DPXPlain: Privately Explaining Aggregate Query Answers

Yuchao Tao, Amir Gilad, Ashwin Machanavajjhala, Sudeepa Roy
Duke University, USA

ABSTRACT

Differential privacy (DP) is the state-of-the-art and rigorous notion of privacy for answering aggregate database queries while preserving the privacy of sensitive information in the data. In today's era of data analysis, however, it 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? In the second case, even the observation made by the users on query results may be wrong. In the first case, can we still mine interesting explanations from the sensitive data while protecting its privacy? To address these challenges, we present a three-phase framework DPXPlain, which is the first system to the best of our knowledge for explaining group-by aggregate query answers with DP. In its three phases, DPXPlain (a) answers a group-by aggregate query with DP, (b) allows users to compare aggregate values of two groups and with high probability assesses whether this comparison holds or is flipped by the DP noise, and (c) eventually provides an explanation table 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. We perform an extensive experimental analysis of DPXPlain with multiple use-cases on real and synthetic data showing that DPXPlain efficiently provides insightful explanations with good accuracy and utility.

PVLDB Reference Format:
Yuchao Tao, Amir Gilad, Ashwin Machanavajjhala, Sudeepa Roy
Duke University, USA. DPXPlain: Privately Explaining Aggregate Query Answers. PVLDB, 14(1): XXX-XXX, 2020.
doi:XX.XX/XXX.XX

1 INTRODUCTION

Differential privacy (DP) [17, 42–44] 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 [3, 40, 63, 89] and companies like Google [48, 102], Microsoft [33], and Apple [94]. 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 [36, 60, 64, 65, 74, 95, 102], and employing DP in different settings [36, 49, 52, 82, 95, 104]. These works assist in bridging the gaps between the functionality of non-DP databases and databases that employ DP.

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 and assist users. 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 [86, 87, 103], Shapley values [72], counterbalance [80], (augmented) provenance [6, 70], responsibility [78, 79], and entropy [47] (discussed in Section 6).

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 shown to users 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. For instance, standard explanations methods in non-DP settings are typically deterministic, while it is known that DP methods must be randomized. Therefore, no deterministic explanations can be provided, even no deterministic scores or ranks of explanations can be displayed in response to user questions if we want to guarantee DP in the explanation system. Third, the system needs to ensure that the returned explanations, scores, and ranks still have high accuracy while being private.

In this paper, we propose DPXPlain, a novel three-phase framework that generates explanations under DP for aggregate queries based on the notion of *intervention* [87, 103]. DPXPlain surmounts the aforementioned challenges and is the first system combining DP and explanations to the best of our knowledge. We illustrate DPXPlain through an example.

Example 1.1. Consider the `Adult` (a subset of Census) dataset [37] 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 the **first phase (Phase-1)** of DPXPlain, 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

Proceedings of the VLDB Endowment, Vol. 14, No. 1 ISSN 2150-8097.
doi:XX.XX/XXX.XX

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.

aggregate values avg-high-income, the user sees a perturbed answer Priv-answer for each group as output by some differentially private mechanism with a given privacy budget (here computed by the Gaussian mechanism with privacy budget $\rho = 0.1$ [17]). The third column True-answer shown in grey (hidden for users) in Figure 1b shows the **true aggregated output** for each group.

In the **second phase (Phase-2)** of DPXPlain, the user selects two groups to compare their aggregate values and asks for explanations. However, unlike standard explanation frameworks [47, 70, 80, 87, 103] where the answers of 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. If the question is valid, our system provides a data dependent explanation of the user question. We explain this below with the running example.

Question-Phase-2: Why avg-high-income of group **"Married-AF-spouse"** > that of group **"Married-civ-spouse"**?

Answer-Phase-2: The 95% confidence interval of group difference is $(-0.259, 0.460)$, hence the noise in the query is possibly the reason.

Figure 2: A user question explained by high noise.

First consider the question in Figure 2 comparing the last two groups in Figure 1b (spouse in armed forces vs. a civilian). In this example, even though the noisy avg-high-income for "Married-AF-spouse" is larger than the noisy value for "Married-civ-spouse", this might not be true in the real data (as is the case in the True-answer column). Hence, our system tests whether the user question could potentially be explained just using the noise introduced by DP rather than from the data itself. To do this, our system tests the validity of the user question by computing a confidence interval around the difference between these two outputs. In this case, the confidence interval is $(-0.259, 0.460)$. Since it includes 0 and negative values, we cannot conclude with high probability that "Married-AF-support" > "Married-civ-spouse" is true in the original data. **Since the validity of the user question is uncertain, we know that any further explanation might not be meaningful and the user may choose to stop here.** In other words, the explanation for the comparison in the user question is primarily attributed to the added noise by the DP mechanism.

Now consider the comparison between two other groups "Never-married" and "Married-civ-spouse", in Figure 1c. In this case, the confidence interval about the difference does not include zero and is tight around a positive number 0.4, which indicates that the user question is correct with high probability. Since the question is valid, the user may continue to the next phase.

In the **third phase (Phase-3)** of DPXPlain, 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**"[1] 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 [87, 103] for explaining aggregate queries in non-DP setting. Intuitively, if we intervene on the database by (hypothetically) removing tuples that satisfy the predicate, and re-evaluate 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 ($>, \geq$ etc.). In Figure 1d, the top-5 simple explanation predicates, as computed by DPXPlain, are shown out of 103 singleton predicates, according to their influences to 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 both their *relative influence* (intuitively, how much they affect the difference of the group aggregates

[1] We note that our notion of explanation table is unrelated to that described by Gebaly et al. [47] for summarizing dimension attributes to explain a binary outcome attribute.

in the question) and their *ranks* in the form of confidence interval (upper and lower bounds) to preserve DP.

From this table, `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. In other words, managers tend to earn more if they are married than those who are single, which probably can 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, (see **??** for another random example) 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).

Our Contributions

- We develop DPXPlain, the first framework, to our knowledge, that generates explanations for query answers under DP adapting the notion of intervention [87, 103]. It explains user questions comparing two group-by aggregate query answers (COUNT, SUM, or AVG) with DP in three phases: private query answering, private user question validation, and private explanation table.
- We develop multiple novel techniques that allow DPXPlain to provide explanations under DP by (a) computing confidence intervals to check the validity of user questions, (b) choosing explanation predicates, and (c) computing confidence intervals around the influence and rank of the predicates. In particular, the technical contributions of our work include: (i) a low sensitivity influence function inspired by previous work on non-private explanations [103], and (ii) a binary search-based algorithm to find the confidence intervals of the ranks of explanations, which overcomes the high sensitivity challenge of the rank function.
- We have implemented a prototype of DPXPlain [1] to evaluate our approach. We include two case studies on a real and a synthetic dataset showing the entire process and the obtained explanations. We have further preformed a comprehensive accuracy and performance evaluation, showing that DPXPlain correctly indicates the validity of the question with 100% accuracy for 8 out of 10 questions, selects at least 80% of the true top-5 explanation predicates correctly for 8 out of 10 questions, and generates descriptions about their influences and ranks with high accuracy.

2 PRELIMINARIES

We now give the necessary background for our model. The DPXPlain framework supports single-block SELECT - FROM - WHERE - GROUP BY queries with aggregates (Figure 3) on single tables[2]. Hence 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 $\mathsf{dom}(A_i)$, 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 $a_i \in \mathsf{dom}(A_i)$ for all i. The domain of a tuple is denoted as $\mathsf{dom}(\mathbb{A}) = \mathsf{dom}(A_1) \times \mathsf{dom}(A_2) \times \ldots \times \mathsf{dom}(A_m)$. We denote $A_i^{max} = \max\{|a| \mid a \in \mathsf{dom}(A_i)\}$ as the maximum absolute value of A_i. The value of the attribute A_i of tuple t is denoted by $t.A_i$.

$q =$ `SELECT` A_{gb}`,` `agg(`A_{agg}`)` `FROM D WHERE` ϕ `GROUP BY` A_{gb}`;`

Figure 3: Group-by query with aggregates supported by DPXPlain**. The true results are denoted by (α_i, o_i) and the noisy results released by a DP mechanism are denoted by $(\alpha_i, \hat{o}_i)$ where α_i is the value of A_{gb} and $o_i, \hat{o}_i$ are aggregate values.**

In this paper, we consider group-by aggregate queries q of the form shown in Figure 3. Here A_{gb} is the group-by attribute and A_{agg} is the aggregate attribute, ϕ is a predicate without subqueries, and $\mathtt{agg} \in \{COUNT, SUM, AVG\}$ is the aggregate function. When query q is evaluated on database D, its result is a set of tuples (α_i, o_i), where $\alpha_i \in \mathsf{dom}(A_{gb})$ and $o_i = agg(\{t.A_{agg} \mid t \in D, \phi(t) = true, t.A_{gb} = \alpha_i\})$. For brevity, we will use $\phi'(D)$ to denote $\{t \mid \phi'(t) = true\}$ for any predicate ϕ', and $agg(A_{agg}, D')$, or simply $agg(D')$ when it is clear from context, to denote $agg(\{t.A_{agg} \mid t \in D'\})$ for any $D' \subseteq D$. Hence, $o_i = agg(A_{agg}, g_i(D))$, where $g_i = \phi \wedge (A_{gb} = \alpha_i)$.

Example 2.1. Consider Example 1.1. The schema is $\mathbb{A}$ = (`marital-status`, `occupation`, `age`, `relationship`, `race`, `workclass`, `sex`, `native-country`, `education`, `high-income`). All the attributes are categorical attributes and the domain of `high-income` is $\{0, 1\}$. The query is shown in Figure 1b and the true result for each group is shown in the `True-answer` column. Here $A_{gb} = \mathtt{marital-status}$, $A_{agg} = \mathtt{high-income}$, and $\mathtt{agg} = AVG$.

Differential Privacy. In this work, we consider query-answering and providing explanations using *differential privacy (DP)* [43] to protect private information in the data. 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 too much (defined in the sequel) even if it is evaluated on any two different but *neighboring* databases defined below.

Definition 2.2 (Neighboring Database). Two databases D and D' are neighboring (denoted by $D \approx D'$) if D' can be transformed from D by adding or removing [3] a tuple in D.

To achieve DP, it is necessary to randomize the query result such that given any two neighboring databases, it is highly possible that

[2] Unlike some standard explanation framework [103], in DP, we cannot consider materialization of join-result for multiple tables, since the privacy guarantee depends on *sensitivity*, and removing one tuple from a table may change the join and query result significantly. We leave it as an interesting future work.

[3] There are two variants of neighboring databases. The definition by addition/deletion of tuples is called "unbounded DP", and by updating tuples is called "bounded DP", since the size of data is fixed. In this work, we assume the unbounded version, while DPXPlain can be adapted also for the bounded version by adapting the noise scale.

the answers are the same. Informally, the more similar the two random distributions are, the harder it is to distinguish which database is the actual database, therefore the privacy is better protected.

In this paper, we consider a relaxation of DP called **ρ-zero-concentrated differential privacy (zCDP)** [17, 44] for several reasons. First, we use Gaussian noise to perturb query answers and derive confidence intervals, which does not satisfy pure ϵ-DP [43] but satisfies approximate (ϵ, δ)-DP [43] and ρ-zCDP. Second, ρ-zCDP only has one parameter ρ, comparing to (ϵ, δ)-DP which has two parameters, so it is easier to understand and control. Third, ρ-zCDP allows for tighter analyses for tracking the privacy loss over multiple private releases, which is the case for this framework. The parameter ρ is also called the *privacy budget* of the mechanism. A lower ρ value implies a lower privacy loss.

Definition 2.3 (Zero-Concentrated Differential Privacy (zCDP) [17]). A mechanism $\mathcal{M}$ is said to be *ρ-zero-concentrated differential private*, or ρ-zCDP for short, if for any neighboring datasets D and D' and all $\alpha \in (1, \infty)$ it holds that

$$D_\alpha(\mathcal{M}(D) \| \mathcal{M}(D')) \leq \rho\alpha$$

where $D_\alpha(\mathcal{M}(D) \| \mathcal{M}(D'))$ denotes the Rényi divergence of the distribution $\mathcal{M}(D)$ from the distribution $\mathcal{M}(D')$ at order α [81].

Unless otherwise stated, from now on, we will refer to zero-concentrated differential privacy simply as DP.

A popular approach for providing zCDP to a query result is to add Gaussian noise to the result before releasing it to user. This approach is called *Gaussian mechanism* [17, 43].

Definition 2.4 (Gaussian Mechanism). Given a query q and a noise scale σ, Gaussian mechanism $\mathcal{M}^G$ is given as:

$$\mathcal{M}^G(D; q, \sigma) = q(D) + N(0, \sigma^2)$$

where $N(0, \sigma^2)$ is a random variable from a normal distribution[4] with mean zero and variance σ^2.

Example 2.5. Suppose there is a database D with 100 tuples. Consider a query q = "`SELECT COUNT(*) FROM D`", which counts the total number of tuples in a database D. Here $q(D) = 100$. Now we use Gaussian mechanism to release $q(D)$, which is to randomly sample a noise z from distribution $N(0, \sigma^2)$. Here we assume $\sigma = 1$. Finally, we got a noisy result $\hat{q}(D) = 102.32$, which we may round to an integer in postprocessing without sacrificing the privacy guarantee (Proposition 2.9 below).

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.

Definition 2.6 (Sensitivity). Given a scalar query q that outputs a single number, its sensitivity is defined as:

$$\Delta_q = \sup_{D \approx D'} |q(D) - q(D')|$$

[4]The probability density function of a normal distribution $N(\mu, \sigma^2)$ is given as $exp(-((x-\mu)/\sigma)^2/2)/(\sigma\sqrt{2\pi})$.

Example 2.7. Continuing Example 2.5, since the query q returns the database size, for any two neighboring databases, their sizes always differ by 1, so the sensitivity of q is 1.

The next theorem provides the bound on DP guaranteed by a Gaussian mechanism.

Theorem 2.8 (Gaussian Mechanism [17]). *Given a query q with sensitivity Δ_q and a noise scale σ, its Gaussian mechanism $\mathcal{M}^G$ satisfies $(\Delta_q^2/2\sigma^2)$-zCDP. Equivalently, given a privacy budget ρ, choosing $\sigma = \Delta_q/\sqrt{2\rho}$ in Gaussian mechanism satisfies ρ-zCDP.*

Composition Rules. In our analysis, we will use the following standard composition rules and other known results from the literature of DP [77] (in particular, zCDP [17]) frequently:

Proposition 2.9. *The following holds for zCDP [17, 77]:*

- **Parallel composition:** *if two mechanisms take disjoint data as input, the total privacy loss is the maximum privacy loss from each.*
- **Sequential composition:** *if we run two mechanisms in a sequence on overlapping inputs, the total privacy loss is the sum of each privacy loss.*
- **Postprocessing:** *if we run a mechanism and postprocess the result without accessing the data, the total privacy loss is only the privacy loss from the mechanism.*

Private Query Answering. Recall that we have group-by aggregation query of the form q = `SELECT` A_{gb}`, agg(`A_{agg}`) FROM D WHERE` ϕ `GROUP BY` A_{gb}, and it returns a list of tuples (α_i, o_i) where $\alpha_i \in \mathtt{dom}(A_{gb})$ and o_i is the corresponding aggregate value. Since no single tuple can exist in more than one group, adding or removing a single tuple can at most change the result of a single group. As mentioned earlier, Phase-1 returns noisy aggregate values $\hat{o}_i$ for each α_i instead of o_i. The following holds:

Observation 2.1. *According to the* parallel composition rule *(Proposition 2.9), if for each α_i, its (noisy) aggregate value $\hat{o}_i$ is released under ρ_q-zCDP, the entire release of results including all groups $\{\alpha_i, \hat{o}_i : \alpha_i \in \mathtt{dom}(A_{gb})\}$ satisfies ρ_q-zCDP.*

For a *COUNT* or *SUM* query, we use the Gaussian mechanism for each group α_i: $\hat{o}_i = o_i + N(0, \sigma^2)$, where the noise scale $\sigma = \Delta_q/\sqrt{2\rho_q}$ to satisfy ρ_q-zCDP by Theorem 2.8. The sensitivity term Δ_q is 1 for *COUNT* and A_{agg}^{max} for *SUM*, the maximum absolute value of the aggregation attribute in its domain. For an *AVG* query, since $AVG = SUM/COUNT$, we decompose the query into a *SUM* query and a *COUNT* query, privately answer each of them by half of the privacy budget $\rho_q/2$, get $\hat{o}_i^S$ and $\hat{o}_i^C$ for each group α_i for the *SUM* query and *COUNT* query separately [5], and release $\hat{o}_i = \hat{o}_i^S/\hat{o}_i^C$ as a post-processing step. The noisy query answers of the group-by query with AVG satisfies ρ_q-zCDP by the sequential composition rule (Proposition 2.9), since each of *SUM* and *COUNT* queries satisfies $\rho_q/2$-zCDP.

Confidence Level and Interval. Confidence intervals are commonly used to determine the error margin in uncertain computations and are used in various fields from estimating the error in predictions by machine learning models [59] to providing query

[5]The intermediate releases $\hat{o}_i^S$ and $\hat{o}_i^C$ are stored in the framework for computing the confidence interval of question, which will be discussed in Section 4.1.

results with added noise due to DP [50]. In our context, we use confidence intervals to measure the uncertainty in the user question and our explanations.

Definition 2.10 (Confidence Level and Interval [101]). Given a confidence level γ and an unknown but fixed parameter θ, a random interval $\mathcal{I} = (\mathcal{I}^L, \mathcal{I}^U)$ is said to be its confidence interval, or CI, with confidence level γ if the following holds:

$$Pr[\mathcal{I}^L \leq \theta \leq \mathcal{I}^U] \geq \gamma$$

Notice that θ is a fixed quantity and $\mathcal{I}^L, \mathcal{I}^U$ are random variables. One interpretation of a confidence interval is that with probability at least γ, a random draw of the pair $(\mathcal{I}^L, \mathcal{I}^U)$ as an interval will contain the unknown parameter θ. Two bounds are sampled together unless they are independent.

Example 2.11. Let $\theta = 0$. Suppose with probability 50% we have $I^L = -1$ and $I^U = 1$, and with another probability 50% we have $I^L = 1$ and $I^U = 2$. Therefore, $Pr[\mathcal{I}^L \leq \theta \leq \mathcal{I}^U] = 50\%$, and we can conclude that the random interval $\mathcal{I} = (\mathcal{I}^L, \mathcal{I}^U)$ is a 50% level confidence interval for θ.

3 PRIVATE EXPLANATIONS IN DPXPLAIN

In this section we provide the model for private explanations of query results in DPXPlain, outline the key technical problems addressed by DPXPlain, and highlight the difference from existing database explanation frameworks.

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[6]:

Definition 3.1 (User Question). Given a database D, a group-by aggregate query q as shown in Figure 3, a DP mechanism $\mathcal{M}$, and two noisy answer tuples $(\alpha_i, \hat{o}_i), (\alpha_j, \hat{o}_j) \in \mathcal{M}(D; q)$ where $\hat{o}_i > \hat{o}_j$, a *user question* has the form *"why is the (noisy) aggregate value $\hat{o}_i$ of group α_i larger than the aggregate value $\hat{o}_j$ of group α_j?"*), which is denoted by "why $(\alpha_i, \alpha_j, >)$?".

Example 3.2. The question from Figure 1c is denoted as *"why (*`'Married-civ-spouse', 'Never-married', >`)?"*.

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 [47, 70, 87, 103]. We follow this paradigm and define explanation predicates.

Definition 3.3 (Explanation Predicate). Given a database D with a set of attributes $\mathbb{A}$, a group-by aggregation query q (Figure 3) with group-by attribute A_{gb} and aggregate attribute A_{agg} and a predicate size l, an explanation predicate p is a Boolean expression of the form $p = \varphi_1 \wedge ... \wedge \varphi_l$, where each φ_i has the form $A_i = a_i$ such that $A_i \in \mathbb{A} \setminus \{A_{gb}, A_{agg}\}$ is an attribute, and $a_i \in \mathtt{dom}(A_i)$ is its value. [7]

[6]Our framework can handle more general user questions involving single group or more than two groups; details are deferred to Appendix C.1

[7]We assume $\mathtt{dom}(A_i)$ is discrete, finite and data-independent. Our framework can also handle inequality $A_i \circ a_i$ where $\circ \in \{>, <, \geq, \leq, \neq\}$ when the constant a_i is from a finite and data-independent set. Conjunctions can also be replaced with disjunctions.

New challenges for explanations with DP. Unlike standard explanation framework on aggregate queries [70, 87, 103], the existing frameworks are not sufficient to support DP and need to be adapted: (i) the question itself might not be valid due to the noise injected into the queries, (ii) the selection of top-k explanation predicates needs to satisfy DP, which further requires the influence function to have low sensitivity so that the selection is less perturbed, and (iii) since the selected explanation predicates are not guaranteed to be the true top-k, it is also necessary to output extra descriptions under DP for each selected explanation predicate about their actual influences and ranks. We detail the adjustments as follows.

