RESEARCH-ARTICLE

A General Framework for Per-record Differential Privacy

XINGHE CHEN, Nanyang Technological University, Singapore City, Singapore

DAJUN SUN, Hong Kong University of Science and Technology, Hong Kong, Hong Kong

QUANGQING XU, Ant group, Hangzhou, Zhejiang, China

WEI DONG, Nanyang Technological University, Singapore City, Singapore

Proceedings of the ACM on Management of Data, Volume 3, Issue 6 (December 2025)
https://doi.org/10.1145/3769752
EISSN: 2836-6573

A General Framework for Per-record Differential Privacy

XINGHE CHEN, Nanyang Technological University, Singapore
DAJUN SUN, Hong Kong University of Science and Technology, China
QUANQING XU, OceanBase, Ant Group, China
WEI DONG*, Nanyang Technological University, Singapore

Differential Privacy (DP) is a widely adopted standard for privacy-preserving data analysis, but it assumes a uniform privacy budget across all records, limiting its applicability when privacy requirements vary with data values. Per-record Differential Privacy (PrDP) addresses this by defining the privacy budget as a function of each record, offering better alignment with real-world needs. However, the dependency between the privacy budget and the data value introduces challenges in protecting the budget's privacy itself. Existing solutions either handle specific privacy functions or adopt relaxed PrDP definitions. A simple workaround is to use the global minimum of the privacy function, but this severely degrades utility, as the minimum is often set extremely low to account for rare records with high privacy needs. In this work, we propose a general and practical framework that enables any standard DP mechanism to support PrDP, with error depending only on the minimal privacy requirement among records actually present in the dataset. Since directly revealing this minimum may leak information, we introduce a core technique called *privacy-specified domain partitioning*, which ensures accurate estimation without compromising privacy. We also extend our framework to the local DP setting via a novel technique, *privacy-specified query augmentation*. Using our framework, we present the first PrDP solutions for fundamental tasks such as count, sum, and maximum estimation. Experimental results show that our mechanisms achieve high utility and significantly outperform existing Personalized DP (PDP) methods, which can be viewed as a special case of PrDP with relaxed privacy protection.

CCS Concepts: • **Security and privacy** → **Database and storage security**; • **Information systems** → *Data management systems*.

Additional Key Words and Phrases: Differential privacy, SJA query processing

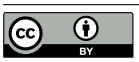
ACM Reference Format:
Xinghe Chen, Dajun Sun, Quanqing Xu, and Wei Dong. 2025. A General Framework for Per-record Differential Privacy. *Proc. ACM Manag. Data* 3, 6 (SIGMOD), Article 287 (December 2025), 27 pages. https://doi.org/10.1145/3769752

1 Introduction

Differential privacy (DP) is a rigorous standard for protecting individual information in private data release. It has been widely adopted in the industry [14, 29, 53] or government census authority [44]. Informally, DP ensures that the presence or absence of a single record in a dataset is ε-indistinguishable from the query results. The parameter ε, known as the privacy budget, controls the strength of the privacy protection—a smaller value indicates stronger protection. Although the standard DP model is widely adopted, it assumes uniform privacy requirements across all

*Wei Dong is the corresponding author.

Authors' Contact Information: Xinghe Chen, Nanyang Technological University, Singapore, Singapore, xinghe001@e.ntu.edu.sg; Dajun Sun, Hong Kong University of Science and Technology, Hong Kong, China, dsunad@connect.ust.hk; Quanqing Xu, OceanBase, Ant Group, Hangzhou, China, xuquanqing.xqq@oceanbase.com; Wei Dong, Nanyang Technological University, Singapore, Singapore, wei_dong@ntu.edu.sg.



ACM 2836-6573/2025/12-ART287
https://doi.org/10.1145/3769752

records. This assumption is undesirable in many cases. For example, when a bank aims to estimate the total deposits of its customers, account records associated with large balances may require stronger privacy protection. In this case, each record can have its own privacy budget, which can be formulated as a function $\mathcal{E}(r)$ of the record r. In the above example, where a larger balance requires higher privacy protection, we can define the privacy function to be inversely proportional to the balance. Such a setting is called per-record DP (PrDP) [2, 49]. Compared with the standard DP model, the PrDP model provides record-dependent privacy protection, thus it is more flexible and can better fit some realistic requirements, like the above example of bank deposit estimation.

