` dataset.**

**Question-Phase-1:**
SELECT `marital-status`, AVG(`high-income`) as `avg-high-income` FROM Adult GROUP BY `marital-status`;

**Answer-Phase-1:**

| group marital-status | Priv-answer avg-high-income | True-answer (hidden) |
|---|---|---|
| Never-married | 0.045511 | 0.045480 |
| Separated | 0.064712 | 0.064706 |
| Widowed | 0.082854 | 0.084321 |
| Married-spouse-absent | 0.089988 | 0.092357 |
| Divorced | 0.101578 | 0.101161 |
| Married-AF-spouse | 0.463193 | 0.378378 |
| Married-civ-spouse | 0.446021 | 0.446133 |

**(b) Phase-1 of** DPXPlain**: Run a query and receive noisy answers by DP. True-answers are not visible to the user and for illustration only.**

**Question-Phase-2:** Why `avg-high-income` of group **`"Married-civ-spouse"`** > that of group **`"Never-married"`**?

**Answer-Phase-2:** The 95% confidence interval of group difference is $(0.399, 0.402)$, hence the noise in the query is possibly not the reason.

**(c) Phase-2 of** DPXPlain**: Ask a comparison question and receive a confidence interval of the comparison.**

**Answer-Phase-3:**

| explanation predicate | Rel Influ 95%-CI L | Rel Influ 95%-CI U | Rank 95%-CI L | Rank 95%-CI U |
|---|---|---|---|---|
| `occupation = "Exec-managerial"` | 3.25% | 10.12% | 1 | 9 |
| `education = "Bachelors"` | 2.93% | 9.80% | 1 | 8 |
| `age = "(40, 50]"` | 2.76% | 9.63% | 1 | 8 |
| `occupation = "Prof-specialty"` | 0.94% | 7.81% | 1 | 18 |
| `relationship = "Own-child"` | -0.49% | 6.38% | 1 | 96 |

**(d) Phase-3 of** DPXPlain**: Receive an explanation table from data for the previous question that passed Phase-2.**

**Figure 1: Database instance and the three phases of the** DPXPlain **framework [11].**

*the comparison between the two groups* `"Never-married"` *and* `"Married-civ-spouse"`*, in Figure 1c. In this case, the confidence interval of the difference does not include 0 and is tight around the positive number 0.4, indicating that the user question is correct with high probability. It is still possible for a valid question to have a confidence interval that includes zero given sufficiently large noise. Since the question is valid, the user may continue to the next phase.*

*In Phase-3, for the questions that are likely to be valid,* DPXPlain *can provide a further detailed data-dependent explanation for the question. To achieve this again with DP, our framework reports an "Explanation Table" to the user as Figure 1d shows, which includes the top-5* explanation predicates*. The explanation predicates explain the user question using the notion of* intervention *as done in previous work [10, 13] for explaining aggregate queries in the non-DP setting. Intuitively, if we intervene in the database by (hypothetically) removing tuples that satisfy the predicate, and re-evaluating the query, then the difference in the aggregate values of the two groups mentioned in the question will reduce. In the simplest form, explanation predicates are singleton predicates of the form "*`attribute = <value>`*", while in general, our framework supports more complex predicates involving conjunction, disjunction, and comparison (>, ≥ etc.). In Figure 1d, the top-5 explanation predicates computed by* DPXPlain *are shown out of 103 singleton predicates, based to their influences on the question but perturbed by noises to satisfy DP. The amount of noise is proportional to the sensitivity of the influence function, the maximum possible change of the influence of any explanation predicate when adding or removing a single tuple from the database. Once the top-5 predicates are selected, the explanation table also shows their relative influence (intuitively, how much they affect the difference of the group aggregates in the question) and their ranks (that might differ from the true top-5) in the form of DP confidence intervals.*

*We see that* `occupation = "Exec-managerial"` *is returned as the top explanation predicate, indicating that the people with this job contribute more to the average high income of the married group compared to the never-married group. That is, managers tend to earn more if they are married than those who are single, which can likely be attributed to the intuition that married people might be older and have more seniority, which is consistent with the third explanation* `age = "(40, 50]"` *in Figure 1d as well. Although these explanations are chosen at random, we observe that the first three explanations are almost constantly included. This is consistent with the narrow confidence interval of rank for the first three explanation predicates, which are all around* $[1, 8]$*. Looking at the confidence intervals of the relative influence and ranks in the explanation table, the user also knows that the first three explanations are likely to have some effect on the difference between the married and unmarried groups. However, for the last two explanations, the confidence intervals of influences are closer to 0 and the confidence intervals of ranks are wider, especially for the fifth one which includes negative influences in the interval and has a wide range of possible ranks (96 out of 103 simple explanation predicates in total).*

We will demonstrate DPXPlain using the `Adult` dataset from the example, along with the real-world dataset `IPUMS-CPS` from Census and a synthetic dataset `Greman-Credit` in an interactive user interface. The visualizations in the graphical user interface added to DPXPlain make the explanations provided by DPXPlain more understandable and make the concept of query explanations with DP using the three-phase framework accessible to non-expert users. In particular, we draw the confidence intervals and plot circles for the 'true values' (available to admins with access to true data, but hidden from end users who only have access to the noisy query answers) to highlight whether the confidence intervals contain the true values and how wide they are.