Question Validation with DP (Phase-2). While the user is asking "why is $\hat{o}_i > \hat{o}_j$?", in reality, it may be the case that the true results satisfy $o_i \leq o_j$, i.e., they have opposite relationship than the one observed by the user. This indicates that $\hat{o}_i > \hat{o}_j$ is the result of the noise being added to the results. In this scenario, one option to explain the user's observation of $\hat{o}_i > \hat{o}_j$ will be releasing the true values (equivalently, the added exact noise values), which will violate DP. Instead, to provide an explanation in such scenarios, we generate a confidence interval for the difference of two (hidden) aggregate values $o_i - o_j$, which can include negative values (discussed in detail in Section 4.1). This leads to the first problem we need to solve in the DPXPlain framework:

Problem 1 (Private Confidence Interval of Question)**.** *Given a dataset D, a query q, a DP mechanism $\mathcal{M}$, a privacy budget ρ_q, a confidence level γ, and a user question $(\alpha_i, \alpha_j, >)$ on the noisy query answers output by $\mathcal{M}$ satisfying ρ_q-zCDP, find a confidence interval (see Definition 2.10) for the user question $\mathcal{I}_{uq} = (\mathcal{I}^L_{uq}, \mathcal{I}^U_{uq})$ for $o_i - o_j$ at confidence level γ without extra privacy cost.*

In Phase-2, the framework returns a confidence interval of $o_i - o_j$ to the user. If it includes zero or negative numbers, it is possible that $o_i \leq o_j$, and the user's observation of $\hat{o}_i > \hat{o}_j$ is the result of the noise added by the DP mechanism. In such cases, the user may stop at Phase-2. If the user is satisfied with the confidence interval for the validity of the question, she can proceed to Phase-3.

Influence Function (Phase-3). When considering DP, the order of the explanation predicates is perturbed by the noise we add to the influences according to the sensitivity of the influence function (discussed in detail in Section 4.3.1). To provide useful explanations, this sensitivity needs to be low, which means the influence does not change too much by adding or removing a tuple from the database. For example, a counting query that outputs the database size n has sensitivity 1, since its result can only change by 1 for any neighboring databases. Following this concept, we propose the second and a core problem for the DPXPlain framework, which is also critical to the subsequent problems defined below.

Problem 2 (Influence Function with Low Sensitivity)**.** *Find an influence function $\textsc{Inf} : \mathcal{P} \rightarrow \mathcal{R}$ that maps an explanation predicate to a real number and has low sensitivity.*

Private Top-k Explanations (Phase-3). In DPXPlain, to satisfy DP, in Phase-3 we output the top-k explanation predicates ordered by the noisy influences, and release the influences and ranks of these predicates in the form of confidence intervals to describe the uncertainty. To achieve this goal, we tackle the following three sub-problems.

Problem 3 (Private Top-k Explanation Predicates)**.** *Given a set of explanation predicates $\mathcal{P}$, an integer k, and a privacy parameter ρ_{Topk}, find the top-k highest influencing predicates $p_1, p_2, \ldots, p_k$ from $\mathcal{P}$ while satisfying ρ_{Topk}-zCDP.*

Problem 4 (Private Confidence Interval of Influence)**.** *Given a confidence level γ, k explanation predicates $p_1, p_2, \ldots, p_k$, and a privacy parameter ρ_{Influ}, find a confidence interval $\mathcal{I}_{influ} = (\mathcal{I}^L_{influ}, \mathcal{I}^U_{influ})$ for influence $\textsc{Inf}(p_u)$ at confidence level γ for each $u \in \{1, \ldots, k\}$ satisfying ρ_{Influ}-zCDP (overall privacy budget).*

Problem 5 (Private Confidence Interval of Rank)**.** *Given a confidence level γ, k explanation predicates $p_1, p_2, \ldots, p_k$, and a privacy parameter ρ_{Rank}, find a confidence interval $\mathcal{I}_{rank} = (\mathcal{I}^L_{rank}, \mathcal{I}^U_{rank})$ for rank of p_u at confidence level γ for each $u \in \{1, \ldots, k\}$ satisfying ρ_{Rank}-zCDP (overall privacy budget).*

4 COMPUTING EXPLANATIONS UNDER DP

Next we provide solutions to problems 1, 2, 3, 4, and 5 in Sections 4.1, 4.2, 4.3.1, 4.3.2, and 4.3.3 respectively, and analyze their properties. We summarize the entire DPXPlain framework in Section 4.4.

4.1 Confidence Interval for a User Question

For **Problem 1**, the goal is to find a confidence interval of $o_i - o_j$ for the user question at the confidence level γ without extra privacy cost in Phase-2. We divide the solution into two cases. (1) When the aggregation is COUNT or SUM, the noisy difference $\hat{o}_i - \hat{o}_j$ follows Gaussian distribution, which leads to a natural confidence interval. (2) When the aggregation is AVG, the noisy difference does not follow Gaussian distribution, but we show that the confidence interval in this case can be derived through multiple partial confidence intervals. The solutions below only take the noisy query result as input, which does not incur extra privacy loss according to the post-processing property of DP (Proposition 2.9). The pseudo codes can be found in . Appendix B.1

Confidence interval for COUNT and SUM. For a COUNT or SUM query, recall from Section 2 that $\hat{o}_i$ and $\hat{o}_j$ are produced by adding Gaussian noises to o_i and o_j with some noise scale σ. Therefore, the difference between $\hat{o}_i$ and $\hat{o}_j$ also follows Gaussian distribution with mean $o_i - o_j$ and scale $\sqrt{2}\sigma$ (since the variance is $2\sigma^2$). Following the standard properties of Gaussian distribution, the interval with center c as $\hat{o}_i - \hat{o}_j$ and margin m as $\sqrt{2}(\sqrt{2}\sigma)\,\mathrm{erf}^{-1}(\gamma)$ [8], or (c-m, c+m), is a γ level confidence interval of $o_i - o_j$ [101].

Confidence interval for AVG. For an AVG query, even the single noisy answer $\hat{o}_i$ does not follow Gaussian distribution, because it is a division between two Gaussian variables as described in Section 2: $\hat{o}_i = \hat{o}^S_i / \hat{o}^C_i$. Hence, first we derive partial confidence intervals for o^S_i and o^C_i as discussed above, denoted by $\mathcal{I}^S$ and $\mathcal{I}^C$, individually at some confidence level β. Let $I^A = \mathcal{I}^S / \mathcal{I}^C := \{x/y \mid x \in \mathcal{I}^S, y \in \mathcal{I}^C\}$ [9] to be the set that includes all possible divisions between any numbers from $\mathcal{I}^S$ and $\mathcal{I}^C$. Especially, if I^C contains zero, we return a trivial confidence interval $(\infty, -\infty)$ that is always valid. Otherwise, I^A is a $2\beta - 1$ level confidence interval for the division, as stated in the following proposition.

Lemma 4.1. *Given $\mathcal{I}^S$ and $\mathcal{I}^C$ as two β level confidence intervals of o^S_i and o^C_i separately, the derived interval $\mathcal{I}^A = \{x/y \mid x \in \mathcal{I}^S, y \in \mathcal{I}^C\}$ is a $2\beta - 1$ level confidence interval of o^S_i / o^C_i.*

Proof. The following holds:
$Pr[o^S_i/o^C_i \in \mathcal{I}^A] \geq Pr[o^S_i \in \mathcal{I}^S \wedge o^C_i \in \mathcal{I}^C] \geq 1 - (Pr[o^S_i \notin \mathcal{I}^S] + Pr[o^C_i \notin \mathcal{I}^C]) \geq 1 - ((1-\beta) + (1-\beta)) = 2\beta - 1$ The first inequality above is due to fact that the second event is sufficient for the first event. The next inequality holds by applying the union bound. □

Furthermore, the difference $\hat{o}_i - \hat{o}_j$ is a subtraction between two ratios of two Gaussian variables, which can be expressed as an arithmetic combination of multiple Gaussian variables: $\hat{o}_i - \hat{o}_j = X_i/Y_i - X_j/Y_j$, where $X_t = N(o^S_t, \sigma^2_S)$ and $Y_t = N(o^C_t, \sigma^2_C)$ for $t \in \{i, j\}$. Similar to Lemma 4.1, we can derive the confidence interval for $\hat{o}_i - \hat{o}_j$ based on 4 partial confidence intervals of o^S_i, o^C_i, o^S_j, and o^C_j instead of 2. The confidence level we set for each partial confidence interval is $\beta = 1 - (1-\gamma)/4$ by applying union bound on the failure probability $1 - \gamma$ that one of the four variables is outside its interval. After we have 4 partial confidence intervals $\mathcal{I}^S_i$, $\mathcal{I}^C_i$, $\mathcal{I}^S_j$, and $\mathcal{I}^C_j$ for o^S_i, o^C_i, o^S_j, and o^C_j separately, similar to Lemma 4.1, we combine them together as $\mathcal{I}^A = \mathcal{I}^S_i / \mathcal{I}^C_i - \mathcal{I}^S_j / \mathcal{I}^C_j$ and derive the confidence interval for $o_i - o_j$ as $(\inf \mathcal{I}^A, \sup \mathcal{I}^A)$, which is guaranteed to be at confidence level γ. If 0 is included in either $\mathcal{I}^C_i$ or $\mathcal{I}^C_j$, we set the confidence interval to be $(\infty, -\infty)$ instead.

4.2 Influence Function with Low Sensitivity

For **Problem 2**, the goal is to design an influence function that has low sensitivity. Inspired by PrivBayes [105], we start by adapting a known influence function to our framework, and then adapt it to have a low sensitivity.

Our influence function of an explanation predicate with respect to a comparison user question is inspired by the Scorpion framework [103], where the user questions seek explanations for outliers in the results of a group-by aggregate query. The Scorpion framework identifies predicates on input data that cause the outliers to disappear from the output results. Given the group-by aggregation query shown in Figure 3 and a group $\alpha_i \in \mathtt{dom}(A_{gb})$, recall from Section 2 that the true aggregate value for α_i is $o_i = agg(A_{agg}, g_i(D))$, where $g_i = \phi \wedge (A_{gb} = \alpha_i)$, i.e., $g_i(D)$ denotes the set of tuples that contribute to the group α_i.

Scorpion measures the influence of an explanation predicate p to some group α_i as the ratio between the change of output aggregate value and the change of group size:

$$\frac{agg(g_i(D)) - agg(g_i(\neg p(D)))}{|g_i(p(D))|} \tag{1}$$

Here $\neg p(D)$ denotes $D - p(D)$, i.e., the set of tuples in D that do not satisfy the predicate p. To adapt this influence function to DPXPlain, we make the following two changes.

- First, it should measure the influence w.r.t. the comparison from the user question $(\alpha_i, \alpha_j, >)$ instead of a single group. A natural extension is to change the target aggregate on g_i in the numerator in (1) to the difference between the aggregate values of two groups g_i, g_j before and after applying the explanation predicate p, and

[8] erf^{-1} is the inverse function of the error function $\mathrm{erf}\, z = (2/\sqrt{\pi}) \int_0^z e^{-t^2} dt$.

[9] In the algorithm, we only need the maximum and the minimum of the set to construct the interval, which can be solved by a numerical optimizer.

change the denominator as the maximum change in g_i or g_j when p is applied, which gives the following influence function:

$$\frac{\big(agg(g_i(D)) - agg(g_j(D))\big) - \big(agg(g_i(\neg p(D))) - agg(g_j(\neg p(D)))\big)}{\max(|g_i(p(D))|, |g_j(p(D))|)} \quad (2)$$

- Second and more importantly, in DPXPlain, we need to preserve DP when we use influence function to sort and rank multiple explanation predicates, or to release the influence and rank of an explanation predicate. Therefore, **we need to account for the sensitivity of the influence function**, which is determined by the worst-case change of influence when a tuple is added or removed from the database. If the predicate only selects a small number of tuples, the denominator in (2) is small and thus changing the denominator in (2) by one (when a tuple is added or removed) can result in a big change in the influence as illustrated in the following example, making (2) unsuitable for DPXPlain.

Example 4.2 (The Issue of the Influence Sensitivity). Suppose there are two groups α_i and α_j in D with 1000 tuples in each, aggregate function $agg = SUM$ on attribute A_{agg} with domain $[0, 100]$, and the explanation predicate p matches only 1 tuple from the group α_i with $A_{agg} = 100$ and no tuple from α_j. Suppose $agg(g_i(D)) = 20,000$, $agg(g_j(D)) = 10,000$, then $agg(g_i(\neg p(D))) = 19,900$ and $agg(g_j(\neg p(D))) = 10,000$. Therefore, from Equation (2), the influence of p is $((20,000-10,000)-(19,900-10,000))/\max\{1,0\} = 100$ on the original database D. However, suppose a new tuple that satisfies p and belongs to group α_i is added with $A_{agg} = 2$. Now the influence in Equation (2) becomes $((20,002-10,000)-(19,900-10,000))/\max\{2,0\} = 102/2 = 51$. Note that while we added a tuple contributing only 2 to the sum, it led to a change of 100-51 = 49 to the influence function because of the small denominator.

Therefore, we propose a new influence function that is inspired by Equation (2) but has lower sensitivity. Note that the denominator in Scorpion's influence function in Equation (2) acts as a normalizing factor, whose purpose is to penalize the explanation predicate that selects too many tuples, e.g., to prohibit removal of the entire database by a dummy predicate. To have a similar normalizing factor with low sensitivity, we multiply the numerator in Equation (2) by $\frac{\min(|g_i(\neg p(D))|, |g_j(\neg p(D))|)}{\max(|g_i(D)|, |g_j(D)|)+1}$. From this new normalizing factor, the numerator captures the minimum of the number of tuples that are not removed from each group, and the denominator is a constant, which does not change for different explanation predicates and keeps the normalizing factor in the interval $[0, 1]$. Similar to Scorpion, if $p(D)$ constitutes a large fraction of D (e.g., if $p(D) = D$), then the normalizing factor is small, reducing the value of the influence. Also note that, unlike standard SQL query answering where only non-empty groups are shown in the results, in DP, all groups from the actual domain have to be considered, hence unlike Equation (1), $g_i(D), g_j(D)$ could be zero, hence 1 is added in the denominator to avoid division by zero. When $agg = AVG$, we remove the constant denominator to boost the signal of the influence and keep the sensitivity low, which will be discussed in the sensitivity analysis after Proposition 4.4 and in Example 4.5. We formally define the influence as follows.

Definition 4.3 (Influence of Explanation Predicates). Given a database D, a query q as shown in Figure 3, and a user question $(\alpha_i, \alpha_j, >)$, the influence of an explanation predicate p is defined as $\textsc{Inf}(p; (\alpha_i, \alpha_j, >), D)$, or simply $\textsc{Inf}(p)$ when it is clear from context:

$$\textsc{Inf}(p) = \big(\big(agg(g_i(D)) - agg(g_j(D))\big) - \big(agg(g_i(\neg p(D))) - agg(g_j(\neg p(D)))\big)\big) \times \begin{cases} \frac{\min(|g_i(\neg p(D))|, |g_j(\neg p(D))|)}{\max(|g_i(D)|, |g_j(D)|)+1} & \text{for } agg \in \{COUNT, SUM\} \\ \min(|g_i(\neg p(D))|, |g_j(\neg p(D)|) & \text{for } agg = AVG \end{cases} \quad (3)$$

The proposition below guarantees the sensitivity of the influence function. We give an intuitive proof as follows, where the formal proofs can be found at Appendix A.1. When agg = COUNT, we combine two group differences $\big(agg(g_i(D)) - agg(g_j(D))\big) - \big(agg(g_i(\neg p(D))) - agg(g_j(\neg p(D)))\big)$ into a single group difference as $agg(g_i(p(D)) - agg(g_j(p(D))$, which is considered as a subtraction between two counting queries. We prove that the sensitivity of a counting query after a multiplication with the normalizing factor will multiply its original sensitivity by 2. Since we have two counting queries, the final sensitivity is 4. When $agg = SUM$, the proof is similar except we need to multiply the final sensitivity by A_{agg}^{max} , the maximum absolute domain value of A_{agg}. For AVG, we view it as a summation of 4 AVG queries that times with $\min(|g_i(\neg p(D))|, |g_j(\neg p(D)|)$. Intuitively, we change AVG to SUM, and, therefore, reduce to the case of SUM and bound the sensitivity. This sensitivity now becomes relatively small since we have amplified the influence.

Proposition 4.4. *[Influence Function Sensitivity] Given an explanation predicate p and an user question with respect to a group-by query with aggregation agg, the following holds:*

(1) If $agg = COUNT$, the sensitivity of $\textsc{Inf}(p)$ is 4.
(2) If $agg = SUM$, the sensitivity of $\textsc{Inf}(p)$ is 4 A_{agg}^{max}.
(3) If $agg = AVG$, the sensitivity of $\textsc{Inf}(p)$ is 16 A_{agg}^{max}.

Intuitively, the sensitivity of $\textsc{Inf}(p)$ is low compared to its value. When $agg = COUNT$, $\textsc{Inf}(p)$ is $O(n)$ and $\Delta_{\textsc{Inf}}$ is $O(1)$, where n is the size of database. When $agg \in \{SUM, AVG\}$, $\textsc{Inf}(p)$ is $O(nA_{agg}^{max})$ and $\Delta_{\textsc{Inf}}$ is $O(A_{agg}^{max})$. Therefore, the sensitivity of influence $\Delta_{\textsc{Inf}}$ is low comparing to the influence itself. However, as the example below shows, if we define the influence function for AVG the same way as $COUNT$ or SUM, both $\textsc{Inf}(p)$ and $\Delta_{\textsc{Inf}}$ will become $O(A_{agg}^{max})$, which makes the sensitivity (relatively) large.

Example 4.5 (The Issue with AVG Influence.). Consider an AVG group-by query where the domain of the aggregate attribute is $[0, 100]$, and an explanation predicate p such that for group α_i we have 2 tuples with $AVG(g_i(D)) = 100/2 = 50$, $AVG(g_i(\neg p(D))) = 0/1 = 0$, and for group α_j we have two tuples with $AVG(g_j(D)) = 100/2 = 50$ and $AVG(g_j(\neg p(D))) = 100/2 = 50$. Suppose we define the influence function for AVG the same way as $COUNT$ or SUM, therefore the influence of p in Equation (3) is $\textsc{Inf}(p) = ((50-50)-(0-50))(\min(1,2)/(\max(2,2)+1) = 50/3$. However, suppose we remove the single tuple from g_i, so $|g_i(\neg p(D))|$ becomes 0, now the influence in Equation (3) (for COUNT/SUM) becomes 0. Note that a single removal of a tuple completely changes the influence to 0, and this change is equal to the influence itself, which is relatively large and therefore is not a good choice for AVG.

Note that the user question "why $(\alpha_i, \alpha_j, >)$" is asked based on the noisy results $\hat{o}_i > \hat{o}_j$, while the influence function uses the true results, i.e., even if $o_i \leq o_j$, we still consider $agg(g_i(D)) -$

$agg(g_j(D))$) in $\text{Inf}(p)$. Hence $\text{Inf}(p)$ can be positive or negative and removing tuples satisfying p can make the gap smaller or larger. In Appendix B.2. we show that $\text{Inf}(p)$ is not monotone with p-s.

4.3 Private Top-k Explanations

In this section, we discuss the computation of the top-k explanation predicates and the confidence intervals of influences and ranks.

4.3.1 **Problem 3: Private Top-k Explanation Predicates.** The goal is to find with DP the top-k explanation predicates from a set of explanation predicates $\mathcal{P}$ in terms of their (true) influences $\text{Inf}(p)$, which is the first step in Phase-3 of DPXPlain (Figure 1). Note that simply choosing the *true* top-k explanation predicates in terms of their $\text{Inf}(p)$ is not differentially private.

In DPXPlain, we adopt the **One-shot Top-k mechanism** [38, 39] to privately select the top-k. It works as follows. For each explanation predicate $p \in \mathcal{P}$, it adds a Gumbel noise [10] to its influence with scale $\sigma = 2\Delta_{\text{Inf}}\sqrt{k/(8\rho_{Topk})}$, where Δ_{Inf} is the sensitivity of the influence function (discussed in Proposition 4.4), reorders all the explanation predicates in a descending order by their noisy influences, and outputs the first k explanation predicates. It satisfies ρ_{Topk}-zCDP [19, 35, 38, 39, 84], since it is equivalent to iteratively applying k exponential mechanisms [43], where each satisfies $\epsilon^2/8$-zCDP [19, 35, 39, 84] and $\epsilon = \sqrt{8\rho_{Topk}/k}$ [38, 39]. Therefore, in total it satisfies $(k\epsilon^2/8)$-zCDP by the sequential composition property (Proposition 2.9) which is also ρ_{Topk}-zCDP. The returned list of top-k predicates is close to that of the true top-k in terms of their influences; the proof is based on the utility proposition of exponential mechanism in Theorem 3.11 of [43]. We summarize the properties of this approach in the following proposition and defer the pseudo codes and proofs to Appendix B.1 and Appendix A.2.

Proposition 4.6. *Given an influence function Inf with sensitivity Δ_{Inf}, a set of explanation predicates $\mathcal{P}$, a privacy parameter ρ_{Topk} and a size parameter k, the following holds:*

(1) One-shot Top-k mechanism finds k explanation predicates while satisfying ρ_{Topk}-zCDP.