Recently, another DP model called personalized DP (PDP) [35, 51] has been introduced to meet personalized privacy needs, where each data record is assigned a fixed, publicly known privacy budget. In contrast, PrDP employs a privacy budget function $\mathcal{E}(\cdot)$ that adapts the budget based on data values. While these models may appear distinct at first glance, we can show that PrDP is a more general notation with stronger privacy guarantees. Specifically, by assigning a user-specified attribute to each record and defining the privacy budget function based on this attribute, any PDP problem can be reduced to a PrDP problem. Moreover, this reduction offers stronger privacy protection, as the user-defined privacy budgets remain private under PrDP, whereas PDP requires them to be publicly disclosed (see Section 3.4 for more details).

Although PrDP is more general than both DP and PDP, it remains challenging to realize in practice. Despite being proposed over a decade ago, no existing work has fully achieved PrDP, even for specific query classes. For example, [2] studies sum estimation under PrDP with a highly restricted privacy function—proportional to the data value—and shows that the Laplace mechanism satisfies PrDP in this special case. Meanwhile, [49] considers more general privacy functions, but under a relaxed version of the PrDP definition. In the following, we identify two key challenges in achieving PrDP.

Task	Basic Counting	Sum Estimation	Distinct Count
ε-DP	$O\left(1/\varepsilon\right)$	$O\left(\log\log U \cdot \mathrm{Max}(D)/\varepsilon\right)$ [24]	$O\left(1/\varepsilon\right)$
PDP	$O\left(\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \cdot 1/\varepsilon_{\min}(D)\right)$ [51]	$O\left(\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \cdot \mathrm{Max}(D)/\varepsilon_{\min}(D)\right)$ [51]	N.A.
PrDP (Ours)	$O\left(\log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \cdot 1/\varepsilon_{\min}(D)\right)$	$O\left((\log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} + \log\log U) \cdot \mathrm{Max}(D)/\varepsilon_{\min}(D)\right)$	$O\left(\log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \cdot 1/\varepsilon_{\min}(D)\right)$
ε-LDP	$O\left(\sqrt{n}/\varepsilon\right)$	$O\left(\sqrt{n} \cdot \log U \cdot \log\log U \cdot \mathrm{Max}(D)/\varepsilon\right)$ [33]	N.A.
PrLDP (Ours)	$O\left(\sqrt{n} \cdot \log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \cdot 1/\varepsilon_{\min}(D)\right)$	$O\left(\sqrt{n} \cdot (\log\log \frac{\hat{\varepsilon}}{\check{\varepsilon}} + \log U \cdot \log\log U) \cdot \mathrm{Max}(D)/\varepsilon_{\min}(D)\right)$	N.A.

Table 1. Comparison of error bounds under different privacy models across tasks.

Challenge 1 (Utility). To achieve PrDP, a naive approach is to use the global minimum privacy budget $\check{\varepsilon}$ for all records, thereby reducing PrDP problems to standard DP problems. This leads to a $\check{\varepsilon}$-dependent error bound. However, as a global lower bound on privacy budgets, $\check{\varepsilon}$ is typically set to a conservatively small value. It corresponds to a hypothetical record with the most stringent privacy requirement—an extreme case rarely observed in practice. In practice, $\check{\varepsilon}$ is also associated with a global maximum value, denoted by U. For example, financial institutions often set the maximum bank account deposit to $U = 10^{12}$, equivalent to 1,000 billion \$. If we use the privacy function discussed earlier, then $\check{\varepsilon} \sim 10^{-12}$. As a result, using such a $\check{\varepsilon}$ leads to significant utility degradation, as most existing DP mechanisms have an error proportional to $1/\varepsilon$. Instead, it is preferable that, for a specific dataset D, the error depends on the minimal privacy budget within the instance, denoted as $\varepsilon_{\min}(D)$. As in the example of bank total deposit estimation, $\varepsilon_{\min}(D)$ depends on the records with the maximum balance in the dataset, which is much smaller than 10^{12}. To address this issue,

existing research assumes a tighter $\check{\varepsilon}$ [31, 47, 48], which is close to $\varepsilon_{\min}(D)$. However, such prior knowledge compromises privacy, as $\varepsilon_{\min}(D)$ is itself correlated with the records.