## 2 TECHNICAL BACKGROUND

We now give the necessary background for our model. The database schema $\mathbb{A} = (A_1, \ldots, A_m)$ is a vector of attributes of a single relational table. Each attribute $A_i$ is associated with a domain $\mathtt{dom}(A_i)$, which can be continuous or categorical. A database (instance) $D$ over a schema $\mathbb{A}$ is a bag of tuples (duplicate tuples are allowed)

$t_i = (a_1, \ldots, a_m)$, where $a_i \in \texttt{dom}(A_i)$ for all $i$. The domain of a tuple is denoted as $\texttt{dom}(\mathbb{A}) = \texttt{dom}(A_1) \times \texttt{dom}(A_2) \times \ldots \times \texttt{dom}(A_m)$. The value of the attribute $A_i$ of tuple $t$ is denoted by $t.A_i$.

We consider group-by aggregate queries $q$ of the form: $q =$ `SELECT` $A_{gb}$`, agg(`$A_{agg}$`) FROM D WHERE` $\phi$ `GROUP BY` $A_{gb}$`;`. Here, $A_{gb}$ is the group-by attribute and $A_{agg}$ is the aggregate attribute, $\phi$ is a predicate without subqueries, and `agg` $\in \{COUNT, SUM, AVG\}$ is the aggregate function.

**Differential Privacy.** We consider query-answering and explanations generation using *differential privacy (DP)* [7] to protect private information. In standard databases, a query result can give an adversary the option to find the presence or absence of an individual in the database, compromising their privacy. DP allows users to query the database without compromising the privacy by guaranteeing that the query result will not change much when it is evaluated on any two *neighboring* databases, i.e., two databases $D$ and $D'$ such that $D'$ can be transformed from $D$ by adding or removing a tuple. We use a relaxation of DP called **$\rho$-zero-concentrated differential privacy ($\rho$-zCDP)** [1], *and refer to it simply as DP if not otherwise stated.* A lower $\rho$ value implies a lower privacy loss, therefore it is considered as a privacy budget. A popular approach for providing $\rho$-zCDP to a query result is to add Gaussian noise to the result before releasing it to a user. This approach is called *Gaussian mechanism* [7]. The privacy guarantee from the Gaussian mechanism depends on both the noise scale it uses and the sensitivity of the query. Query sensitivity reflects how sensitive the query is to the change of the input. More noise is needed for a more sensitive query to achieve the same level of privacy protection.

**Private Query Answering.** For a $COUNT$ or $SUM$ query, we use the Gaussian mechanism for each group. For a $AVG$ query, since $AVG = SUM/COUNT$, we decompose it into a $SUM$ and a $COUNT$ query, privately answer each of them using the Gaussian mechanism for each group, and release the division.

**Confidence Level and Interval.** Confidence intervals are commonly used to determine the error margin in uncertain computations and are used in various fields including machine learning and DP. In our context, we use confidence intervals to measure the uncertainty in the user question and our explanations.

**User Question and Standard Explanation Framework.** In Phase-2 of DPXPlain, given the noisy results of a group-by aggregation query from Phase-1, users can ask questions comparing the aggregate values of two groups. To explain a user question, several previous approaches return top-k predicates that have the most influences to the group difference in the question as explanations [10, 13]. We follow this paradigm and define explanation predicates as Boolean expressions of the form $p = \varphi_1 \wedge \ldots \wedge \varphi_l$, where each $\varphi_i$ has the form $A_i = a_i$ such that $A_i \in \mathbb{A} \setminus \{A_{gb}, A_{agg}\}$ is an attribute, and $a_i \in \texttt{dom}(A_i)$ is its value. More details can be found in [11].

## 3 SYSTEM IMPLEMENTATION: DPXPLAIN

The graphical user interface of DPXPlain has five pages: landing, data, query, result, and explanation. It is built using a Vue-based UI library Element, where the query highlights are via highlight.js, tooltips are via CodeMirror, visualization grpahs are via Apache ECharts and databases via PostgreSQL. The algorithms used to privately compute the query answers and explanations are implemented in Python 3.9 using the Pandas, NumPy, and SciPy libraries.