(2) Denote by $OPT^{(i)}$ the i-th highest (true) influence, and by $\mathcal{M}^{(i)}$ the i-th explanation predicate selected by the One-shot Top-k mechanism. For $\forall t$ and $\forall i \in \{1, 2, \ldots, k\}$, we have

$$Pr\left[\text{Inf}(\mathcal{M}^{(i)}) \leq OPT^{(i)} - \frac{2\Delta_{Inf}}{\sqrt{8\rho_{Topk}/k}}(\ln(|\mathcal{P}|) + t)\right] \leq e^{-t} \quad (4)$$

Example 4.7. Reconsider the user question in Figure 1c. For this question, we have in total 103 explanation predicates as the set of explanation predicates. The privacy budget $\rho_{Topk} = 0.05$, the size parameter $k = 5$, and the sensitivity $\Delta_{\text{Inf}} = 16$. For each of the explanation predicate, we add a Gumbel noise with scale $\sigma = 113$ to their influences. For example, for the predicates shown in Figure 1d, their noisy influences are 990, 670, 645, 475, 440, which are the highest 5 among all the noisy influences. The true influences for these five ones are 547, 501, 555, 434, 118. To see how close it is to the true top-5, we compare their true influences with the true highest five influences: 555, 547, 501, 434, 252, which shows the corresponding differences in terms of influence are 8, 46, 54, 0, 134. By Equation (4), in theory the probability that such difference is beyond 864 is at most 5% for each explanation predicate. Finally, we sort explanation predicates by their noisy influences and report the top-k. These k predicates will be reordered as discussed in Section 4.4.

[10] For a Gumbel noise $Z \sim Gumbel(\sigma)$, its CDF is $Pr[Z \leq z] = \exp(-\exp(-z/\sigma))$.

4.3.2 **Problem 4: Private Confidence Interval of Influence.** The goal is to generate a confidence interval of influence $\text{Inf}(p)$ (Definition 4.3) of each explanation predicate $\text{Inf}(p_1)$, $\text{Inf}(p_2)$, ..., $\text{Inf}(p_k)$ from the selected top-k (Section 4.3.1). For each $\text{Inf}(p_i)$, we apply the Gaussian mechanism (Theorem 2.8) with privacy budget ρ_{Influ}/k to release a noisy influence $\widehat{\text{Inf}}_i$ with noise scale $\sigma = \Delta_{\text{Inf}}/\sqrt{2\rho_{Influ}/k}$. The sensitivity term Δ_{Inf} is determined by Proposition 4.4. Following the standard properties of Gaussian distribution, for each $\text{Inf}(p_i)$, we set the confidence interval by a center c as $\widehat{\text{Inf}}_i$ and a margin m as $\sqrt{2}\sigma\,\text{erf}^{-1}(\gamma)$, or (c-m, c+m), as a γ level confidence interval of $\text{Inf}(p_i)$ [101]. Together, it satisfies ρ_{Influ}-zCDP according to the composition property by Proposition 2.9. Pseudo codes can be found in Appendix B.4.

4.3.3 **Problem 5: Private Confidence Interval of Rank.** The goal is to find the confidence interval of the rank of each explanation predicate from the selected top-k (Section 4.3.1). We denote $\text{rank}(p)$ as the rank of $p \in \mathcal{P}$ by the natural ordering of the predicates imposed by their (true) influences according to the influence function Inf, and denote $\text{rank}^{-1}(t)$ (for an integer $1 \leq t \leq |\mathcal{P}|$) as the predicate ranked in the t-th place according to Inf. One trivial example of a confidence interval of rank is $[1, |\mathcal{P}|]$, which has no privacy loss and always includes the true rank.

Unlike the sensitivity of the influence function, the sensitivity of $\text{rank}(p)$ is high, since adding one tuple could possibly changing the highest influence to be the lowest and vice versa. Fortunately, we can employ a critical observation about rank and influence.

Proposition 4.8. *Given a set of explanation predicates $\mathcal{P}$, an influence function Inf with global sensitivity Δ_{Inf}, and an integer $1 \leq t \leq |\mathcal{P}|$, $\text{Inf}(\text{rank}^{-1}(t))$ has sensitivity Δ_{Inf}.*

The intuition behind this proof (details in Appendix A.2) is that, fixing an explanation predicate $p = \text{rank}^{-1}(t)$, for a neighboring database, if its influence is increased, its rank will be moved to the top which pushes down other explanation predicates with lower influences, so the influence at the rank t in the neighboring database is still low. For a target explanation predicate p, since both $\text{Inf}(p)$ and $\text{Inf}(\text{rank}^{-1}(t))$ have low sensitivity as Δ_{Inf}, intuitively we can check whether t is close to the rank of p by checking whether their influences $\text{Inf}(p)$ and $\text{Inf}(\text{rank}^{-1}(t))$ are close by adding a little noise to satisfy DP. Given this observation, we devise a binary-search based strategy to find the confidence interval of rank.

Noisy binary search mechanism. We decompose the problem into finding two bounds of the confidence interval separately by a subroutine $\text{RankBound}(p, \rho, \beta, dir)$ that guarantees that it will find a lower ($dir = -1$) or upper ($dir = +1$) bound of rank with probability β for the explanation predicate p using privacy budget ρ. We return $(\text{RankBound}(p_u, 0.1\rho, \beta, -1), \text{RankBound}(p_u, 0.9\rho, \beta, +1))$ as the confidence interval of rank for each predicate p_u for $u \in \{1, \ldots, k\}$, where $\rho = \rho_{Rank}/k$ to divide the privacy budget equally, and $\beta = (\gamma + 1)/2$ to ensure an overall confidence of γ.

The subroutine RankBound(p, ρ, β, dir) works as follows. It is a noisy binary search with at most $N = \lceil \log_2 |\mathcal{P}| \rceil$ loops. We initialize the search pointers $t_{low} = 1$ and $t_{high} = |\mathcal{P}|$ as the two ends of possible ranks. Within each loop, we check the difference of influences at $t = \lfloor (t_{high} + t_{low})/2 \rfloor$ by adding a Gaussian noise:

$$\hat{s} = \text{Inf}(p) - \text{Inf}(\text{rank}^{-1}(t)) + \mathcal{N}(0, \sigma^2) \tag{5}$$

The noise scale is set as $\sigma = (2\Delta_{\text{Inf}})/\sqrt{2(\rho/N)}$ to satisfy ρ/N-zCDP. Instead of comparing the noisy difference $\hat{s}$ with 0 to check whether t is a close bound of rank(p), we compare it with the following slack constant ξ below so that t is a true bound of rank(p) with high probability.

$$\xi = \sigma\sqrt{2\ln(N/(1-\beta))} \times dir \tag{6}$$

We update the binary search pointers by the comparison as follows: if $\hat{s} \geq \xi$, we set $t_{high} = \max\{t-1, 1\}$, otherwise $t_{low} = \min\{t+1, |\mathcal{P}|\}$. The binary search stops when $t_{high} \leq t_{low}$ and returns t_{high} as the rank bound. Example 4.9 gives an illustration. We defer the pseudo codes of the **noisy binary search mechanism** to Appendix B.5.

Example 4.9. Figure 4 shows an example of RankBound for finding the upper bound of the confidence interval for rank(p) for some explanation predicate p (with true rank 3 shown in red). The upper part of the figure shows the influences of all the explanation predicates in a descending order, and the lower part shows the status of the binary search pointers in each loop. The search contains three loops starting from $t_{low} = 1$ and $t_{high} = 15$. Within each loop, to illustrate the idea, it is equivalent to adding a Gaussian noise to $\text{Inf}(\text{rank}^{-1}(t))$, which is shown as a blue circle, compare it with $\text{Inf}(p) - \xi$, which is shown as a dashed line, and update the pointers accordingly. For example, in loop 1, the blue circle 1 is in the green region, so the pointer t_{high} is moved from 15 to 7 (shown in the lower part). Finally it breaks at $t_{low} = t_{high} = 5$.

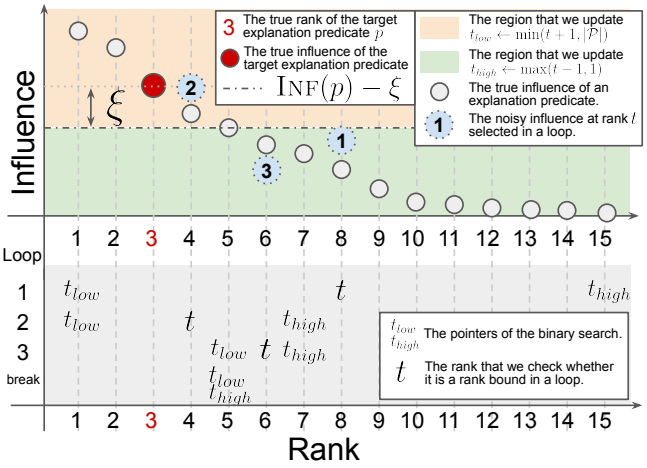


Figure 4: The execution of RankBound for finding the upper bound of the confidence interval of rank for the predicate p (with true rank 3 shown in red) from a toy example.

We now show that **noisy binary search mechanism** satisfies the privacy requirement, and outputs valid confidence intervals. In Section 5, we show that the interval width is empirically small.

Theorem 4.10. *Given a database D, a predicate space $\mathcal{P}$, an influence function Inf with sensitivity Δ_{Inf}, explanation predicates $p_1, p_2, \ldots, p_k$, a confidence level γ, and a privacy parameter ρ_{Rank}, noisy binary search mechanism returns confidence intervals $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_k$ such that*

(1) Noisy binary search mechanism satisfies ρ_{Rank}-zCDP.

(2) For $\forall u \in [1, k]$, $\mathcal{I}_u$ is a γ level confidence interval of rank(p_u).

The proof of item 1 follows from the composition theorem and the property of Gaussian mechanism [17]. The proof of item 2 is based on the property of the random binary search. We defer the formal proofs and a week utility bound to Appendix A.2.

4.4 Putting it All Together

After we selected top-k explanation predicates (Section 4.3.1), and constructed the confidence intervals of their influences (Section 4.3.2) and ranks (Section 4.3.3), the final step is to combine them together into a single explanation table.

Relative Influence. Recall that the influence defined from Definition 4.3 is the difference of $(o_i - o_j)$ before and after removing the tuples related to an explanation predicate (first term), and multiplies with a normalizer to penalize trivial predicates (second term). Since the absolute value of influence is hard to interpret, to help user better understand the confidence interval of influence, we show the *relative influence* compared to the original difference $|o_i - o_j|$ as a percentage. However, we cannot divide the influence by $|o_i - o_j|$ since using the actual data values will incur additional privacy loss, hence, for SUM and COUNT we divide the true influence by $|\hat{o}_i - \hat{o}_j|$ as an approximation since the normalizer in the second term is bounded in $[0, 1]$. However, when $agg = AVG$, the normalizer $\min(|g_i(\neg p(D))|, |g_j(\neg p(D)|)$ (second term) is not bounded in $[0, 1]$, so we further divide the influence by another constant, the minimum of the noisy counts/sizes of the groups, i.e., $|\min(\hat{o}_i^C, \hat{o}_j^C)|$ (approximating the upper bound $\min(|g_i(D)|, |g_j(D)|)$ of the normalizer to avoid additional privacy loss). As a summary, we define the relative influence $\widetilde{\text{Inf}}(p; (\alpha_i, \alpha_j, >), D)$, or simply $\widetilde{\text{Inf}}(p)$, as follows, which is only used for display purposes.

$$\widetilde{\text{Inf}}(p) = \text{Inf}(p) / \begin{cases} |\hat{o}_i - \hat{o}_j| & \text{for } agg \in \{COUNT, SUM\} \\ |\hat{o}_i - \hat{o}_j| \times |\min(\hat{o}_i^C, \hat{o}_j^C)| & \text{for } agg = AVG \end{cases} \tag{7}$$

Explanation Table. We define the explanation table as follows.

Definition 4.11 (Explanation Table containing top-k explanations). Given a database D, a group-by aggregate query q as shown in Figure 3, a user question $(\alpha_i, \alpha_j, >)$, a predicate space $\mathcal{P}$, a confidence level γ, and an integer k, a table of top-k explanations is a list of k 5-element tuples $(p_u, \mathcal{I}^L_{relinflu_u}, \mathcal{I}^U_{relinflu_u}, \mathcal{I}^L_{rank_u}, \mathcal{I}^U_{rank_u})$ for $u = 1, 2, \ldots, k$ such that p_u is an explanation predicate, $(\mathcal{I}^L_{relinflu_u}, \mathcal{I}^U_{relinflu_u})$ is a confidence interval of relative influence $\widetilde{\text{Inf}}(p_u)$ with confidence level γ, and $(\mathcal{I}^L_{rank_u}, \mathcal{I}^U_{rank_u})$ is a confidence interval of rank(p_u) with confidence level γ

Sorting the explanations in the explanation table. Since this table contains the bounds of the influences and ranks (see the last four columns in Figure 1d for an example), it is natural to present the table as a sorted list. Since the numbers in the table are generated by random processes, each column may imply a different sorting. In this paper, we sort the selected top-k explanations by the upper bound of the relative influence CI (the third column in Figure 1d)

in a descending order; if there is a tie, we break it using the upper bound of the rank confidence interval (the fifth column in Figure 1d). Finding a principled way for sorting the explanation predicates is an intriguing subject of future work.

Overall DP guarantee. We summarize the privacy guarantee of our DPXPlain framework as follows: (i) the private noisy query answers returned by Gaussian mechanism in Phase-1 satisfy ρ_q-zCDP together (see Section 2) ; (ii) Phase-2 only returns the confidence intervals of the noisy answers in Phase-1 as defined by the Gaussian mechanism and does not have any additional privacy loss (discussed in Section 4.1) ; (iii) Phase-3 returns k explanation predicates and their upper and lower bounds on relative influence and ranks given a required confidence interval, and uses three privacy parameters $\rho_{Topk}, \rho_{Influ}, \rho_{Rank}$ (discussed in Section 4.3.1, 4.3.2 and 4.3.3). The following theorem summarizes the total privacy guarantee.

Theorem 4.12. *Given a group-by query q and a user question comparing two aggregate values in the answers of q, the* DPXPlain *framework guarantees $(\rho_q + \rho_{Topk} + \rho_{Influ} + \rho_{Rank})$-zCDP.*

5 EXPERIMENTS

In this section, we evaluate the quality and efficiency of the explanations generated by DPXPlain with the following questions:

(1) How accurate are the confidence intervals (CI) generated in Phase-2 by DPXPlain in validating the user question?
(2) How accurate are the noisy top-k explanations compared to true top-k, along with the CI of influence and rank in Phase-3?
(3) How efficient is DPXPlain in computing the explanation table?

We have implemented DPXPlain in Python 3.7.4 using the Pandas [97], NumPy [55], and SciPy [100] libraries. All experiments were run on Intel(R) Core(TM) i7-7700 CPU @ 3.60GHz with 32 GB of RAM. The source code can be found in [1].

5.1 Experiment Setup

We first detail the data, queries, questions, and parameters.

Datasets. We consider two datasets in our experiments.

- **`IPUMS-CPS` (real data)**: A dataset of Current Population Survey from the U.S. Census Bureau [51]. We focus on the survey data from year 2011 to 2019. The dataset contains 8 categorical attributes where domain sizes vary from 3 to 36 and one numerical attribute. The attribute `AGE` is discretized as 10 years per range, e.g., [0,10] is considered a single value. To set the domain of numerical attributes, we only include tuples with attribute `INCTOT` (the total income) smaller than 200k as a domain bound. The total size of the dataset is 1,146,552.
- **`Greman-Credit` (synthetic data)**: A corrected collection of credit data [54]. It includes 20 attributes where the domain sizes vary from 2 to 11 and a numerical attribute. Attributes `duration`, `credit-amount`, and `age` are discretized. The domain of attribute `good-credit` is zero or one. We synthesize the dataset to 1 million rows by combining a Bayesian network learner [8] and XGBoost [16] following the strategy of QUAIL [85].

Queries and Questions. The queries and questions used on the experiments are shown in Table 1.

Default setting of DPXPlain**.** Unless mentioned otherwise, the following default parameters are used: $\rho_q = 0.1$, $\rho_{Topk} = 0.5$, $\rho_{Influ} = 0.5$, $\rho_{Rank} = 1.0$, $\gamma = 0.95$, $k = 5$, and the number of conjuncts in explanation predicates $l = 1$ (Definition 3.3).

Table 1: Queries and questions for the experiments; `Valid` indicates if it is a valid question on the hidden true data.

Data	Query	Question	Valid
`IPUMS-CPS`	q_1: AVG(`INCTOT`) by `SEX`	I1: Why `Male` > `Female` ?	Yes
	q_2: `INCTOT` by `RELATE`	I2: Why `Grandchild` > `Foster children` ?	Yes
		I3: Why `Head/householder` > `Spouse` ?	No
	q_3: `INCTOT` by `EDUC`	I4: Why `Bachelor` > `High school` ?	Yes
		I5: Why `Grade 9` > `None or preschool` ?	No
`German-Credit`	q_4: AVG(`good-credit`) by `status`	G1: Why `no balance` > `no chk account` ?	Yes
	q_5: AVG(`good-credit`) by `purpose`	G2: Why `car (new)` > `car (used)`?	Yes
		G3: Why `business` > `vacation` ?	No
	q_6: AVG(`good-credit`) by `residence`	G4: Why `"< 1 yr"` > `">= 7 yrs"` ?	Yes
		G5: Why `"[1, 4) yrs"` > `"[4, 7) yrs"` ?	No

explanation predicate	Rel Influ 95%-CI		Rank 95%-CI		Rel Influ (hidden)	Rank (hidden)
	L	U	L	U		
`RELATE = "Head/householder"`	12.18%	12.52%	1	1	12.41%	1
`EDUC = "Bachelor's degree"`	7.10%	7.45%	2	3	7.32%	2
`RACE = "White"`	6.41%	6.75%	2	5	6.54%	3
`RELATE = "Spouse"`	5.70%	6.04%	2	5	6.01%	4
`CLASSWKR = "NIU"`	3.83%	4.18%	2	6	4.22%	5

Figure 5: Phase-3 of DPXPlain **for the case IPUMS-CPS.**

5.2 Case Studies

Case-1, IPUMS-CPS. We present a case study with the dataset `IPUMS-CPS` and default parameters. In **Phase-1**, the user submits a query q_1 from Table 1, and gets a noisy result: (`"Female"`, 31135.25) and (`"Male"`, 45778.46). The hidden true values are (`"Female"`, 31135.78) and (`"Male"`, 45778.39). Next, in **Phase-2**, since there is a gap of 14643.21 between the noisy avg-income from two groups, the user asks a question I1 from Table 1. The framework then quantifies the noise in the question by reporting a confidence interval of the difference between two groups as (14636.63, 14649.79). Since the interval does not include zero, the framework suggests that this is a valid question, which is correct. Finally, in **Phase-3**, the framework presents top-5 explanations to the user as Figure 5 shows. The last two columns are the true relative influences and true ranks. We correctly find the top-5 explanation predicates, and the first and fourth explanations together suggests that a married man tends to earn more than a married woman, which is supported by the wage disparities in the labor market [99]. The second and third explanations also match the wage disparities within the educated group and white people. The total runtime for preparing the explanations in Phase-2 and Phase-3 is 67 seconds.

Case-2, German-Credit. We now present a case study over the `German-Credit` dataset with default parameters. In **Phase-1**, the user submits a query q_4 from Table 1, and gets a noisy result: (`"no checking account"`, 0.526571) and (`"no balance"`, 0.574447). The true hidden result is (`"no checking account"`, 0.526574) and (`"no balance"`, 0.574466). Next, in **Phase-2**, since there is a gap of 0.047876 between the noisy avg-credit from two groups, the user asks a question G1 from Table 1. The framework then quantifies the noise in the question by reporting a confidence interval of the difference between two groups as (0.047786, 0.047967). Since the interval does not include zero, the framework

explanation predicate	Rel Influ 95%-CI		Rank 95%-CI		Rel Influ	Rank
	L	U	L	U	(hidden)	(hidden)
`existing-credits = "1"`	77.90%	78.99%	1	1	78.16%	1
`job = "skilled employee / official"`	71.21%	72.29%	1	2	71.83%	2
`sex-marst = "male : married/widowed"`	54.34%	55.42%	2	4	55.10%	3
`credit-amount = "(500, 2500]"`	50.01%	51.10%	2	5	50.27%	4
`credit-history = "no credits taken/all credits paid back duly"`	49.07%	50.16%	4	5	49.14%	5

Figure 6: Phase-3 of DPXPLAIN for the case German-Credit.

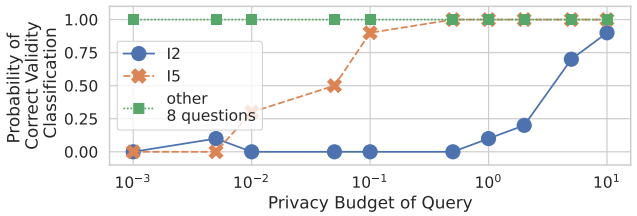


Figure 7: The probability of correctly validating user questions. All questions except I2 and I5 (Figure 7) are at 100%.

suggests that this is a valid question, which is correct. Finally, in **Phase-3**, the framework presents top-5 explanations to the user as Figure 6 shows. The last two columns are the true relative influences and true ranks. We correctly find the top-5 explanations, and the first explanation suggests that for a person who already has a credit in the bank, the bank tends to mark its credit as good if she has a checking account even with zero balance with higher probability than the case of no account, which is consistent to the intuition that a person having a credit account but no checking account is risky to the bank. The total runtime for preparing the explanations in Phase-2 and Phase-3 is 40 seconds.

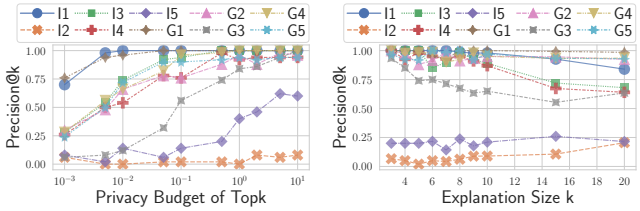
5.3 Accuracy and Performance Analysis

We detail our experimental analysis for the different questions and configurations of DPXPLAIN. All results are averaged over 10 runs.

Correctness of noise interval. In Phase-2 of DPXPLAIN, user has the option to either stop or proceed based on the confidence interval of the question. We evaluate a simple strategy for judging the validity of the question: if the interval contains non-positive numbers, the question is invalid, otherwise valid. From Figure 7, we find that 8 out of 10 questions (plotted together for clarity) from Table 1 are classified correctly with accuracy 100% given a wide range of privacy budget of query ρ_q. However, there are two questions, I2 and I5, only show high accuracy given a large privacy budget of $\rho_q = 10$. One reason is that the minimum group size involved in these questions is small compared to other questions, and, therefore, the partial confidence intervals in the denominators of the *AVG* query are low, which makes the final confidence intervals wider including negative numbers when it should not. For I2, the minimum group size is at least 600 times smaller than the other questions of `IPUMS-CPS` while this number for I5 is 60.

Accuracy of top-k explanation predicates. In Phase-3 of DPXPLAIN, we first select top-k explanation predicates. We measure the accuracy of the selection by Precision@k [56], the fraction of the selected top-k explanation predicates that are actually ranked within top-k. Another experiment on the full ranking is included in Appendix D.3 From Figure 8a, we find that the privacy budget of top-k selection ρ_{Topk} has a positive effect to Precision@k at k = 5 for various questions. When $\rho_{Topk} = 1.0$, all the questions except I2 and I5 have Precision@k ≥ 0.8. The selection accuracy of question I2 and I5 are generally lower because of small group sizes, and, therefore, the influences of explanation predicates are small and the rankings are perturbed by the noise more significantly.

From Figure 8b, we find that the trend of Precision@k by k is different across questions and there is no clear trend that Precision@k increases as k increases. For example, for G3, it first decreases from k=3 to k=5, but increases from k=5 to k=6. When k = 3, most questions have high Precision@k; this is because the highest three influences are much higher than the others, which makes the probability high to include the true top three. When k is large, for explanation predicates that have similar scores, they have equal probability to be included in top-k and therefore the top-k selected by the algorithm are different from the true top-k selections. The relationship between Precision@k and k depends on the distribution of all the explanation predicate influences. We also study the relationship between Precision@k at k=5 with the conjunction size l. For two questions I1 and G1, their Precision@k stays at 1.0 when l varies in 1, 2, and 3. Although the size of explanation predicates grows exponentially with the conjunction size l, DPXPLAIN can select the top-k explanation predicates with high accuracy even from a large space.

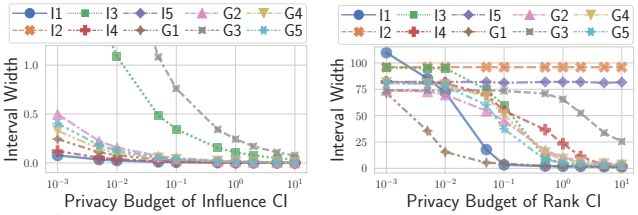


(a) Trend by ρ_{Topk}. **(b) Trend by k.**

Figure 8: Precision@k of top-k selection by DPXPLAIN.

Precision of relative influence and rank confidence Interval (CI). In Phase-3, the last step is to describe the selected top-k explanation predicates by a CI of relative influence and rank for each. To measure the precision of the description, we adopt the measure of **interval width** [50]. Figure 9 illustrates the average width of k CIs of relative influence and rank. From Figure 9a and 9b, we find that the increase of privacy budget ρ_{Influ} and ρ_{Rank} shrinks the interval width of relative influence CI and rank CI separately. In particular, when $\rho_{Influ} \geq 0.5$, 6 out of 10 questions have the interval width of relative influence CI ≤ 0.025 ; when $\rho_{Rank} \geq 1.0$, 2 questions have the interval width of rank CI ≤ 2 and 6 questions have this number ≤ 10. We also measure the **effect of confidence level** γ to the CI by changing γ from 0.1 to 0.9 by step size 0.1 and from 0.95 and 0.99. Figures can be found in Appendix D.2. The results show that it has a non-significant effect to the interval width, as it changes < 0.03 for the influence CI of 6 questions, and changes < 5 for the rank CI of 8 questions.

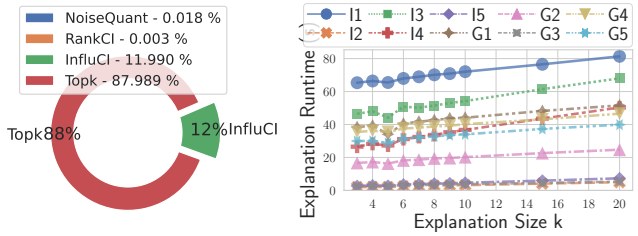
Runtime analysis. Finally, we analyze the runtime of DPXPLAIN for preparing the explanations from Phase-2 and Phase-3. From Figure 10a, a runtime breakdown in average for all the questions from Table 1 with total runtime 32 seconds in average, shows that 88% of the time is used for the top-k explanation predicate selection procedure, especially on computing the influences for all the explanation predicates. The next highest time usage is for computing the confidence interval of influence, which needs to evaluate



(a) Relative Influence **(b) Rank**

Figure 9: The width of confidence intervals by DPXPLAIN.

each sub queries. For the step noise quantification and confidence interval of rank, the time usage is not significant since the first only needs to find the image of two intervals and the second is a binary search. From Figure 10b, we find the runtime is linearly proportional to the size of explanations k, and the difference between questions is due to the difference of group sizes. We also find the runtime grows exponentially with the number of conjuncts l as the number of explanation predicates grows exponentially: for $l = 1, 2, 3$, the runtime about question I1 is 67, 3078 and 79634, and for question G1 it is 40, 1587 and 39922 seconds.



(a) Breakdown of time **(b) Trend by k**

Figure 10: Runtime analysis of DPXPLAIN.

The **summary of our findings** in the experiments is listed below:

(1) DPXPLAIN can correctly suggests the validity of the question with 100% accuracy for 8 out of 10 questions at $\rho_q = 0.1$.
(2) DPXPLAIN can select at least 80% of the true top-5 explanation predicates correctly for 8 out of 10 questions at $\rho_{Topk} = 0.5$, and the associated confidence intervals of relative influence have width ≤ 0.015 for 6 questions at $\rho_{Influ} = 0.5$ and the confidence intervals of rank have width ≤ 10 for 6 questions at $\rho_{Rank} = 1.0$.
(3) The runtime of DPXPLAIN for preparing the explanations is in average 32 seconds for the 10 examined questions.

6 RELATED WORK

We next survey related work in the fields of DP and explanations for query results. *To the best of our knowledge,* DPXPLAIN *is the first work that explains aggregate query results while satisfying DP.*

Explanations for query results. The database community has proposed several approaches to explaining aggregate and non-aggregate queries in multiple previous works. Proposed approaches include provenance [20, 30, 31, 57, 58, 67, 68, 98], intervention [86, 87, 103], entropy [47], responsibility [78, 79], Shapley values [72, 83], counterbalance [80] and augmented provenance [70], and several of these approaches have used predicates on tuple values as explanations like DPXPLAIN, e.g., [47, 70, 87, 103]. We note that any approaches that consider individual tuples or explicit tuple sets in any form as explanations (e.g., [30, 68, 72, 78]) cannot be applied in the DP setting since they would violate privacy. Among the other summarization or predicate-based approaches, Scorpion [103] explains outliers in query results with the intervention of most influential predicates. Our influence function (Section 4.2) is inspired by the influence function of Scorpion, but has been modified to deliver accurate results while satisfying DP. Another intervention-based work [87] that also uses explanation predicates, models inter-dependence among tuples from multiple relations with causal paths. DPXPLAIN does not support joins in the queries, which is a challenging future work (see Section 7).

Differential privacy. Private SQL query answering systems [36, 60, 64, 65, 74, 95, 102] consider a workload of aggregation queries with or without joins on a single or multi-relational database, but none supports explanation under differential privacy. The selection of private top-k candidates is well-studied by the community [11, 13, 14, 18, 21, 34, 39, 66, 71, 75, 76, 82, 96]. We adopt One-shot Top-k mechanism [82] since it is easy to understand. Private confidence interval is a new trend of estimating the uncertainty under differential privacy [15, 24, 49], however the current bootstrap based methods measure the uncertainty from both the sampling process and the noise injection, while we only focus on the second part which is likely to give tighter intervals. The most relevant work to the private rank estimation in our framework is private quantile [5, 23, 41, 52, 61, 69, 91], which is to find the value given a position such as median, but the problem of rank estimation in our setting is to find the position given a value.

Privacy and provenance. As mentioned earlier, data provenance is often used for explaining query results, mainly for non-aggregate queries. Within the context of provenance privacy [7, 12, 22, 25, 26, 88, 90, 93], one line of work [25–27] studied the preservation of workflow privacy (privacy of data transferred in a workflow with multiple modules or functions), with a privacy criterion inspired by l-diversity [73]. A recent work [32] explored what can be inferred about the *query* from provenance-based explanations and found that the query can be reversed-engineered from the provenance in various semirings [53]. To account for this, a follow-up paper [29] proposed an approach for provenance obfuscation that is based on abstraction. This work uses k-anonymity [92] to measure how many 'good' queries can generate concrete provenance that can be mapped to the abstracted provenance, thus quantifying the privacy of the underlying query. Devising techniques for releasing provenance of non-aggregate and aggregate queries while satisfying DP is an interesting research direction.

7 FUTURE WORK

There are several interesting future directions. First, the current DPXPLAIN framework does not support queries with joins. Unlike standard explanation frameworks like [103] where the join results can be materialized before running the explanation mechanism, in the DP settings with multiple relations, a careful sensitivity analysis of adding/removing tuples from different tables is needed [95]. Extending the framework to more general queries and questions is an important future work. Second, the complexity of the top-k selection algorithm is high since it needs to iterate over all the explanation predicates, leaving room for future improvements. Finally, other interesting notions of explanations for query answers (e.g., [70, 72, 80]) can be explored in the DP setting.

REFERENCES

[1] Dpxplain: Explaining query results under differential privacy. https://anonymous.4open.science/r/Private-Explanation-System-083D.

[2] New york city taxi and limousine commission (tlc) trip record data. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page. Accessed: 2022-01-01.

[3] J. M. Abowd. The us census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 2867–2867, 2018.

[4] H. Akoglu. User's guide to correlation coefficients. *Turkish journal of emergency medicine*, 18(3):91–93, 2018.

[5] D. Alabi, O. Ben-Eliezer, and A. Chaturvedi. Bounded space differentially private quantiles. *CoRR*, abs/2201.03380, 2022.

[6] Y. Amsterdamer, D. Deutch, and V. Tannen. Provenance for aggregate queries. In *PODS*, pages 153–164, 2011.

[7] P. Anderson and J. Cheney. Toward provenance-based security for configuration languages. In U. A. Acar and T. J. Green, editors, *4th Workshop on the Theory and Practice of Provenance, TaPP*, 2012.

[8] A. Ankan and A. Panda. pgmpy: Probabilistic graphical models using python. In *Proceedings of the 14th python in science conference (scipy 2015)*, pages 6–11. Citeseer, 2015.

[9] H. Asi and J. C. Duchi. Instance-optimality in differential privacy via approximate inverse sensitivity mechanisms. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[10] H. Asi and J. C. Duchi. Near instance-optimality in differential privacy. *arXiv preprint arXiv:2005.10630*, 2020.

[11] M. Bafna and J. Ullman. The price of selection in differential privacy. In *Conference on Learning Theory*, pages 151–168. PMLR, 2017.

[12] E. Bertino, G. Ghinita, M. Kantarcioglu, D. Nguyen, J. Park, R. S. Sandhu, S. Sultana, B. M. Thuraisingham, and S. Xu. A roadmap for privacy-enhanced secure data provenance. *J. Intell. Inf. Syst.*, 43(3):481–501, 2014.

[13] R. Bhaskar, S. Laxman, A. Smith, and A. Thakurta. Discovering frequent patterns in sensitive data. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 503–512, 2010.

[14] L. Bonomi and L. Xiong. Mining frequent patterns with differential privacy. *Proceedings of the VLDB Endowment*, 6(12):1422–1427, 2013.

[15] T. Brawner and J. Honaker. Bootstrap inference and differential privacy: Standard errors for free. *Unpublished Manuscript*, 2018.

[16] J. Brownlee. *XGBoost With python: Gradient boosted trees with XGBoost and scikit-learn*. Machine Learning Mastery, 2016.

[17] M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference*, pages 635–658. Springer, 2016.

[18] R. S. Carvalho, K. Wang, L. Gondara, and C. Miao. Differentially private top-k selection via stability on unknown domain. In *Conference on Uncertainty in Artificial Intelligence*, pages 1109–1118. PMLR, 2020.

[19] M. Cesar and R. Rogers. Bounding, concentrating, and truncating: Unifying privacy loss composition for data analytics. In *Algorithmic Learning Theory*, pages 421–457. PMLR, 2021.

[20] A. Chapman and H. V. Jagadish. Why not? In *SIGMOD*, pages 523–534, 2009.

[21] K. Chaudhuri, D. Hsu, and S. Song. The large margin mechanism for differentially private maximization. *arXiv preprint arXiv:1409.2177*, 2014.

[22] J. Cheney. A formal framework for provenance security. In *CSF*, pages 281–293, 2011.

[23] G. Cormode, T. Kulkarni, and D. Srivastava. Answering range queries under local differential privacy. *Proceedings of the VLDB Endowment*, 12(10):1126–1138, 2019.

[24] C. Covington, X. He, J. Honaker, and G. Kamath. Unbiased statistical estimation and valid confidence intervals under differential privacy. *arXiv preprint arXiv:2110.14465*, 2021.

[25] S. B. Davidson, S. Khanna, T. Milo, D. Panigrahi, and S. Roy. Provenance views for module privacy. In *PODS*, pages 175–186, 2011.

[26] S. B. Davidson, S. Khanna, S. Roy, J. Stoyanovich, V. Tannen, and Y. Chen. On provenance and privacy. In *ICDT*, pages 3–10, 2011.

[27] S. B. Davidson, S. Khanna, V. Tannen, S. Roy, Y. Chen, T. Milo, and J. Stoyanovich. Enabling privacy in provenance-aware workflow systems. In *CIDR*, pages 215–218, 2011.

[28] V. A. E. de Farias, F. T. Brito, C. Flynn, J. C. Machado, S. Majumdar, and D. Srivastava. Local dampening: Differential privacy for non-numeric queries via local sensitivity. *Proc. VLDB Endow.*, 14(4):521–533, 2020.

[29] D. Deutch, A. Frankenthal, A. Gilad, and Y. Moskovitch. On optimizing the trade-off between privacy and utility in data provenance. In *SIGMOD*, pages 379–391, 2021.

[30] D. Deutch, N. Frost, and A. Gilad. Explaining natural language query results. *VLDB J.*, 29(1):485–508, 2020.

[31] D. Deutch, N. Frost, A. Gilad, and T. Haimovich. Explaining missing query results in natural language. In *EDBT*, pages 427–430, 2020.

[32] D. Deutch and A. Gilad. Reverse-engineering conjunctive queries from provenance examples. In *EDBT*, pages 277–288, 2019.

[33] B. Ding, J. Kulkarni, and S. Yekhanin. Collecting telemetry data privately. *Advances in Neural Information Processing Systems*, 30, 2017.

[34] Z. Ding, D. Kifer, T. Steinke, Y. Wang, Y. Xiao, D. Zhang, et al. The permute-and-flip mechanism is identical to report-noisy-max with exponential noise. *arXiv preprint arXiv:2105.07260*, 2021.

[35] J. Dong, D. Durfee, and R. Rogers. Optimal differential privacy composition for exponential mechanisms. In *International Conference on Machine Learning*, pages 2597–2606. PMLR, 2020.

[36] W. Dong, J. Fang, K. Yi, Y. Tao, and A. Machanavajjhala. R2t: Instance-optimal truncation for differentially private query evaluation with foreign keys. In *Proc. ACM SIGMOD International Conference on Management of Data*, 2022.

[37] D. Dua and C. Graff. UCI machine learning repository, 2017.

[38] D. Durfee and R. Rogers. One-shot dp top-k mechanisms. DifferentialPrivacy.org, 08 2021. https://differentialprivacy.org/one-shot-top-k/.

[39] D. Durfee and R. M. Rogers. Practical differentially private top-k selection with pay-what-you-get composition. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. B. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 3527–3537, 2019.

[40] C. Dwork. Differential privacy and the us census. In *Proceedings of the 38th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, pages 1–1, 2019.

[41] C. Dwork and J. Lei. Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 371–380, 2009.

[42] 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.

[43] C. Dwork, A. Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[44] C. Dwork and G. N. Rothblum. Concentrated differential privacy. *arXiv preprint arXiv:1603.01887*, 2016.

[45] B. Efron. *The jackknife, the bootstrap and other resampling plans*. SIAM, 1982.

[46] B. Efron. Bootstrap confidence intervals for a class of parametric problems. *Biometrika*, 72(1):45–58, 1985.

[47] 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.

[48] Ú. Erlingsson, V. Pihur, and A. Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*, pages 1054–1067, 2014.

[49] C. Ferrando, S. Wang, and D. Sheldon. General-purpose differentially-private confidence intervals. *arXiv preprint arXiv:2006.07749*, 2020.

[50] C. Ferrando, S. Wang, and D. Sheldon. Parametric bootstrap for differentially private confidence intervals, 2021.

[51] S. Flood, M. King, R. Rodgers, S. Ruggles, J. R. Warren, and M. Westberry. Integrated public use microdata series, current population survey: Version 9.0 [dataset]. *Minneapolis, MN: IPUMS*, 2021. https://doi.org/10.18128/D030.V9.0.

[52] J. Gillenwater, M. Joseph, and A. Kulesza. Differentially private quantiles. In M. Meila and T. Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 3713–3722. PMLR, 2021.

[53] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *PODS*, pages 31–40, 2007.

[54] U. Grömping. South german credit data: Correcting a widely used data set. *Rep. Math., Phys. Chem., Berlin, Germany, Tech. Rep*, 4:2019, 2019.

[55] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

[56] J. L. Herlocker, J. A. Konstan, L. G. Terveen, and J. T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems (TOIS)*, 22(1):5–53, 2004.

[57] M. Herschel and M. A. Hernández. Explaining missing answers to SPJUA queries. *PVLDB*, 3(1):185–196, 2010.

[58] J. Huang, T. Chen, A. Doan, and J. F. Naughton. On the provenance of non-answers to queries over extracted data. *PVLDB*, 1(1):736–747, 2008.

[59] B. Jiang, X. Zhang, and T. Cai. Estimating the confidence interval for prediction errors of support vector machine classifiers. *J. Mach. Learn. Res.*, 9:521–540, 2008.

[60] N. Johnson, J. P. Near, and D. Song. Towards practical differential privacy for sql queries. *Proceedings of the VLDB Endowment*, 11(5):526–539, 2018.

[61] H. Kaplan, S. Schnapp, and U. Stemmer. Differentially private approximate quantiles. *CoRR*, abs/2110.05429, 2021.

[62] M. G. Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

[63] C. T. Kenny, S. Kuriwaki, C. McCartan, E. T. Rosenman, T. Simko, and K. Imai. The use of differential privacy for census data and its impact on redistricting: The case of the 2020 us census. *Science advances*, 7(41):eabk3283, 2021.

[64] I. Kotsogiannis, Y. Tao, X. He, M. Fanaeepour, A. Machanavajjhala, M. Hay, and G. Miklau. Privatesql: a differentially private sql query engine. *Proceedings of the VLDB Endowment*, 12(11):1371–1384, 2019.

[65] I. Kotsogiannis, Y. Tao, A. Machanavajjhala, G. Miklau, and M. Hay. Architecting a differentially private sql engine. In *CIDR*, 2019.

[66] J. Lee and C. W. Clifton. Top-k frequent itemsets via differentially private fp-trees. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 931–940, 2014.

[67] S. Lee, S. Köhler, B. Ludäscher, and B. Glavic. A sql-middleware unifying why and why-not provenance for first-order queries. In *ICDE*, pages 485–496, 2017.

[68] S. Lee, B. Ludäscher, and B. Glavic. PUG: a framework and practical implementation for why and why-not provenance. *VLDB J.*, 28(1):47–71, 2019.

[69] J. Lei. Differentially private m-estimators. *Advances in Neural Information Processing Systems*, 24, 2011.

[70] C. Li, Z. Miao, Q. Zeng, B. Glavic, and S. Roy. Putting things into context: Rich explanations for query answers using join graphs. In *SIGMOD*, pages 1051–1063, 2021.

[71] N. Li, W. H. Qardaji, D. Su, and J. Cao. Priv'basis: Frequent itemset mining with differential privacy. *Proc. VLDB Endow.*, 5(11):1340–1351, 2012.

[72] E. Livshits, L. E. Bertossi, B. Kimelfeld, and M. Sebag. The shapley value of tuples in query answering. In *ICDT*, volume 155, pages 20:1–20:19, 2020.

[73] A. Machanavajjhala, D. Kifer, J. Gehrke, and M. Venkitasubramaniam. *L*-diversity: Privacy beyond *k*-anonymity. *TKDD*, 1(1):3, 2007.