**Admin and user mode.** We assume the user of DPXPlain can be an *admin* or an *end user*. The main functionalities of these two modes are the same, except that an admin is also allowed to see the *true* query answers and compare with noisy query answers, whereas an end user only has access to the noisy answers. Below we describe the interactions for the admin mode.

**Aggregate query computation.** Given a group-by aggregate query written by the user and a privacy budget, DPXPlain will run the query using PostgreSQL and return the *private* results to the user. These results have noise added to them and are therefore distorted. The results screen also features a button that allows admins to view the hidden true results (will not be shown to end users).

**Initial explanation generation.** Users can add a question about a comparison between two groups in the given results view and give a confidence level for the explanation. In response, DPXPlain generates a confidence interval that suggests where the difference between the two groups lies. It does so by applying the confidence interval of the Gaussian distribution and the union bound rule.

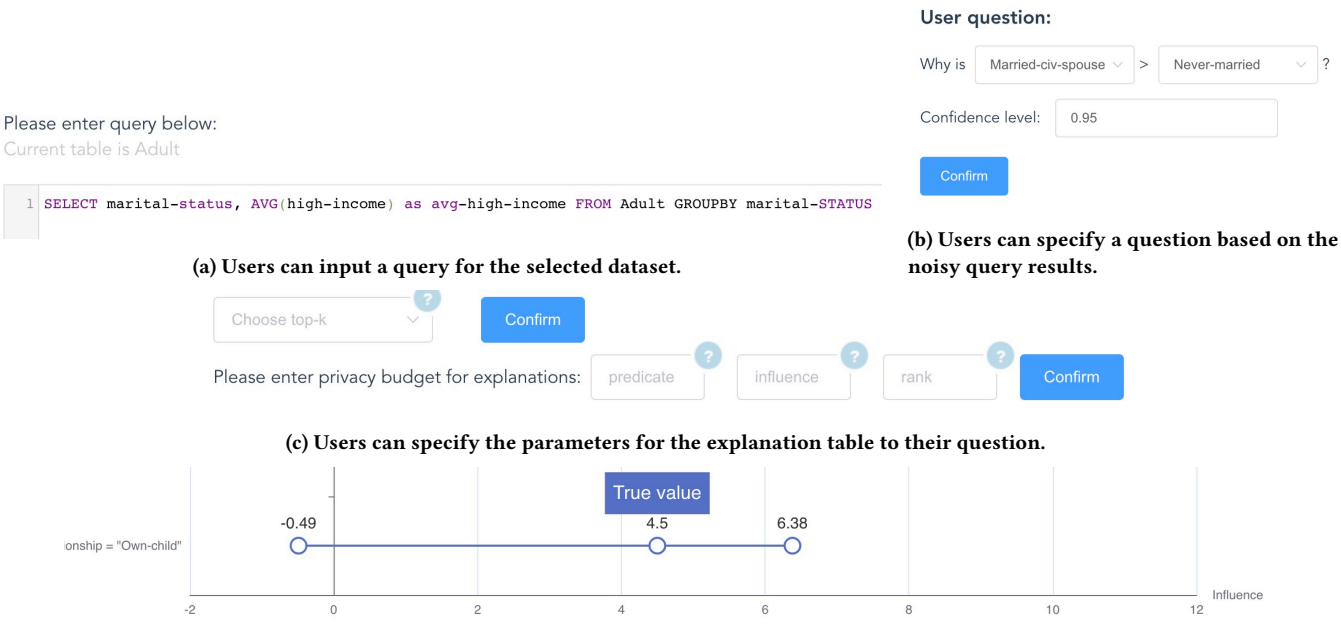
**Explanation table computation.** Users can proceed to spend more privacy budget to get an explanation table for the question. DPXPlain uses the **One-shot Top-k mechanism** [4, 5] to privately select top-$k$ explanation predicates, ranked by their influences on the user question. In [11], we devise a novel influence function with low sensitivity, which allows us to get accurate estimations for the predicate influence despite the added noise.

**Visualization.** We visualize the explanation table by drawing the confidence intervals of relative influences and ranks for each explanation predicate with different colors, and (for admins) adding the true relative influence and rank floating as circle markers to the lines to indicate if the confidence interval includes the true value.

## 4 DEMO SCENARIO

We will demonstrate DPXPlain using the `Adult` dataset containing details about individuals, including their personal and employment information. We will also include the `IPUMS-CPS` dataset from Census and the `Greman-Credit` dataset for users to experiment with. The demonstration will include the following steps.