[74] R. McKenna, G. Miklau, M. Hay, and A. Machanavajjhala. Optimizing error of high-dimensional statistical queries under differential privacy. *Proc. VLDB Endow.*, 11(10):1206–1219, 2018.

[75] R. McKenna and D. R. Sheldon. Permute-and-flip: A new mechanism for differentially private selection. *Advances in Neural Information Processing Systems*, 33:193–203, 2020.

[76] F. McSherry and K. Talwar. Mechanism design via differential privacy. In *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS'07)*, pages 94–103. IEEE, 2007.

[77] F. D. McSherry. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data*, pages 19–30, 2009.

[78] A. Meliou, W. Gatterbauer, K. F. Moore, and D. Suciu. The complexity of causality and responsibility for query answers and non-answers. *Proc. VLDB Endow.*, 4(1):34–45, 2010.

[79] A. Meliou, W. Gatterbauer, S. Nath, and D. Suciu. Tracing data errors with view-conditioned causality. In T. K. Sellis, R. J. Miller, A. Kementsietsidis, and Y. Velegrakis, editors, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*, pages 505–516. ACM, 2011.

[80] Z. Miao, Q. Zeng, B. Glavic, and S. Roy. Going beyond provenance: Explaining query answers with pattern-based counterbalances. In *SIGMOD*, pages 485–502, 2019.

[81] I. Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[82] G. Qiao, W. J. Su, and L. Zhang. Oneshot differentially private top-k selection. *arXiv preprint arXiv:2105.08233*, 2021.

[83] A. Reshef, B. Kimelfeld, and E. Livshits. The impact of negation on the complexity of the shapley value in conjunctive queries. In D. Suciu, Y. Tao, and Z. Wei, editors, *PODS*, pages 285–297, 2020.

[84] R. Rogers and T. Steinke. A better privacy analysis of the exponential mechanism. DifferentialPrivacy.org, 07 2021. https://differentialprivacy.org/exponential-mechanism-bounded-range/.

[85] L. Rosenblatt, X. Liu, S. Pouyanfar, E. de Leon, A. Desai, and J. Allen. Differentially private synthetic data: Applied evaluations and enhancements. *arXiv preprint arXiv:2011.05537*, 2020.

[86] S. Roy, L. J. Orr, and D. Suciu. Explaining query answers with explanation-ready databases. *Proc. VLDB Endow.*, 9(4):348–359, 2015.

[87] S. Roy and D. Suciu. A formal approach to finding explanations for database queries. In C. E. Dyreson, F. Li, and M. T. Özsu, editors, *SIGMOD*, pages 1579–1590, 2014.

[88] P. Ruan, G. Chen, A. Dinh, Q. Lin, B. C. Ooi, and M. Zhang. Fine-grained, secure and efficient data provenance for blockchain. *Proc. VLDB Endow.*, 12(9):975–988, 2019.

[89] S. Ruggles, C. Fitch, D. Magnuson, and J. Schroeder. Differential privacy and census data: Implications for social and economic research. In *AEA papers and proceedings*, volume 109, pages 403–08, 2019.

[90] J. L. C. Sanchez, J. B. Bernabé, and A. F. Skarmeta. Towards privacy preserving data provenance for the internet of things. In *WF-IoT*, pages 41–46, 2018.

[91] A. Smith. Privacy-preserving statistical estimation with optimal convergence rates. In *Proceedings of the forty-third annual ACM symposium on Theory of computing*, pages 813–822, 2011.

[92] L. Sweeney. K-anonymity: A model for protecting privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, 2002.

[93] Y. S. Tan, R. K. L. Ko, and G. Holmes. Security and data accountability in distributed systems: A provenance survey. In *HPCC/EUC*, pages 1571–1578, 2013.

[94] J. Tang, A. Korolova, X. Bai, X. Wang, and X. Wang. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753*, 2017.

[95] Y. Tao, X. He, A. Machanavajjhala, and S. Roy. Computing local sensitivities of counting queries with joins. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 479–494, 2020.

[96] A. G. Thakurta and A. Smith. Differentially private feature selection via stability arguments, and the robustness of the lasso. In *Conference on Learning Theory*, pages 819–850. PMLR, 2013.

[97] The pandas development team. pandas-dev/pandas: Pandas, Feb. 2020.

[98] Q. T. Tran and C.-Y. Chan. How to conquer why-not questions. In *SIGMOD*, pages 15–26, 2010.

[99] G. Vandenbroucke. Married men sit atop the wage ladder. 24, 2018.

[100] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[101] L. Wasserman. *All of statistics: a concise course in statistical inference*, volume 26. Springer, 2004.

[102] R. J. Wilson, C. Y. Zhang, W. Lam, D. Desfontaines, D. Simmons-Marengo, and B. Gipson. Differentially private sql with bounded user contribution. *arXiv preprint arXiv:1909.01917*, 2019.

[103] E. Wu and S. Madden. Scorpion: Explaining away outliers in aggregate queries. *Proc. VLDB Endow.*, 6(8):553–564, 2013.

[104] Z. Yan, G. Li, and J. Liu. Private rank aggregation under local differential privacy. *International Journal of Intelligent Systems*, 35(10):1492–1519, 2020.

[105] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.

A THEOREMS AND PROOFS

A.1 Influence Function

Proposition 4.4. *[Influence Function Sensitivity] Given an explanation predicate p and an user question with respect to a group-by query with aggregation agg, the following holds:*

(1) If agg = COUNT, the sensitivity of Inf(p) is 4.
(2) If agg = SUM, the sensitivity of Inf(p) is 4 A_{agg}^{max}.
(3) If agg = AVG, the sensitivity of Inf(p) is 16 A_{agg}^{max}.

Proof. **(1) COUNT.** Recall the definition of influence function:

$$\begin{aligned}\textsc{Inf}(p;Q,D) =&\Big(\big(q(g_i(D)) - q(g_j(D))\big) \\ &- \big(q(g_i(\neg p(D))) - q(g_j(\neg p(D)))\big)\Big) \\ &\times \frac{\min_{t\in\{i,j\}} |g_t(\neg p(D))|}{\max_{t\in\{i,j\}} |g_t(D)|+1}\end{aligned}$$

We interpret and consider the following equations or notations:

$$\begin{aligned} q(D) &= |D| \\ \phi_i &= (\phi \wedge A_{gb} = \alpha_i) \\ g_i(D) &= \phi_i(D) \\ g_i(p(D)) &= (\phi_i \wedge p)(D) \\ g_i(\neg p(D)) &= (\phi_i \wedge \neg p)(D) \\ f(D) &= \min_{t\in\{i,j\}} |g_t(\neg p(D))| \\ g(D) &= \max_{t\in\{i,j\}} |g_t(D)| + 1 \\ h_i(D) &= q((\phi_i \wedge p)(D)) f(D)/g(D) \end{aligned}$$

Since q is a counting query , we have $q(g_i(D)) - q(g_i(\neg p(D))) = q(g_i(p(D)))$, and by replacing $g_i(p(D))$ with $(\phi_i \wedge p)(D)$ we have $q(g_i(D)) - q(g_i(\neg p(D))) = q((\phi_i \wedge p)(D))$. By further replacing the last numerator and denominator in the influence function with $f(D)$ and $g(D)$, we have $\textsc{Inf}(p;Q,D) = h_i(D) - h_j(D)$.

We prove the sensitivity bound by the following inequality chains.

$$\Delta_{\textsc{Inf}} = max_{D\approx D'} |\textsc{Inf}(p;Q_{CNT},D) - \textsc{Inf}(p;Q_{CNT},D'| \tag{8}$$

We first replace Inf according to $\textsc{Inf}(p;Q,D) = h_i(D) - h_j(D)$, and then apply Lemma A.6 (see Appendix A.3) to bound the sensitivity by the sum of sensitivities of h_i and h_j.

$$\leq \sum_{t\in\{i,j\}} \max_{|D'|=|D|+1} |h_t(D') - h_t(D)| \tag{9}$$

The second inequality is by Lemma A.9 (see Appendix A.3), since f is a non-negative query with sensitivity 1 and g is a monotonic positive and positive query with sensitivity 1.

$$\leq \sum_{t\in\{i,j\}} \frac{2|(\phi_t \wedge p)(D)| + f(D) + 1}{g(D)} \Delta_q \tag{10}$$

The next equality is by replacing the variables. Since q is a counting query, it has sensitivity $\Delta_q = 1$.

$$= \sum_{t\in\{i,j\}} \frac{2|(\phi_t \wedge p)(D)| + \min_{s\in\{i,j\}} |(\phi_s \wedge \neg p)(D)| + 1}{\max_{s\in\{i,j\}} |g_s(D)| + 1} \tag{11}$$

The third inequality is by the property of *min* and *max*, since $\min_{s\in\{i,j\}} |(\phi_s \wedge \neg p)(D)| \leq |(\phi_t \wedge \neg p)(D)|$ and $\max_{s\in\{i,j\}} |g_s(D)| \geq |g_t(D)|$.

$$\leq \sum_{t\in\{i,j\}} \frac{|(\phi_t \wedge p)(D)| + |(\phi_t \wedge p)(D)| + |(\phi_t \wedge \neg p)(D)| + 1}{|g_t(D)| + 1} \tag{12}$$

The next equality is due to that $\phi_t = (\phi_t \wedge p) \vee (\phi_t \wedge \neg p)$.

$$= \sum_{t\in\{i,j\}} \frac{|(\phi_t \wedge p)(D)| + (|\phi_t(D)| + 1)}{|g_t(D)| + 1} \tag{13}$$

The fourth inequality is due to that $|(\phi_t \wedge p)(D)| \leq |\phi_t(D)| = |g_t(D)| \leq |g_t(D)| + 1$.

$$\leq \sum_{t\in\{i,j\}} \frac{(|g_t(D)| + 1) + (|g_t(D)| + 1)}{|g_t(D)| + 1} \tag{14}$$

$$\leq 4 \tag{15}$$

(2) SUM. Similar to the proof of the sensitivity of CNT influence, but with $\Delta_q = A_{agg}^{max}$, which should be replaced at Equation (10).
(3) AVG.

$$\begin{aligned}&\textsc{Inf}(p;Q_{AVG},D) \\ =&\Big(\big(\frac{SUM(\phi_i(D),A_{agg})}{|\phi_i(D)|} - \frac{SUM(\phi_j(D),A_{agg})}{|\phi_j(D)|}\big) - \\ &\big(\frac{SUM((\phi_i \wedge \neg p)(D),A_{agg})}{|(\phi_i \wedge \neg p)(D)|} - \frac{SUM((\phi_j \wedge \neg p)(D),A_{agg})}{|(\phi_j \wedge \neg p)(D)|}\big)\Big) \\ &\min_{t\in\{i,j\}} |(\phi_t \wedge \neg p)(D)|\end{aligned}$$

Now we consider decompose this query into four parts (for example, $\frac{SUM(\phi_i(D),A_{agg})}{|\phi_i(D)|} \min_{t\in\{i,j\}} |(\phi_t \wedge \neg p)(D)|$ as one part) , and analyze the sensitivity for each part and finally sum up. Consider query q as summing up A_{agg} with sensitivity $\Delta_q = A_{agg}^{max}$. By Lemma A.9, we can show that the sensitivity of each part is 4 Δ_q. Together, the total sensitivity is bounded by 16 Δ_q. □

A.2 Private Explanations

Proposition 4.6. *Given an influence function Inf with sensitivity Δ_{Inf}, a set of explanation predicates $\mathcal{P}$, a privacy parameter ρ_{Topk} and a size parameter k, the following holds:*

(1) One-shot Top-k mechanism finds k explanation predicates while satisfying ρ_{Topk}-zCDP.
(2) Denote by $OPT^{(i)}$ the i-th highest (true) influence, and by $\mathcal{M}^{(i)}$ the i-th explanation predicate selected by the One-shot Top-k mechanism. For $\forall t$ and $\forall i \in \{1,2,\ldots,k\}$, we have

$$Pr\left[\textsc{Inf}(\mathcal{M}^{(i)}) \leq OPT^{(i)} - \frac{2\Delta_{Inf}}{\sqrt{8\rho_{Topk}/k}}(\ln(|\mathcal{P}|) + t)\right] \leq e^{-t} \tag{4}$$

Proof. **(1) Differential Privacy.** it is equivalent to iteratively applying k exponential mechanisms [43] that satisfies $\epsilon^2/8$-zCDP [19, 35, 39, 84] for each, where $\epsilon = \sqrt{8\rho_{Topk}/k}$ [38, 39], therefore in total it satisfies $(k\epsilon^2/8)$-zCDP which is also ρ_{Topk}-zCDP.

(2) Utility Bound. It is extended from the utility theorem of EM in Thm 3.11 of [43], which states that

$$Pr\left[\textsc{Inf}(\mathcal{M}^{(1)}) \le OPT^{(1)} - \frac{2\Delta_{\textsc{Inf}}}{\epsilon}(\ln(|\mathcal{P}|)+t)\right] \le e^{-t}$$

where $\epsilon = \sqrt{8\rho_{Topk}/k}$. To extend from $i = 1$ to $\forall i \in \{1, 2, \ldots, k\}$, we follow the original proof:

$$Pr[\textsc{Inf}(\mathcal{M}^{(i)}) \le c] \le \frac{|\mathcal{P}|\exp(\epsilon c/(2\Delta_{\textsc{Inf}}))}{\exp(\epsilon OPT^{(i)}/(2\Delta_{\textsc{Inf}}))}$$

by giving a upper bound and lower bound of the numerator and denominator. Replacing c with the appropriate value will give this theorem. □

Proposition 4.8. *Given a set of explanation predicates $\mathcal{P}$, an influence function Inf with global sensitivity Δ_{Inf}, and an integer $1 \le t \le |\mathcal{P}|$, $Inf(\text{rank}^{-1}(t))$ has sensitivity Δ_{Inf}.*

Proof. Drop $\mathcal{P}$ and Inf from $\text{rank}^{-1}(t; D, \mathcal{P}, \textsc{Inf})$ for simplicity. Next we show that for any two neighboring datasets $D' \sim D$, we have $|\textsc{Inf}(\text{rank}^{-1}(t; D'); D') - \textsc{Inf}(\text{rank}^{-1}(t; D); D)| \le \Delta_{\textsc{Inf}}$, which is equivalent to showing $-\Delta_{\textsc{Inf}} \le \textsc{Inf}(\text{rank}^{-1}(t; D'); D') - \textsc{Inf}(\text{rank}^{-1}(t; D); D) \le \Delta_{\textsc{Inf}}$.

Case 1, lower bound. This is to show that for any $D' \approx D$, we have $\textsc{Inf}(\text{rank}^{-1}(t; D'); D') - \textsc{Inf}(\text{rank}^{-1}(t; D); D) \ge -\Delta_{\textsc{Inf}}$.

By the definition of global sensitivity, for any explanation predicate p, we have $|\textsc{Inf}(p; D') - \textsc{Inf}(p; D)| \le \Delta_{\textsc{Inf}}$, and therefore $\textsc{Inf}(p; D') \ge \textsc{Inf}(p; D) - \Delta_{\textsc{Inf}}$. By replacing p with $\text{rank}^{-1}(j; D)$ for some j, we have $\textsc{Inf}(\text{rank}^{-1}(j; D); D') \ge \textsc{Inf}(\text{rank}^{-1}(j; D); D) - \Delta_{\textsc{Inf}}$. For any $j \le t$, by the property of ranking , we have $\textsc{Inf}(\text{rank}^{-1}(j; D); D) \ge \textsc{Inf}(\text{rank}^{-1}(t; D); D)$. Together, for any $j \le t$, we have $\textsc{Inf}(\text{rank}^{-1}(j; D); D') \ge \textsc{Inf}(\text{rank}^{-1}(j; D); D) - \Delta_{\textsc{Inf}} \ge \textsc{Inf}(\text{rank}^{-1}(t; D); D) - \Delta_{\textsc{Inf}}$. This means there are at least t elements in D' such that their scores are above $\textsc{Inf}(\text{rank}^{-1}(t; D); D) - \Delta_{\textsc{Inf}}$, which implies for the t-th largest score in D' we have $\textsc{Inf}(\text{rank}^{-1}(t; D'); D') \ge \textsc{Inf}(\text{rank}^{-1}(t; D); D) - \Delta_{\textsc{Inf}}$.

Case 2, upper bound. This is to show that for any $D' \approx D$, we have $\textsc{Inf}(\text{rank}^{-1}(t; D'); D') - \textsc{Inf}(\text{rank}^{-1}(t; D); D) \le \Delta_{\textsc{Inf}}$.

By the definition of global sensitivity, for any explanation predicate p, we have $|\textsc{Inf}(p; D') - \textsc{Inf}(p; D)| \le \Delta_{\textsc{Inf}}$, and therefore $\textsc{Inf}(p; D') \le \textsc{Inf}(p; D) + \Delta_{\textsc{Inf}}$. By replacing p with $\text{rank}^{-1}(j; D)$ for some j, we have $\textsc{Inf}(\text{rank}^{-1}(j; D); D') \le \textsc{Inf}(\text{rank}^{-1}(j; D); D) + \Delta_{\textsc{Inf}}$. For any $j \ge t$, by the property of ranking , we have $\textsc{Inf}(\text{rank}^{-1}(j; D); D) \le \textsc{Inf}(\text{rank}^{-1}(t; D); D)$. Together, for any $j \ge t$, we have $\textsc{Inf}(\text{rank}^{-1}(j; D); D') \le \textsc{Inf}(\text{rank}^{-1}(j; D); D) + \Delta_{\textsc{Inf}} \le \textsc{Inf}(\text{rank}^{-1}(t; D); D) + \Delta_{\textsc{Inf}}$. This means there are at most $t - 1$ elements in D' such that their scores can be above $\textsc{Inf}(\text{rank}^{-1}(t; D); D) + \Delta_{\textsc{Inf}}$, which implies for the t-th largest score in D' we have $\textsc{Inf}(\text{rank}^{-1}(t; D'); D') \le \textsc{Inf}(\text{rank}^{-1}(t; D); D) + \Delta_{\textsc{Inf}}$. □

Lemma A.1. *Given a set of predicates $\mathcal{P}$, an influence function Inf with global sensitivity Δ_{Inf} and a number t, then the function $s(D) = Inf(p; D) - Inf(\text{rank}^{-1}(t; D, \mathcal{P}, Inf); D)$ has global sensitivity $2\Delta_{Inf}$.*

Proof. The sensitivity of Inf is $\Delta_{\textsc{Inf}}$ by definition and the sensitivity of $\textsc{Inf}(\text{rank}^{-1}(t; D, \mathcal{P}, \textsc{Inf})$ is $\Delta_{\textsc{Inf}}$ by Proposition 4.8. By Lemma A.6, together it has sensitivity $2\Delta_{\textsc{Inf}}$. □

Theorem 4.10. *Given a database D, a predicate space $\mathcal{P}$, an influence function Inf with sensitivity Δ_{Inf}, explanation predicates $p_1, p_2, \ldots, p_k$, a confidence level γ, and a privacy parameter ρ_{Rank}, noisy binary search mechanism returns confidence intervals $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_k$ such that*

(1) Noisy binary search mechanism satisfies ρ_{Rank}-zCDP.

(2) For $\forall u \in [1, k]$, $\mathcal{I}_u$ is a γ level confidence interval of $\text{rank}(p_u)$.

Proof. Please find Appendix B.5 for the full description of the noisy binary search mechanism as Algorithm 4.

(1) Differential Privacy. Now we discuss why Algorithm 4 satisfies ρ_{Rank}-zCDP.

The main structure of Algorithm 4 is a for-loop of k explanation predicates from line 12 to 14, and within each for-loop, we first prepare the parameters at line 13 and 13, make two calls to the sub-routine RankBound and construct the confidence interval by the sub-routine outputs. We first show below that each call to the sub-routine RankBound with parameters (p, ρ, β, dir) satisfies ρ-zCDP. Given this is true, we then show that our two calls $\textsc{RankBound}(p_u, 0.1\rho, \beta, -1)$ and $\textsc{RankBound}(p_u, 0.9\rho, \beta, +1)$ at Line 14 satisfies 0.1ρ-zCDP and 0.9ρ-zCDP, which together satisfies ρ-zCDP by the composition rule (Lemma A.4). By line 13, we set $\rho = \rho_{Rank}/k$, therefore each loop satisfies (ρ_{Rank}/k)-zCDP, and after in total k loops, it satisfies ρ_{Rank}-zCDP by the composition rule (Lemma A.4).

Next we show that $\textsc{RankBound}(p, \rho, \beta, dir)$ satisfies ρ-zCDP, from line 1 to 11. We first prepare some parameters at the start of the sub-routine, which does not touch the data, and then enters a while loop with at most $N = \lceil \log_2 |\mathcal{P}| \rceil$ loops. Denote $s = \textsc{Inf}(p) - \textsc{Inf}(\text{rank}^{-1}(t))$. Within each loop, we add a Gaussian noise to a secret s at line 8. The value of s touches the sensitive data, but by adding a Gaussian noise to s, the release of $\hat{s}$ satisfies zCDP. By Theorem 2.8, with noise scale σ, it satisfies $(\Delta_q^2)/2\sigma^2$-zCDP where Δ_q is the sensitivity of the function that we want to release. Since we set $\sigma = (2\Delta_{\textsc{Inf}})/\sqrt{2(\rho/N)}$ at line 3 and the sensitivity of s is $2\Delta_{\textsc{Inf}}$ by Lemma A.1, it satisfies (ρ/N)-zCDP. Since we have at most N noisy releases of S using the Gaussian mechanism, by composition rule (Lemma A.4), the entire while loop satisfies ρ-zCDP, and so is the sub-routine.

(2) Confidence Interval. Now we discuss that the confidence interval outputted from the sub-routine $\textsc{RankBound}(p, \rho, \beta, dir)$, from line 1 to 11, is a γ-level confidence interval.

The sub-routine RankBound with direction as upper is mirror to the sub-algorithm RankBound with direction as lower. We first show that RankBound returns a bound in either upper or lower case such that it is a true bound with probability $\beta = \frac{\gamma+1}{2}$, therefore the target rank is within two bounds with probability γ. We give the proof for the case when direction is upper for the sub-algorithm RankBound, and skip the proof for the case when direction is lower due to the similarity.

The sub-routine RankBound is a random binary search algorithm with in total N loops. To ensure that the final t_{high} is a rank bound, one sufficient condition is that t_{high} is always an upper bound of rank during all the loops. Recall that in the noisy binary search, in each loop we first find t as the middle of t_{high} and t_{low}, check $s = \textsc{Inf}(p) - \textsc{Inf}(\text{rank}^{-1}(t)) \le 0$, add noise a Gaussian noise

to s to get $\hat{s}$ and compare $\hat{s}$ with margin, which is ξ in this case. If $\hat{s} \geq \xi$, notice that at line 9, we change t_{high} to t. If in this case, $s \leq 0$, which means t is not an upper bound of rank, we never have chance to make t_{high} to be a valid upper bound of rank since it will only decrease in the further loops. Therefore, We say a loop is a failure if during that loop, $s \leq 0$ but $\hat{s} > \xi$. To have a valid rank upper bound, it is necessary to have no loop failure during the entire noisy binary search. We next show that the probability of no such a failure occur is at least β.

See the chain of inequalities below.

$$Pr[\mathcal{I}_u^U \text{ is an upper bound of } \text{rank}(p_u; D, \mathcal{P}, I)] \tag{16}$$

The first inequality is due to the bound of the number of while loops. To be a rank bound, it cannot fail at each loop, therefore it has to success for all the N loops. These are independent events, so we can use a product for all the events happen together.

$$\geq (1 - Pr[\text{loop failure}])^N \tag{17}$$

The second inequality is due to the bound of $Pr[\text{loop failure}]$. Since any case such that $S \leq 0$ but $\hat{s} > \xi$ is considered as a loop failure, $\hat{s}$ is achieved by adding a Gaussian noise to s and ξ is a constant, the probability of a loop failure only depends on the value of s. Since here we have a condition about $s \leq 0$, $\sup_{s \leq 0} Pr[\hat{s} > \xi]$ is an upper bound of $Pr[\text{loop failure}]$.

$$\geq (1 - \sup_{s \leq 0} Pr[\hat{S} > \xi])^N \tag{18}$$

The next equality is because $\sup_{s \leq 0} Pr[\hat{s} > \xi] = Pr[N(0, \sigma^2) > \xi]$. Recall that $\hat{s} = s + N(0, \sigma^2)$ in line 8, therefore $\sup_{s \leq 0} Pr[\hat{s} > \xi] = \sup_{s \leq 0} Pr[s + N(0, \sigma^2) > \xi] = \sup_{s \leq 0} Pr[N(0, \sigma^2) > \xi - s]$. Since $Pr[N(0, \sigma^2) > \xi - s]$ increases as s increases, it achieves maximum at $s = 0$ for $s \leq 0$. Therefore, $\sup_{s \leq 0} Pr[\hat{s} > \xi] = Pr[N(0, \sigma^2) > \xi]$.

$$= Pr[\mathcal{N}(0, \sigma^2) \leq \xi]^N \tag{19}$$

The third bound is due to Chernoff bound of the Q-function (Lemma A.3). Since $Pr[\mathcal{N}(0, \sigma^2) \leq \xi] = 1 - Pr[\mathcal{N}(0, 1) > \xi/\sigma]$, by Chernoff bound we have $Pr[\mathcal{N}(0, 1) > \xi/\sigma] \leq \exp(-(\xi/\sigma)^2/2)$ and therefore $Pr[\mathcal{N}(0, \sigma^2) \leq \xi] \geq 1 - \exp(-(\xi/\sigma)^2/2)$.

$$\geq \left(1 - \exp(-(\xi/\sigma)^2/2)\right)^N \tag{20}$$

The fourth bound is due to $(1 + x)^r \geq 1 + rx$ for $x \geq -1$ and $r \geq 1$.

$$\geq 1 - N \exp(-(\xi/\sigma)^2/2) \tag{21}$$

The final equality is by plugging $\xi = \sigma\sqrt{2 \ln(N/(1 - \beta))}$.

$$= \beta \tag{22}$$

Similarly, we have $Pr[\mathcal{I}_u^L$ is a lower bound of $\text{rank}(p_u; D, \mathcal{P}, I)] \geq \beta$. Together, the probability of $\mathcal{I}_u$ is a γ level confidence interval of $\text{rank}(p_u; D, \mathcal{P}, I)$ equals to both events $\mathcal{I}_u^U$ is an upper bound of $\text{rank}(p_u; D, \mathcal{P}, I)$ and $\mathcal{I}_u^L$ is a lower bound of $\text{rank}(p_u; D, \mathcal{P}, I)$ happen together, which is greater than or equal to the probability sum of each single event minus one (Lemma A.8, which is $\beta + \beta - 1 = 2\beta - 1$. By plugging $\beta = (\gamma + 1)/2$ from line 13, we have $2\beta - 1 = \gamma$, which is the confidence interval level for the final confidence interval. □

Theorem A.2. *Given a database D, a predicate space $\mathcal{P}$, an influence function Inf with sensitivity Δ_{Inf}, explanation predicates $p_1, p_2, \ldots, p_k$, a confidence level γ, and a privacy parameter ρ_{Rank}, noisy binary search mechanism returns confidence intervals $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_k$ such that for $\forall u \in \{1, 2, \ldots, k\}$ and $\forall x \geq 0$, the confidence interval $\mathcal{I}_u = (\mathcal{I}_u^L, \mathcal{I}_u^U)$ satisfies*

$$Pr[A \leq \text{Inf}(p_u) \leq B] \geq 1 - 2e^{-x}$$

where $A = \text{Inf}(\text{rank}^{-1}(\mathcal{I}_u^L)) - (|\xi_{-1}| + \sigma_{-1}\sqrt{2(x + \ln N)})$ and $B = \text{Inf}(\text{rank}^{-1}(\mathcal{I}_u^U)) + (|\xi_{+1}| + \sigma_{+1}\sqrt{2(x + \ln N)})$.

Proof. We first show that each utility bound has probability $\geq 1 - e^{-x}$, then use the union probability rule to show together it is bounded by $\geq 1 - 2e^{-x}$.

Consider the upper utility bound. One sufficient condition for the upper utility bound to be true is that $\text{Inf}(\text{rank}^{-1}(\mathcal{I}_{rank}^U; D, \mathcal{P}, \text{Inf}); D) \geq \text{Inf}(p, D) - (\xi + \sigma\sqrt{2(x + \ln N)})$ is always true. Similar to the proof of confidence rank bound, here we say a loop is a failure if $S > \xi + \sigma\sqrt{2(x + \ln N)}$ but $\hat{S} \leq \xi$. The proof of the inequality chain below is similar to the proof of confidence interval, except for that the third inequality is due to that $(1 - a)^x \geq 1 - ax$ for $a \in (0, 1)$ and $x \geq 1$.

$$\begin{aligned}
&Pr[\text{Inf}(\text{rank}^{-1}(\mathcal{I}_{rank}^U; D, \mathcal{P}, \text{Inf})) - \sigma(\xi + \sqrt{2(x + \ln N)})] \\
\geq &(1 - Pr[\text{loop failure}])^N \\
\geq &(1 - \sup_{S > \xi + \sigma\sqrt{2(x + \ln N)}} Pr[\hat{S} \leq \xi])^N \\
= &(1 - Pr[N(0, \sigma^2) \leq -\sigma\sqrt{2(x + \ln N)}])^N \\
\geq &1 - NPr[N(0, \sigma^2) \leq -\sigma\sqrt{2(x + \ln N)}] \\
\geq &1 - N \exp(-(\sqrt{2(x + \ln N)})^2/2) \\
= &1 - \exp(-x)
\end{aligned}$$

□

A.3 Supplementary

Lemma A.3 (Chernoff bound of Q function). *Given a Q function: $Q(x) = Pr[X > x]$, where $X \sim N(0, 1)$ is a standard Gaussian distribution, if $x \geq 0$, we have*

$$Q(x) \leq \exp(-x^2/2)$$

Proof. By Chernoff bound, we have $Pr[X > x] \leq E[e^{tX}]/e^{tx}$ for any $t \geq 0$. By the property of Gaussian distribution, we have $E[e^{tX}] = e^{t^2/2}$. Together, we have $Pr[X > x] \leq e^{t^2/2 - tx}$. Since $x \geq 0$, we can choose $t = x$, and have $Pr[X > x] \leq e^{-t^2/2}$. □

Lemma A.4 (Composition [17]). *Let $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ and $\mathcal{M}' : \mathcal{X}^n \to \mathcal{Z}$ be randomized algorithms. Suppose $\mathcal{M}$ satisfies ρ-zCDP and $\mathcal{M}'$ satisfies ρ'-zCDP. Define $\mathcal{M}'' : \mathcal{X}^n \to \mathcal{Y} \times \mathcal{Z}$ by $\mathcal{M}''(x) = (\mathcal{M}(x), \mathcal{M}'(x))$. Then $\mathcal{M}''$ satisfies $(\rho + \rho')$-zCDP.*

Lemma A.5 (Postprocessing [17]). *Let $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ and $f : \mathcal{Y} \to \mathcal{Z}$ be randomized algorithms. Suppose $\mathcal{M}$ satisfies ρ-zCDP. Define $\mathcal{M}' : \mathcal{X}^n \to \mathcal{Z}$ by $\mathcal{M}'(x) = f(\mathcal{M}(x))$. Then $\mathcal{M}'$ satisfies ρ-zCDP.*

Lemma A.6. *Given two functions f ang g with sensitivities Δ_f and Δ_g, the sum of two functions have sensitivity $\Delta_f + \Delta_g$*

Proof. BY definition, we have $\max_{D\approx D'}|f(D) - f(D')| \le \Delta_f$ and $\max_{D\approx D'}|g(D) - g(D')| \le \Delta_g$. Therefore, $\max_{D\approx D'}|(f(D) + g(D)) - (f(D') + g(D'))| = \max_{D\approx D'}|(f(D) - f(D') + (g(D) - g(D'))| \le \max_{D\approx D'}|(f(D) - f(D')| + \max_{D\approx D'}|(g(D) - g(D'))| = \Delta_f + \Delta_g$. The inequality is due to the property of absolute. □

Lemma A.7 (Gaussian Confidence Interval [101]). *Given a Gaussian random variable $Z \sim N(\mu, \sigma^2)$ with unknown location parameter μ and known scale parameter σ. Let $\mathcal{I}^L = Z - \sigma\sqrt{2}\,\mathrm{erf}^{-1}(\gamma)$ and $\mathcal{I}^U = Z + \sigma\sqrt{2}\,\mathrm{erf}^{-1}(\gamma)$, then $\mathcal{I} = (\mathcal{I}^L, \mathcal{I}^U)$ is a γ level confidence interval of μ.*

Proof. By Theorem 6.16 from the text book [101]. □

Lemma A.8. *Given events $A_1, A_2, \ldots, A_\ell$, the following inequality holds:*

$$Pr[\bigwedge_{i=1}^{\ell} A_i] \ge \sum_{i=1}^{\ell} Pr[A_i] - (\ell - 1)$$

Proof. First we show that given events A and B, we have $Pr[A \wedge B] \ge Pr[A] + Pr[B] - 1$ since $1 \ge Pr[A \vee B] = Pr[A] + Pr[B] - Pr[A \wedge B]$. Next we show that

$$Pr[\bigwedge_{i=1}^{\ell} A_i] \ge Pr[\bigwedge_{i=1}^{\ell-1} A_i] + Pr[A_\ell] - 1$$

using the previous rule. This gives a recursive expression and can be reduced to the final formula in the lemma. □

Lemma A.9. *Given a COUNT or SUM query q with sensitivity Δ_q, a predicate ϕ, a non-negative query $f : \mathcal{D} \to \mathbb{N}_0$ with sensitivity 1 and another monotonic [11] and positive query $g : \mathcal{D} \to \mathbb{N}^+$ with sensitivity 1. Denote $h(D)$ as*

$$h(D) = q(\phi(D))\frac{f(D)}{g(D)}$$

For any two neighboring datasets D and D' such that $|D'| = |D| + 1$, we have

$$|h(D') - h(D)| \le \frac{2|\phi(D)| + f(D) + 1}{g(D)}\Delta_q$$

Proof. Denote $x = q(\phi(D))$, $x' = q(\phi(D'))$, $n = |\phi(D)|$. Since x is the aggregation over tuples from $\phi(D)$ and x has sensitivity Δ_q, we have $|x| \le n\Delta_q$. Denote $\delta_x = x' - x$. Since x has sensitivity Δ_q, we have $|\delta_x| \le \Delta_q$. Since $g(D)$ is monotonic and has sensitivity 1, we have $g(D) \le g(D') \le g(D) + 1$. Since f has sensitivity 1, we have $|f(D) - f(D')| \le 1$.

$$\begin{aligned}
&|h(D') - h(D)| \\
=&|x'\frac{f(D')}{g(D')} - x\frac{f(D)}{g(D)}| \\
=&|(x + \delta_x)\frac{f(D')}{g(D')} - x\frac{f(D)}{g(D)}| \\
=&|x\left(\frac{f(D')}{g(D')} - \frac{f(D)}{g(D)}\right) + \delta_x\frac{f(D')}{g(D')}|
\end{aligned}$$

Now we divide into two cases depending on the sign of the factor of x in the formula above.

Case 1, the factor of x is non-negative.

$$\begin{aligned}
&|h(D') - h(D)| \\
\le&n\Delta_q\left(\frac{f(D')}{g(D')} - \frac{f(D)}{g(D)}\right) + \Delta_q\frac{f(D')}{g(D')} \\
=&\left[(n+1)\frac{f(D')}{g(D')} - n\frac{f(D)}{g(D)}\right]\Delta_q \\
\le&\left[(n+1)\frac{f(D)+1}{g(D)} - n\frac{f(D)}{g(D)}\right]\Delta_q \\
\le&\frac{f(D)+n+1}{g(D)}\Delta_q
\end{aligned}$$

Case 2, the factor of x is non-positive.

$$\begin{aligned}
&|h(D') - h(D)| \\
\le&n\Delta_q\left(\frac{f(D)}{g(D)} - \frac{f(D')}{g(D')}\right) + \Delta_q\frac{f(D')}{g(D')} \\
\le&\left[n\left(\frac{f(D')+2}{g(D')} - \frac{f(D')}{g(D')}\right) + \frac{f(D')}{g(D')}\right]\Delta_q \\
\le&\frac{2n + f(D')}{g(D')}\Delta_q \\
\le&\frac{2n + f(D) + 1}{g(D)}\Delta_q
\end{aligned}$$

In conclusion, $|h(D') - h(D)| \le \frac{2n+f(D)+1}{g(D)}\Delta_q$. □

[11] A query q is monotonic if for any two databases D' and D such that $|D'| \ge |D|$, we have $q(D') \ge q(D)$.

B EXTRA ALGORITHM DESCRIPTIONS

B.1 Confidence Interval of Question

In this section, we elaborate the algorithm of Section 4.1 in the form of pseudo codes.

Confidence interval for COUNT and SUM. In Algorithm 1, at line 2, we set the noise scale σ according to aggregation as *COUNT* (*SUM*), and at line 6 and 7, we set the confidence interval from the standard properties of Gaussian distribution by a margin as $\sqrt{2}(\sqrt{2}\sigma)\,\mathrm{erf}^{-1}(\gamma)$ for both bounds [12] [101] .

Confidence interval for AVG. In Algorithm 1, at line 9, we set the sub confidence level $\beta = 1-(1-\gamma)/4$ for each individual confidence interval, so that the final confidence level for $o_i - o_j$ is γ. At line 10 and 11, we set the noise level σ for *SUM* and *COUNT*. From line 12 to 16, we extract all the intermediate numerators and denominators, and construct individual confidence intervals. At line 17 and 18, we compute the infimum and supremum of the image of the cross product of individual confidence intervals, which is also the confidence interval at level γ.

Algorithm 1 Compute Confidence Interval of User Question

Require: A user question $Q = (\alpha_i, >, \alpha_j)$ with respect to the query `SELECT` A_{gb}`,` `agg(`A_{agg}`)` `FROM R WHERE` ϕ `GROUP BY` A_{gb}, the noisy results $\hat{o}_i$ and $\hat{o}_j$, the privacy budget ρ_q for the private query answering, and the confidence level γ.

Ensure: A γ-level confidence interval of $o_i - o_j$.

1: **if** agg = *COUNT* or agg = *SUM* **then**
2: **if** agg = *COUNT* **then**
3: $\sigma \leftarrow 1/\sqrt{2\rho_q}$
4: **else if** agg = *SUM* **then**
5: $\sigma \leftarrow A^{max}_{agg}/\sqrt{2\rho_q}$
6: $\mathcal{I}^L \leftarrow \hat{o}_i - \hat{o}_j - 2\sigma\,\mathrm{erf}^{-1}(\gamma)$
7: $\mathcal{I}^U \leftarrow \hat{o}_i - \hat{o}_j + 2\sigma\,\mathrm{erf}^{-1}(\gamma)$
8: **else if** agg = *AVG* **then**
9: $\beta \leftarrow 1-(1-\gamma)/4$
10: $\sigma_S \leftarrow A^{max}_{agg}/\sqrt{2\rho_q/2}$
11: $\sigma_C \leftarrow 1/\sqrt{2\rho_q/2}$
12: **for** $t \in \{i, j\}$ **do** /* *Recall that* $\hat{o}_t = \hat{o}^S_t/\hat{o}^C_t$ */
13: $\hat{o}^S_t \leftarrow$ numerator of $\hat{o}_t$.
14: $\mathcal{I}^S_t \leftarrow (\hat{o}^S_t - \sigma_S\sqrt{2}\,\mathrm{erf}^{-1}(\beta), \hat{o}^S_t + \sigma_S\sqrt{2}\,\mathrm{erf}^{-1}(\beta))$
15: $\hat{o}^C_t \leftarrow$ denominator of $\hat{o}_t$.
16: $\mathcal{I}^C_t \leftarrow (\hat{o}^C_t - \sigma_C\sqrt{2}\,\mathrm{erf}^{-1}(\beta), \hat{o}^C_t + \sigma_C\sqrt{2}\,\mathrm{erf}^{-1}(\beta))$
17: $\mathcal{I}^L \leftarrow \inf\{\mathcal{I}^S_i/\mathcal{I}^C_i - \mathcal{I}^S_j/\mathcal{I}^C_j\}$
18: $\mathcal{I}^U \leftarrow \sup\{\mathcal{I}^S_i/\mathcal{I}^C_i - \mathcal{I}^S_j/\mathcal{I}^C_j\}$
19: $\mathcal{I} \leftarrow (\mathcal{I}^L, \mathcal{I}^U)$
20: **return** $\mathcal{I}$

B.2 Influence Function Monotonicity

The influence function $\textsc{Inf}(p)$ from Section 4.2 is not monotone w.r.t. predicate implication even without the normalizing factor in the function. We demonstrate this property in the example below.

Example B.1. Start with a database with three binary attributes: A, B, C and two tuples: (0, 0, 0), (1, 0, 1). Consider an $agg = COUNT$ query with group by on A, so we have $agg(g_0(D)) = 1$ and $agg(g_1(D)) = 1$ for two groups $A = 0$ and $A = 1$. Consider three explanation predicates for the user question $(\alpha_0, \alpha_1, >)$ (note that the noisy values can be different from the true values): $p_1 : B = 0$, $p_2 : B = 0 \wedge C = 0$ and $p_3 : B = 0 \wedge C = 1$, which satisfy $p_2 \Rightarrow p_1$ and $p_3 \Rightarrow p_1$. However, while $\textsc{Inf}(p_1) = 0$, we have $\textsc{Inf}(p_2) = 1$ and $\textsc{Inf}(p_3) = -1$, i.e., $\textsc{Inf}(p_3) < \textsc{Inf}(p_1) < \textsc{Inf}(p_2)$.

B.3 Private Top-k Explanation Predicates

In this section, we restate the One-shot Top-k mechanism based on the exponential mechanism [43] from Section 4.3.1 with pseudo codes.

Given a score function $\mathrm{s} : \mathcal{P} \to \mathcal{R}$ that maps an explanation predicate p to a number, the exponential mechanism (EM) [43] randomly samples p from $\mathcal{P}$ with probability proportional to $exp(\epsilon\,\mathrm{s}(p)/(2\Delta_{\mathrm{s}}))$ with some privacy parameter ϵ and satisfies $(\epsilon^2/8)$-zCDP [19, 35, 39, 84]. The higher the score is, the more possible that an explanation predicate is selected. In DPXPlain, we use the influence function as the score function.