**Guiding users through** DPXPlain**.** We will first help the users get familiar with the interfaces and the entire flow of the system. The users will be first shown with several rows of the `Adult` dataset, while allowing them to explore and to sort different columns. We then move on to the next page and show users how to write an aggregation SQL query using the example in Figure 1a. Here we will explain the format of the queries that the system can support, which includes $COUNT$, $SUM$, and $AVG$ with predicates and group-by on multiple groups. We then explain differential privacy to the users and why we need to specify a privacy budget to show the query result. In the next page, a noisy query result will be shown to the user, and we will guide the users to click the "Show True Results" to reveal the true results to help understand the effect of DP. In this page, users can explore the query answers, and we will prepare a question and enter a confidence level to the system, so the users can follow the question to understand how the system will explain the question raised from the noisy query results. In the next page, the system will present an initial explanation, and then we will show how to input the remaining parameters to present the explanation table, and how to understand the visualization.




**(a) Users can input a query for the selected dataset.**

**(b) Users can specify a question based on the noisy query results.**

**(c) Users can specify the parameters for the explanation table to their question.**

**(d) The visualization of the confidence intervals of the relative influences.**

**Figure 2: Major interfaces of DPXPLAIN (admin mode with true query answers).**

**Dataset selection and query formulation.** After guiding the users through the entire process, they will start with dataset selection and data exploration. After clicking "Enter for query", users can enter the query in the query box (Figure 2a), specify the privacy budget, and finally click "Run Query". If the users have no clue to choose a proper privacy budget, one can click "Choose a budget for me" to pick a random number between $[0.1, 5]$.

**Analysis of the results and initial question.** In the next page, DPXPLAIN will output a table of query results, with one row for each group. Users can click "Show True Result" or "Hide True Results" to toggle the display of the true results. Users can analyze the results to find questions of the form "Group A" larger than "Group B", specify a confidence level for the system to find a corresponding level of confidence interval for the difference between two groups and finally click "Confirm" (Figure 2b).

**Question validity check and explanation parameters.** After clicking "Confirm", DPXPLAIN will present an initial explanation by a question validity check in a new page. Users are then prompted with an option "Proceed to predicate explanations?". If confirmed, users will input the $k$ value for the top-$k$ explanations, and three budgets for privately choosing top-$k$ explanations, privately finding the 95%-level confidence intervals of their relative influences and 95%-level confidence levels of their ranks (Figure 2c). Users can click the "?" marks for more tips.

**Examining the explanation table and visualizations.** The explanation table will be shown below once the users click "Confirm" to submit the parameters. Each row of the explanation table contains an explanation predicate, the confidence intervals of their relative influences and ranks. Users can explore this table, or scroll down to inspect the corresponding visualizations to the table (Figure 2d). The visualization also comes with a button "Show/Hide True Value" for the true relative influence or rank. Users can click "Back to Question" or "Back to Query" in this page at any time to repeat the process and explore other queries or questions.

## ACKNOWLEDGMENTS

This work was supported by the NSF awards IIS-2147061, IIS-2016393, IIS-2008107, IIS-1703431, and IIS-1552538.

## REFERENCES

[1] M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.
[2] W. Dong, J. Fang, K. Yi, Y. Tao, and A. Machanavajjhala. R2t: Instance-optimal truncation for differentially private query evaluation with foreign keys. In *SIGMOD*, pages 759–772, 2022.
[3] D. Dua and C. Graff. UCI machine learning repository, 2017.
[4] D. Durfee and R. Rogers. One-shot dp top-k mechanisms. DifferentialPrivacy.org, 08 2021. https://differentialprivacy.org/one-shot-top-k/.
[5] D. Durfee and R. M. Rogers. Practical differentially private top-k selection with pay-what-you-get composition. In *NeurIPS*, pages 3527–3537, 2019.
[6] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.
[7] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.
[8] K. El Gebaly, P. Agrawal, L. Golab, F. Korn, and D. Srivastava. Interpretable and informative explanations of outcomes. *Proc. VLDB Endow.*, 8(1):61–72, sep 2014.
[9] I. Kotsogiannis, Y. Tao, X. He, M. Fanaeepour, A. Machanavajjhala, M. Hay, and G. Miklau. Privatesql: a differentially private sql query engine. *Proc. VLDB Endow.*, 12(11):1371–1384, 2019.
[10] S. Roy and D. Suciu. A formal approach to finding explanations for database queries. In *SIGMOD*, pages 1579–1590, 2014.
[11] Y. Tao, A. Gilad, A. Machanavajjhala, and S. Roy. Dpxplain: Privately explaining aggregate query answers. *Proc. VLDB Endow.*, 16(1):113–126, 2022.
[12] Y. Tao, X. He, A. Machanavajjhala, and S. Roy. Computing local sensitivities of counting queries with joins. In *SIGMOD*, pages 479–494, 2020.
[13] E. Wu and S. Madden. Scorpion: Explaining away outliers in aggregate queries. *Proc. VLDB Endow.*, 6(8):553–564, 2013.