We denote the exponential mechanism as $\mathcal{M}_E$. To find 'top-k' explanation predicate satisfying DP, we can first apply $\mathcal{M}_E$ to find one explanation predicate, remove it from the entire explanation predicate space, and then apply $\mathcal{M}_E$ again until k explanation predicates are found. It was shown by previous work [38, 39] that this process is identical to adding i.i.d. Gumbel noise[13] to each score and releasing the top-k predicates by the noisy scores (i.e., there is no need to remove predicates after sampling) . We, therefore, use this result to devise a similar solution that is presented in Algorithm 2. In line 1, we set the noise scale. In lines 2–4, we randomly sample Gumbel noise with scale σ and add it to the influence of each explanation predicate from the space $\mathcal{P}$. In line 5, we sort the noisy scores in the descending order, and in line 6, we find the top-k explanation predicates by their noisy scores. This algorithm satisfies ρ_{Topk}-zCDP (as formally stated in Proposition 4.6), and can be applied to questions on SUM, COUNT, or AVG queries, with different score functions and sensitivity values for different aggregates.

Algorithm 2 Noisy Top-k Predicates

Require: An influence function Inf with sensitivity Δ_{Inf}, a set of explanation predicates $\mathcal{P}$, a privacy parameter ρ_{Topk} and a size parameter k.

Ensure: Top-k explanation predicates.

1: $\sigma \leftarrow 2\Delta_{\mathrm{Inf}}\sqrt{k/(8\rho_{Topk})}$
2: **for** $u \leftarrow 1 \ldots |\mathcal{P}|$ **do**
3: $s_u \leftarrow \textsc{Inf}(p_u) + Gumbel(\sigma)$
4: Sort $s_1 \ldots s_{|\mathcal{P}|}$ in the descending order.
5: Let $p_1, p_2, \ldots, p_k$ be the top-k elements in the list.
6: **return** $p_1, p_2, \ldots, p_k$

B.4 Private Confidence Interval of Influence

In this section, we elaborate the algorithm of Section 4.3.2 in the form of pseudo codes.

[12] erf^{-1} is the inverse function of the error function erf.

[13] For a Gumbel noise $Z \sim Gumbel(\sigma)$, its CDF is $Pr[Z \le z] = \exp(-\exp(-z/\sigma))$.

Algorithm 3 takes a privacy budget ρ_{Influ} as input. In Line 2 we divide the privacy budget ρ_{Influ} into k equal portions for each explanation predicate p_u for $u \in \{1, \ldots, k\}$. In Line 3, we calibrate the noise scale according to the sensitivity of the influence function. In Line 9, we add a Gaussian noise to the influence $\textsc{Inf}(p_u)$ of explanation predicate p_u, and finally in Lines 10 and 11, we derive the confidence interval based on the Gaussian property [101].

Algorithm 3 Compute Confidence Interval of Influence

Require: An influence function Inf with respect to the question $(\alpha_i, >, \alpha_j)$, k explanation predicates $p_1, p_2, \ldots, p_k$, a private database D, a privacy budget ρ_{Influ}, and a confidence level γ.

Ensure: A list of γ-level confidence intervals of the influence $\textsc{Inf}(p_u)/(\hat{o}_i - \hat{o}_j)$ for $u \in \{1, 2, \ldots, k\}$.

1: **for** $u \in \{1, 2, \ldots, k\}$ **do**
2: $\quad \rho \leftarrow \rho_{Influ}/k$
3: $\quad$ **if** agg = $COUNT$ **then**
4: $\quad\quad \sigma \leftarrow 4/\sqrt{2\rho}$
5: $\quad$ **else if** agg = SUM **then**
6: $\quad\quad \sigma \leftarrow 4A_{agg}^{max}/\sqrt{2\rho}$
7: $\quad$ **else if** agg = AVG **then**
8: $\quad\quad \sigma \leftarrow 16A_{agg}^{max}/\sqrt{2\rho}$
9: $\quad \hat{\textsc{Inf}} \leftarrow \textsc{Inf}(p_u) + N(0, \sigma^2)$
10: $\quad \mathcal{I}_u^L \leftarrow \hat{\textsc{Inf}} - \sqrt{2}\sigma \operatorname{erf}^{-1}(\gamma)$
11: $\quad \mathcal{I}_u^U \leftarrow \hat{\textsc{Inf}} + \sqrt{2}\sigma \operatorname{erf}^{-1}(\gamma)$
12: $\quad \mathcal{I}_u \leftarrow (\mathcal{I}_u^L, \mathcal{I}_u^U)$
13: **return** $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_k$

B.5 Problem 5: Private Confidence Interval of Rank

In this section, we elaborate the algorithm of Section 4.3.2, noisy binary search mechanism, in the form of pseudo codes as Algorithm 4 shows.

In line 1, RankBound takes four parameters: an explanation predicate p, a privacy budget ρ, a sub confidence level β and a direction $dir \in \{-1, +1\}$. It guarantees that it will find a lower ($dir = -1$) or upper ($dir = +1$) bound of rank with confidence β for the explanation predicate p using privacy budget ρ. In line 2, we set the maximum depth N of the binary search. In line 3, we set the noise scale σ_{-1} or σ_{+1}, which depends on the sensitivity of $\textsc{Inf}(p) - \textsc{Inf}(\text{rank}^{-1}(t))$ (in line 8), which is $2\Delta_{\textsc{Inf}}$; and the number of Gaussian mechanisms used in the binary search, which is N. In line 4, we set the margin ξ_{+1} or ξ_{-1}, which will be discussed in line 9. In line 5, we initialize the binary search by setting two pointers, t_{low} and t_{high}, as the first and last rank. In lines 6–10 there is a while loop for the binary search. In line 7, we pick a rank that is at the middle of two pointers. In line 8, we add a Gaussian noise with scale σ to the difference between the influence of the target explanation predicate p and the influence of the explanation predicate that has rank t. From line 9 to 10 we update one of the pointer according to the relationship between the noisy difference and the margin ξ_{dir}. If we are trying to find a rank upper bound ($dir = +1$), we want the binary search to find the rank such that the difference (without noise) is above zero. Due to the noise injected, even if the noisy difference is above zero, the true difference could be negative. To secure the goal with high probability, we requires the noisy difference to be above a margin ξ_{dir}, as shown in line 9. In this case, we narrow down the search space by moving t_{high} to $\max\{t - 1, 1\}$. The strategy is similar when we are looking for a rank lower bound ($dir = -1$).

Now, we describe the usage of the sub-routine RankBound. We repeat the following for each explanation predicate. In line 13, we allocate an even portion from the total privacy budget ρ_{Rank}, and set the sub confidence level to $\beta = (\gamma + 1)/2$ so the final confidence interval has confidence level $2\beta - 1 = \gamma$ by the rule of union bound. In lines 14, we divide the privacy budget ρ, and make two calls to the sub-routine RankBound to find a rank upper bound and a rank lower bound for the explanation predicate p_u, and finally merge them into a single confidence interval. We spend more budget for the rank upper bank since this is more important in the explanation.

Algorithm 4 Compute Confidence Interval of Rank

Require: A dataset D, a predicate space $\mathcal{P}$, an influence function Inf with sensitivity $\Delta_{\textsc{Inf}}$, explanation predicates $p_1, p_2, \ldots, p_k$, a confidence level γ, and a privacy parameter ρ_{Rank}.

Ensure: A list of γ-level confidence intervals of the influence $\text{rank}(p_u; D, \mathcal{P}, \textsc{Inf})$ for $u \in \{1, 2, \ldots, k\}$.

1: **function** RankBound(p, ρ, β, dir)
2: $\quad N \leftarrow \lceil \log_2 |\mathcal{P}| \rceil$
3: $\quad \sigma_{dir} \leftarrow (2\Delta_{\textsc{Inf}})/\sqrt{2(\rho/N)}$
4: $\quad \xi_{dir} \leftarrow \sigma_{dir}\sqrt{2\ln(N/(1-\beta))} \times dir$
5: $\quad t_{low}, t_{high} \leftarrow 1, |\mathcal{P}|$
6: $\quad$ **while** $t_{high} \geq t_{low}$ **do**
7: $\quad\quad t \leftarrow \lfloor \frac{t_{high} + t_{low}}{2} \rfloor$
8: $\quad\quad \hat{s} \leftarrow \textsc{Inf}(p) - \textsc{Inf}(\text{rank}^{-1}(t)) + \mathcal{N}(0, \sigma^2)$
9: $\quad\quad$ **if** $\hat{s} \geq \xi_{dir}$ **then** $t_{high} \leftarrow \max\{t - 1, 1\}$
10: $\quad\quad$ **else** $t_{low} \leftarrow \min\{t + 1, |\mathcal{P}|\}$
11: $\quad$ **return** t_{high}
12: **for** $u \leftarrow 1, 2, \ldots, k$ **do**
13: $\quad \rho, \beta \leftarrow \rho_{Rank}/k, (\gamma + 1)/2$
14: $\quad \mathcal{I}_u \leftarrow (\textsc{RankBound}(p_u, 0.1\rho, \beta, -1), \textsc{RankBound}(p_u, 0.9\rho, \beta, +1))$
15: **return** $\mathcal{I}_1, \mathcal{I}_2, \ldots, \mathcal{I}_k$

C ALGORITHM VARIANTS

C.1 General User Question

In this section, we introduce a general form of user question through weighted sum, such that more groups can be involved in the question and the comparison between groups can be more flexible. This covers the cases of the original questions, since a single group difference can also be treated as a weighted sum between two groups. We also discuss how the explanation framework should be adapted to this general form. Finally, we give a use case for privately explaining a general user question.

Definition C.1 (General User Question). Given a database D an aggregate query q, a DP mechanism $\mathcal{M}$, and noisy group aggregation releases $\hat{o}_{i_1}, \hat{o}_{i_2}, \ldots, \hat{o}_{i_m}$ of the groups $\alpha_{i_1}, \alpha_{i_2}, \ldots, \alpha_{i_m}$ from the query q, a general user question Q is represented by m weights and a constant c: $(w_{i_1}, w_{i_2}, \ldots, w_{i_m}, c)$. Intuitively, the question is interpreted as "Why $\sum_{j=i_1}^{i_m} w_j \hat{o}_j \geq c$".

Definition C.1 allows more interesting questions, such as "Why the total salary of group A and B is larger than the total salary of group C and D?" or "Why the average salary of group A is 10 times larger than the one of group B?". Next we illustrate how the algorithms for each problem related to our framework should be adapted in the case of general user question.

Private Confidence Interval of Question. Given a general user question $(w_{i_1}, w_{i_2}, \ldots, w_{i_m}, c)$, we discuss how to derive the confidence interval of $\sum_{j=i_1}^{i_m} w_j o_j - c$. Comparing to the case of a simple user question $(\alpha_i, >, \alpha_j)$, where the target of confidence interval is $o_i - o_j$, here we have a weighted sum of multiple group results. Therefore, when *agg* is *CNT* or *SUM*, the noisy weighted sum follows the Gaussian distribution with scale $\sqrt{\sum_{j=i_1}^{i_m} w_j^2}\sigma$, where σ is the noise scale used in query answering. When *agg* is *AVG*, the noisy weighted sum can also be viewed as a combination of multiple Gaussian variables. In conclusion, we consider the adaptaions as follows:

(1) For $agg = CNT$ or $agg = SUM$, update the margin $\sqrt{2}(\sqrt{2}\sigma)\,\mathrm{erf}^{-1}(\gamma)$ as $\sqrt{2}(\sqrt{\sum_{j=i_1}^{i_m} w_j^2}\sigma)\,\mathrm{erf}^{-1}(\gamma)$.
(2) For $agg = AVG$, update the sub confidence level β to be $(\gamma - 1)/(2m) + 1$, and the image of sub confidence intervals to be $\sum_{j=i_1}^{i_m} \mathcal{I}_j^S/\mathcal{I}_j^C - c$.

Private Top-k Explanation Predicates.

Since the user question has a new form, the influence function and its corresponding score function should also be adapted. We consider their natural extensions as follows:

Definition C.2 (General Influence Function). Given a database D and a general user question $Q = (w_{i_1}, w_{i_2}, \ldots, w_{i_m}, c)$ with respect to the query `SELECT` A_{gb}`,` `agg(`A_{agg}`)` `FROM R WHERE` ϕ `GROUP BY` A_{gb}, the influence of an explanation predicate p is defined follows:

$$\textsc{Inf}(p; Q, D) = \left(\sum_{j=i_1}^{i_m} w_j q(g_j(D)) - \sum_{j=i_1}^{i_m} w_j q(g_j(\neg p(D)))\right) \times \begin{cases} \dfrac{\min_{t\in\{i_1,i_2,\ldots,i_m\}} |g_t(\neg p(D))|}{\max_{t\in\{i_1,i_2,\ldots,i_m\}} |g_t(D)|+1} & , agg \in \{COUNT, SUM\} \\ \min_{t\in\{i_1,i_2,\ldots,i_m\}} |g_t(\neg p(D))| & , agg = AVG \end{cases}$$

We can plug-in the new influence function into algorithm 2 to find the noisy top-k explanation predicates. The corresponding sensitivity of the new influence function is given as follows:

Theorem C.3. *[General Influence Function Sensitivity] Given an explanation predicate p and a general user question $Q = (w_{i_1}, w_{i_2}, \ldots, w_{i_m}, c)$ with respect to a group-by query with aggregation agg, the following holds:*

(1) If $agg = CNT$, the sensitivity of $\textsc{Inf}(p; Q, D)$ is $2\sum_{j=i_1}^{i_m} |w_j|$.
(2) If $agg = SUM$, the sensitivity of $\textsc{Inf}(p; Q, D)$ is $2\sum_{j=i_1}^{i_m} |w_j| A_{agg}^{max}$.
(3) If $agg = AVG$, the sensitivity of $\textsc{Inf}(p; Q, D)$ is $8\sum_{j=i_1}^{i_m} |w_j| A_{agg}^{max}$.

Proof. It is a weighted version of Proposition 4.4. □

We also allow explanation predicates to include disjunction and allow the framework to specify a specific set of explanation predicates by enumeration.

Private Confidence Interval of Influence. We can plug-in the new influence function and their sensitivities into the original algorithm to find the confidence interval of influence.

Private Confidence Interval of Rank. We can plug-in the new influence function and their sensitivities into algorithm 4 to find the confidence interval of rank.

Use Case: Taxi-Imbalance. We consider the New York City taxi trips dataset [2] in January and February, 2019, as a use case. We preprocessed the dataset such that it includes 4 columns: `PU_Zone`, `PU_Borough`, `DO_Zone`, `DO_Borough`. In this case we analyze the traffic volume between boroughs. With privacy budget $\rho_q = 0.1$, the framework answers the user query as "SELECT PU_Borough, DO_Borough, CNT(*) FROM R GROUP BY PU_Borough, DO_Borough". There are in total 49 groups, and among the query answers we have *(Brooklyn, Queens): 11,431* and *(Queens, Brooklyn): 121,934*. User then asks "Why Queens to Brooklyn has more than 10 times the number of trips from Brooklyn to Queens?" This corresponds to the question "why $q_1 - 10q_2 \geq 0$", or in the form of weights $(1, -10, 0)$. The confidence interval of the question is $(7580, 7668)$, which validates the question. To explain the question, we consider a predicate space of the form "PU_Zone = <zone> ∨ DO_Zone = <zone>" with in total 127 different zones. With $\rho_{Topk} = 0.025$, $\rho_{Influ} = 0.025$, and $\rho_{Rank} = 0.95$, we have the explanation table as shown in fig. 11. The relative influence is relative to the noisy difference $\hat{o}_1 - 10\hat{o}_2 = 7624$. From this table, we can find that two airports, JFK and LaGuardia airports that are located in Queens, are the major reasons for why there are more traffic volume from Queens to Brooklyn since there are more incoming taxi traffic to the airports instead of outgoing taxi traffic.

C.2 Finding Top-k by arbitrary influence function

In the noisy binary search of algorithm 4, we use the difference between $\textsc{Inf}(p; D)$ and $\textsc{Inf}(\mathrm{rank}^{-1}(t; D, \mathcal{P}, \textsc{Inf}); D)$ as an indicator for each branch. The utility of this algorithm depends on the global sensitivity of the influence function. When we extend the

explanation predicate	Rel Influ 95%-CI		Rank 95%-CI	
	L	U	L	U
zone = "JFK Airport"	55.21%	72.18%	1	1
zone = "LaGuardia Airport"	28.75%	45.72%	1	3
zone = "Bay Ridge"	-6.64%	23.60%	3	127
zone = "Queensboro Hill"	-10.75%	6.22%	3	127
zone = "Flushing"	-12.52%	4.25%	3	127

Figure 11: Top-5 explanations for Taxi-Imbalance.

entire framework to support more queries and questions, the influence function can be more complex and sensitive. For example, given a question such as why $q_1(D)/q_2(D)$ is higher than expected, for some query q_1 and q_2 in the first phase, one influence function could be $\textsc{Inf}(p;D) = (1 - |p(D)|/|D|)(q_1(D)/q_2(D) - q_1(\neg p(D))/q_2(\neg p(D)))$. In this case, one can always find p and D and D' such that the absolute difference between $\textsc{Inf}(p;D)$ and $\textsc{Inf}(p;D')$ is arbitrary high. A typical work around is to bound the ranges of basic queries; however, it introduces bias and may destroy the ranking order. Moreover, the bound needs to be chosen without looking the data, which makes it even more impossible.

On the other hand, the difference between $\textsc{Inf}(p;D)$ and $\textsc{Inf}(\text{rank}^{-1}(t;D,\mathcal{P},\textsc{Inf});D)$ is not the only choice of branch indicator. Denote S as a function of the form $S(p,t;D,\mathcal{P},\textsc{Inf})$. In general, if this function satisfies three properties as listed in the theorem below, which are also the only properties that the proof of Theorem 4.10 requires, using this function as the branch indicator in algorithm 4 still allows this algorithm to satisfy ρ-zCDP and the guarantee of confidence interval of rank.

Theorem C.4. *Substituting $\textit{Inf}(p) - \textit{Inf}(\text{rank}^{-1}(t))$ by $S(p,t;D,\mathcal{P},\textit{Inf})$ and $2\Delta_{\textit{Inf}}$ by Δ_S for algorithm 4, the new algorithm satisfies ρ-zCDP and outputs correct confidence intervals of rank if the following holds for S and Δ_S:*

- *Center Zero.* $S(p, \text{rank}(p;D,\mathcal{P},\textit{Inf});D,\mathcal{P},\textit{Inf}) = 0$
- *Non-Decreasing. For any $i < j$,* $S(p,i;D,\mathcal{P},\textit{Inf}) \le S(p,j;D,\mathcal{P},\textit{Inf})$.
- *Stable. For any two neighboring datasets $D \approx D'$,* $|S(p,t;D,\mathcal{P},\textit{Inf}) - S(p,t;D',\mathcal{P},\textit{Inf})| \le \Delta_S$.

A natural choice of S is to define $S(p,t;D,\mathcal{P},I) = \textsc{Inf}(p;D) - \textsc{Inf}(\text{rank}^{-1}(t;D,\mathcal{P},I);D)$, the difference between the influence of p and the t-th largest influence. With the "Center Zero" and "Non-Decreasing" properties, the indicator function S can tell that a number t is a rank bound of $\text{rank}(p;D,\mathcal{P},\textsc{Inf})$ if $S(p,t;D,\mathcal{P},\textsc{Inf}) > 0$. If i and j are both rank bound of $\text{rank}(p;D,\mathcal{P},\textsc{Inf})$ and i is closer to the target rank than j, $S(p,i;D,\mathcal{P},\textsc{Inf})$ is also closer to 0 than $S(p,j;D,\mathcal{P},\textsc{Inf})$. However, for the natural choice of S, sensitivity $\Delta_S = 2\Delta_{\textsc{Inf}}$ and $\Delta_{\textsc{Inf}}$ could be unbounded for some Inf, which results in poor utility. Instead, we can define S in a way such that it still reflects the difference between the influence of p and the t-th largest influence, but has low sensitivity.

Inspired by inverse sensitivity and other techniques that share the same spirit [9, 10, 28, 91], we present a stable branch indicator function $S(p,t;D,\mathcal{P},\textsc{Inf})$ such that it is approximately the least number of tuples that need to be changed to move the rank of p beyond t. Specially, when $t \le \text{rank}(p;D,\mathcal{P},\textsc{Inf})$, $S(p,t;D,\mathcal{P},\textsc{Inf}) = 0$.

Denote $D \triangle D'$ as the symmetric difference between two datasets D and D'. Denote influence lower bound $ILB(p,d;D,\textsc{Inf}) = \inf\{\textsc{Inf}(p;D') \mid |D'\triangle D| \le d\}$ the least influence of p and influence upper bound $IUB(p,d;D,\textsc{Inf}) = \sup\{\textsc{Inf}(p;D') \mid |D'\triangle D| \le d\}$ the largest influence of p within distance d to D. Given two predicates p and $\tilde{p}$, if $IUB(\tilde{p},d;D,\textsc{Inf}) < ILB(p,d;D,\textsc{Inf})$, it indicates that there is no dataset D' within distance d to D such that the influence of $\tilde{p}$ is higher than or equal to the one of p.

Denote the complementary size of such predicate $\tilde{P}$ in $\mathcal{P}$ as $B(p,d;D,\mathcal{P},\textsc{Inf}) = |\mathcal{P}| - |\{\tilde{p} \in \mathcal{P} \mid IUB(\tilde{p},d;D,\textsc{Inf}) < ILB(p,d;D,\textsc{Inf})\}|$. This gives a rank bound of $\text{rank}(p;D',\mathcal{P},\textsc{Inf})$ for any dataset D' such that $|D\Delta D'| \le d$.

Example C.5. Suppose $\mathcal{P}$ has 5 predicates p_1, p_2, p_3, p_4 and p_5. Now we show $B(p_3,2)$. Suppose at distance 2, $IUB(p_1,2) = 1$, $IUB(p_2,2) = 3$, $IUB(p_3,2) = 6$, $IUB(p_4,2) = 7$, $IUB(p_5,2) = 10$, and $ILB(p_3,2) = 4$. In this case, $B(p_3,2) = 5 - 2 = 3$ since predicate p_1 and p_2 have lower IUB than the ILB of p_3. This indicates, by adding or removing 2 tuples from D, the rank of p_3 cannot be beyond 3.

Lemma C.6. *Given a predicate p, a dataset D, a set of predicates $\mathcal{P}$ and an influence function Inf, for any dataset D' such that $|D\Delta D'| \le 1$ and any distance d, we have:*

$$B(p,d;D,\mathcal{P},\textit{Inf}) \le B(p,d+1;D',\mathcal{P},\textit{Inf}) \tag{23}$$

Proof. Denote $\mathcal{D}_1 = \{D'' \mid |D''\Delta D| = d\}$ and $\mathcal{D}_2 = \{D'' \mid |D''\Delta D'| = d+1\}$. Notice that $B(p,d;D,\mathcal{P},\textsc{Inf})$ (or $B(p,d+1;D',\mathcal{P},\textsc{Inf})$) is counting the complementary size of predicate $\tilde{p}$ in $\mathcal{P}$ such that no dataset D'' in $\mathcal{D}_1$ (or $\mathcal{D}_2$) satisfies $\textsc{Inf}(\tilde{p};D'') \ge \textsc{Inf}(p;D'')$. Since $|D\Delta D'| \le 1$, we have $\mathcal{D}_1 \subseteq \mathcal{D}_2$, therefore $B(p,d;D,\mathcal{P},\textsc{Inf}) \le B(p,d+1;D',\mathcal{P},\textsc{Inf})$. □

If IUB is a loose influence upper bound and ILB is a loose influence lower bound, the lemma above still holds. We show an example of the function B on two neighboring datasets as follows in Table 2.

Table 2: Example of B

d	0	1	2	3	4
$B(p,d;D)$	2	2	4	6	10
$B(p,d;D')$	2	3	5	7	8

Definition C.7. Given a predicate p, a dataset D, a set of predicates $\mathcal{P}$ and an influence function Inf, ω is a stable branch indicator function as

$$\omega(p,t;D,\mathcal{P},\textsc{Inf}) = \min\{d \ge 0 \mid B(p,d;D,\mathcal{P},\textsc{Inf}) \ge t\}$$

Below, we show an example of a stable branch indicator in Table 3.

Table 3: Example of ω

t	1	2	3	4	5	6	7	…
$\omega(p,t)$	0	0	2	2	3	3	10	…

Theorem C.8. *Given a predicate p, a dataset D, a set of predicates $\mathcal{P}$ and an influence function Inf, ω, the three conditions of theorem C.4 is satisfied if function $S = \omega$ and sensitivity $\Delta_S = 1$.*

Proof.

Center Zero. $B(p,0;D,\mathcal{P},I) = \text{rank}(p;D,\mathcal{P},I)$.

Non-Decreasing. Since B is non-decreasing in terms of d given $D, \mathcal{P}, I$, $\omega_t(D)$ is also non-decreasing.

Stable. Drop $\mathcal{P}$, Inf for simplicity. Suppose t is fixed. Denote $d^* = \omega(p, t; D)$. By definition, we have $B(p, d^*; D) \geq t$. For any neighboring dataset $D' \sim D$, since $B(p, d^*; D) \leq B(p, d^* + 1; D')$, it indicates $B(p, d^* + 1; D') \geq t$ and thus we have $\omega(p, t; D') \leq d^* + 1$.

When $d^* < 2$, which means $\omega(p, t; D) < 2$, it is impossible to have $\omega(p, t; D') - \omega(p, t; D) < -1$ since $\omega(p, t; D') \geq 0$ is always true. When $d^* \geq 1$, we show that it is impossible to have $B(p, d^* - 2; D') \geq t$. If $B(p, d^* - 2; D') \geq t$, we have $B(p, d^* - 1; D) \geq B(p, d^* - 2; D') \geq t$, which indicates $\omega(p, t; D) \leq d^* - 1$ and leads to a contradiction. Therefore, we have $B(p, d^* - 2; D') < t$, which indicates the impossibility of $\omega(p, t; D') \leq d^* - 2$. Therefore, we have $\omega(p, t; D') \geq d^* - 1$.

Since $d^* - 1 \leq \omega(p, t; D') \leq d^* + 1$, we have $|\omega(p, t; D) - \omega(p, t; D')| \leq 1$. □

The branch indicator function ω finds the minimum d such that $B(p, d; D, \mathcal{P}, I) \geq t$. If we add a constraint $d \geq C$ with some constant C, the theorem above still holds.

C.3 Large Domain Private Top-k Selection

Algorithm 5 gives a practical version for find top-k elements given an score function from a large domain. It assumes that the domain $\mathcal{P}$ is partitioned into an active domain $\mathcal{P}_{act}$ and an idle domain $\mathcal{P}_{idle}$, such that the elements in the idle domain all have the same score C. We assume a random draw from the idle domain $\mathcal{P}_{idle}$ could be done in $O(1)$, so the runtime of the algorithm only depends on the size of the active domain $\mathcal{P}_{act}$ as $O(k|\mathcal{P}_{act}|)$. This algorithm satisfies $k\epsilon^2/8$-zCDP or $k\epsilon$-DP.

Algorithm 5 Report Noisy Top-k Elements from a Large Domain

Require: A private dataset D, an active domain $\mathcal{P}_{act}$ of predicate class $\mathcal{P}$, an idle domain $\mathcal{P}_{idle}$ of predicate class $\mathcal{P}$, a score function u, global sensitivity of u as Δ_u, a constant C as the score for any element from the idle domain $\mathcal{P}_{idle}$, and a privacy parameter ϵ.

Ensure: Top-k predicates ordered by scores.

Compute $u(p, D)$ for every $p \in \mathcal{P}_{act}$ without releasing the results.
for $i \leftarrow 1 \dots k$ **do**
  $s \leftarrow -\infty$
  for $p \leftarrow$ iterate the space of $\mathcal{P}_{act}$ **do**
    $s' \leftarrow u(p, D) + Gumbel(2\Delta_u/\epsilon)$
    if $s' > s$ **then**
      $p_{r_i} \leftarrow p$
      $s \leftarrow s'$
  $s' \leftarrow \frac{2\Delta_u}{\epsilon} \ln(|\mathcal{P}_{idle}) + C + Gumbel(2\Delta_u/\epsilon)$
  if $s' > s$ **then**
    $\hat{p}_i \leftarrow$ a random draw from $\mathcal{P}_{idle}$
  if $\hat{p}_i \in \mathcal{P}_{act}$ **then**
    $\mathcal{P}_{act} \leftarrow \mathcal{P}_{act} \setminus \{\hat{p}_i\}$
  else
    $\mathcal{P}_{idle} \leftarrow \mathcal{P}_{idle} \setminus \{\hat{p}_i\}$
return $(\hat{p}_1, \hat{p}_2, \dots, \hat{p}_k)$.

C.4 Computing Confidence Interval of General Arithmetic Combinations

Formally, a query q can be expressed as an arithmetic combination of queries if it can be expressed as $q(D) = f(q_1(D), q_2(D), \dots, q_\ell(D))$ where function f includes the operators in $\{+, -, *, /, \exp, \log\}$ and for each sub-query q_i, a noisy answer $\hat{s}_i = N(q_i(D), \sigma_i^2)$ is released under ρ-zCDP [14], where $\sigma_i = \Delta_{q_i}/\sqrt{2\rho/\ell}$ and Δ_{q_i} is the sensitivity for sub-query q_i. The rest of this sub section discusses how to derive the confidence interval for $q(D)$ based on the noisy releases $\hat{s}_1, \hat{s}_2, \dots, \hat{s}_\ell$ and function f.

Given the noisy releases $\hat{s}_1, \hat{s}_2, \dots, \hat{s}_\ell$ through the Gaussian mechanism, the confidence intervals of $q_1(D)$, $q_2(D)$, $\dots$, $q_\ell(D)$ can be derived by Gaussian confidence interval , and there is a clear connection between these queries to $q(D)$ through function f. Therefore, we can compute the image of these confidence intervals through the function f, which is also a valid confidence interval for $q_f(D)$. Given a function $f : X \to Y$, denote by $f[A]$ the image of f under $A \subseteq X$ i.e. $f[A] = \{f(a) : a \in A\}$.

Theorem C.9. *Given a database D and a query q that can be expressed as $q_f(D) = f(q_1(D), q_2(D), \dots, q_\ell(D))$, where f includes the operators in $\{+, -, *, /, \exp, \log\}$, and confidence intervals at confidence level β for $q_1(D), q_2(D), \dots, q_\ell(D)$ as $\mathcal{I}_1, \mathcal{I}_2, \dots, \mathcal{I}_\ell$. Let $\mathcal{I} = f[\mathcal{I}_1 \times \mathcal{I}_2 \times \dots \times \mathcal{I}_\ell]$ be the image of $\mathcal{I}_1 \times \mathcal{I}_2 \times \dots \times \mathcal{I}_\ell$ under f, i.e., the set of numbers composed of each mapping of f for a combination of values from $\mathcal{I}_1 \times \mathcal{I}_2 \times \dots \times \mathcal{I}_\ell$. Also assume that f is defined for each vector in $\mathcal{I}_1 \times \mathcal{I}_2 \times \dots \times \mathcal{I}_\ell$. Let $\mathcal{I}^L = \inf I$ and $\mathcal{I}^U = \sup I$. We have*

$$Pr[\mathcal{I}^L \leq q_f(D) \leq \mathcal{I}^U] \geq \ell(\beta - 1) + 1$$

Proof. Since the event $\bigwedge_{i=1}^{\ell} q_i(D) \in \mathcal{I}_i$ implies $q_f(D) \in \mathcal{I}$ due to $\mathcal{I}$ is the image of $\mathcal{I}_1 \times \mathcal{I}_2 \times \dots \times \mathcal{I}_\ell$ under f, we have $Pr[q_f(D) \in \mathcal{I}] \geq Pr[\bigwedge_{i=1}^{\ell}(q_i(D) \in \mathcal{I}_i)]$. Secondly, by Lemma A.8 and by definition about $Pr[q_i(D) \in \mathcal{I}_i] \geq \beta$ for $\forall i$, we have $Pr[\bigwedge_{i=1}^{\ell}(q_i(D) \in \mathcal{I}_i)] \geq \sum_{i=1}^{\ell} Pr[q_i(D) \in \mathcal{I}_i] - (\ell - 1) \geq \ell(\beta - 1) + 1$. Thirdly, since $\mathcal{I}^L = \inf I$ and $\mathcal{I}^U = \sup I$, we have $I \subseteq [\mathcal{I}^L, \mathcal{I}^U]$, and therefore $Pr[\mathcal{I}^L \leq q_f(D) \leq \mathcal{I}^U] \geq Pr[q_f(D) \in \mathcal{I}]$. Together, we have $Pr[\mathcal{I}^L \leq q_f(D) \leq \mathcal{I}^U] \geq Pr[q_f(D) \in \mathcal{I}] \geq Pr[\bigwedge_{i=1}^{\ell}(q_i(D) \in \mathcal{I}_i)] \geq \ell(\beta - 1) + 1$.

□

Although it might not be obvious to find the analytical form of the image, we can use numerical methods to find the approximations of the supremum and infimum of the image. The width of such a interval is not determined.

Given a query q decomposed by function f and the noisy answers, algorithm 6 summarizes the approach for deriving the confidence interval for $q(D)$. We first derive the confidence interval of each sub-query (line 2 to 3) with confidence level $\beta = 1 - \frac{1-\gamma}{\ell}$ (line 1) and finally compute the confidence interval for $q(D)$ (line 4 and 5).

Algorithm 6 Image-based Confidence Interval

Require: A query q such that $q(D) = f(q_1(D), q_2(D), \ldots, q_\ell(D))$, noisy answers $\hat{s}_1, \hat{s}_2, \ldots, \hat{s}_\ell$ of queries $q_1, q_2, \ldots, q_\ell$ using Gaussian mechanisms with scales σ_1, σ_2, $\ldots$, σ_ℓ, confidence level γ.

$\beta \leftarrow 1 - (1-\gamma)/\ell$
for $i \in 1 \ldots \ell$ **do**
    $\mathcal{I}_i = (\hat{s}_i - \sigma_i\sqrt{2}\,\mathrm{erf}^{-1}(\beta), \hat{s}_i + \sigma_i\sqrt{2}\,\mathrm{erf}^{-1}(\beta))$
$\mathcal{I}^L \leftarrow \inf_{x_i \in \mathcal{I}_i \forall x_i} f(x_1, x_2, \ldots, x_\ell)$
$\mathcal{I}^U \leftarrow \sup_{x_i \in \mathcal{I}_i \forall x_i} f(x_1, x_2, \ldots, x_\ell)$
$\mathcal{I}_q = (\mathcal{I}^L, \mathcal{I}^U)$
return $\mathcal{I}_q$.

Algorithm 7 Bootstrap Confidence Interval

Require: A query q such that $q(D) = f(q_1(D), q_2(D), \ldots, q_\ell(D))$, noisy answers $\hat{o}_1, \hat{o}_2, \ldots, \hat{o}_\ell$ of queries $q_1, q_2, \ldots, q_\ell$ using Gaussian mechanisms with scales σ_1, σ_2, $\ldots$, σ_ℓ, confidence level γ, and a bootstrap step size B.

Ensure: A confidence interval for $q_f(D)$ at confidence level γ.

for $b \leftarrow 1 \ldots B$ **do**
    for $i \leftarrow 1 \ldots \ell$ **do**
        $o_i^* \leftarrow \hat{o}_i + N(0, \sigma_i^2)$
    $\theta_b^* \leftarrow f(o_1^*, o_2^*, \ldots, o_\ell^*)$
$\hat{\theta} \leftarrow f(\hat{o}_1, \hat{o}_2, \ldots, \hat{o}_\ell)$
$z_0 = \Phi^{-1}(\frac{1}{B}\sum_{b=1}^{B} \mathbb{1}_{\theta_b^* < \hat{\theta}})$
$\mathcal{I}^L \leftarrow \min\{s | \frac{1}{B}\sum_{b=1}^{B} \mathbb{1}_{\theta_b^* < s} \geq \Phi(2z_0 + \Phi^{-1}(\frac{1-\gamma}{2})\}$
$\mathcal{I}^U \leftarrow \max\{s | \frac{1}{B}\sum_{b=1}^{B} \mathbb{1}_{\theta_b^* < s} \leq \Phi(2z_0 + \Phi^{-1}(\frac{1+\gamma}{2})\}$
$\mathcal{I}_q = (\mathcal{I}^L, \mathcal{I}^U)$
return $\mathcal{I}_q$.

C.5 Bootstrap Confidence Interval

Bootstrap is an old yet powerful technique started by Bradley Efron in late 70's [45] which can be used for computing the confidence interval of an unknown statistic. Although it is not an exact confidence interval, it enjoys a theoretical guarantee on the correctness of the approximation. Traditional bootstrap assumes there are multiple samples that is samples from a unknown distribution with the parameters of interest. Here the parameters of interest is $q_1(D)$ to $q_\ell(D)$, the distribution is a multivariate Gaussian distribution, and we only observe one sample from it. Therefore, we apply the method introduced from [46] to construct an confidence interval for $q(D)$, which consider the similar problem of finding the confidence interval of $q(D)$ with one observation of noisy $q_1(D)$ to $q_\ell(D)$.

In section 5 of [46], it describes a parametric bootstrap. The main idea is to assume the observation is from a parametric distribution, use the observation to infer the parameters using maximum likelihood estimate, resample from the estimated distribution, and use bias-corrected percentile method to construct a confidence interval. algorithm 7 illustrates an application of parametric bootstrap confidence interval for $q(D)$.

The most related work to this private confidence interval problem that are also based on bootstrap are [24, 49], which construct the CI that encodes the randomness from both sampling and noise, while we only consider the randomness from noise. Traditional CI is closely related to sampling, which assumes some population parameters and data is sampled according to those parameters, therefore the population parameter can be inferred from the sampled data. However, for differential privacy, the setting is totally different. Data is considered as fixed and the statistics of interest is only based on the fixed data. Therefore, there is no randomness in sampling a dataset.

[14] Although user doesn't see the intermediate differentially private query result, we assume the framework stores them.

D SUPPLEMENTARY EXPERIMENT

D.1 Another example of Figure 1d

See Figure 12.

Answer-Phase-3:

explanation predicate	Rel Influ 95%-CI		Rank 95%-CI	
	L	U	L	U
`education = "Bachelors"`	4.51%	11.38%	1	5
`occupation = "Exec-managerial"`	3.04%	9.91%	1	8
`age = "(40, 50]"`	1.98%	8.85%	1	14
`relationship = "Own-child"`	-1.53%	5.34%	1	51
`workclass = "Self-emp-inc"`	-2.34%	4.53%	1	87

Figure 12: Another example (in a random run) of Figure 1d for Phase-3 of DPXPLAIN.

D.2 Confidence Level

Figure 13 shows the relationship between the average interval width of the confidence interval of relative influence and the confidence level.

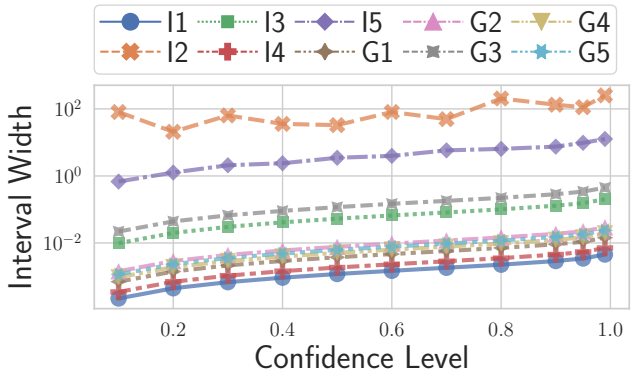


Figure 13: The width of confidence interval of relative influence versus the confidence level.

Figure 14 shows the relationship between the average interval width of the confidence interval of rank and the confidence level.

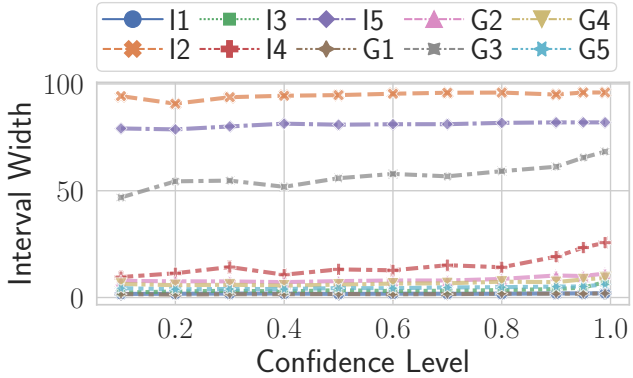


Figure 14: The width of confidence interval of rank versus the confidence level.

D.3 Full Ranking

To further understand the performance of the top-k selection in DPXPLAIN, we set k to be the maximum size to have a full ranking of all the explanation predicates and stops DPXPLAIN at the step of top-k selection. We measure the quality of the full ranking by Kendall-Tau [62]. From Figure 15, we find that for question G1 its Kendall-Tau is always above 0.4 for privacy budget of topk $\rho_{Topk} \geq 0.001$, while for question I1 its Kendall-Tau starts to be above 0.4 when $\rho_{Topk} \geq 0.1$. Though the interpretation of Kendall-Tau is not unified, a correlation coefficient above 0.4 indicates a moderate rank association to the true ranking and above 0.7 indicates a strong rank association [4]. However, when we increase the number of conjuncts l from 1 to 2 to 3, the correlation coefficient drops significantly: for I1, it drops from 0.513 to 0.029 to 0.001, and for G1, it drops from 0.947 to 0.466 to 0.060 . This is because increasing the number of conjuncts l will exponentially increase the number of explanation predicates and thus increase the difficulty of a full ranking.

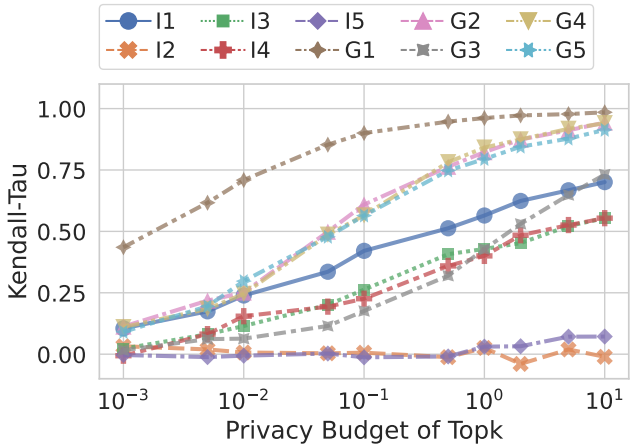


Figure 15: Kendall-Tau of top-k selection by DPXPLAIN.