Challenge 2 (Privacy). To avoid relying on the global minimum privacy budget, a common strategy is the *clipping mechanism* [1, 33]. For example, in deposit estimation, all values above a threshold τ are clipped to τ, and a DP sum estimator is applied to the clipped data using a privacy budget $\mathcal{E}(\tau)$. However, this approach further presents two problems. First, utility highly depends on the choice of τ, which ideally should be dataset-dependent. For example, to achieve an error dependency on $\varepsilon_{\min}(D)$, we can set τ to the maximum deposit in D, $\mathrm{Max}(D)$—a sensitive value that must itself be estimated under PrDP—thus reintroducing the need to select an appropriate privacy budget. Besides choosing an appropriate τ, another problem is that such an idea will break the privacy of the clipped records. This is because the privacy budget is determined by the original values and is unaffected by clipping. Specifically, even if the balance of r is clipped to τ, its privacy budget must still be $\mathcal{E}(r)$ rather than $\mathcal{E}(\tau)$. To address this, Jeremy Seeman et al. [49] propose an alternative that sacrifices privacy guarantees for records exceeding τ. However, this approach relies heavily on an appropriate choice of τ, which requires strong prior knowledge, and fails to fully satisfy the requirements of PrDP.

In addition to these two challenges, PrDP solutions should also be general, i.e., not only supporting arbitrary privacy budgets, but also accommodating a wide range of query types. Above all, this raises a question:

Problem Statement. Does there exist a general framework that enables an arbitrary DP mechanism $\mathcal{M}$ to provide a PrDP guarantee with error dependency on $\varepsilon_{\min}(D)$?

1.1 Our Results

In this paper, we answer the question affirmatively by presenting a novel general framework that enables existing DP mechanisms to solve PrDP problems. To address the issue of the clipping mechanism, we first propose a technique called *privacy-specified domain partitioning*, which partitions records based on their privacy budgets. This method partitions the entire record universe into $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ disjoint privacy-specified domains, where $\hat{\varepsilon}$ denotes the predefined global maximum privacy budget. Within each domain, all records have similar privacy budgets. We then estimate the number of records in D for each domain using the smallest privacy budget assigned to that domain. Based on these estimates, we identify the "heavy" domains and use the smallest privacy budget among them, denoted by ε_τ, to approximate $\varepsilon_{\min}(D)$. Naturally, this idea leads directly to a PrDP counting algorithm by aggregating the counting of "heavy" domains. For other types of queries, we further take the union of all domains with privacy budgets exceeding ε_τ, and construct a dataset containing only the records within this unioned domain. Finally, we invoke an existing DP mechanism on the new dataset with privacy budget ε_τ to compute the final query results. Building on this idea, we can extend any standard DP protocol to satisfy PrDP, with an error that depends on $\varepsilon_{\min}(D)$.

Our main contributions are threefold:

- **Basic counting under PrDP.** Section 5 introduces our approach to the basic counting problem under PrDP. Algorithm 1 achieves an error bound of $O\left(\log\log(\hat{\varepsilon}/\check{\varepsilon})/\varepsilon_{\min}(D)\right)$[1]. Compared to the standard DP error bound of $O(1/\varepsilon)$, our mechanism replaces ε with $\varepsilon_{\min}(D)$, incurring only a doubly logarithmic overhead.
- **General problems under PrDP.** Section 6 presents a general framework for arbitary PrDP problems, built upon our basic counting algorithm. The error bound of this framework is somewhat technical. Roughly speaking, given any standard DP mechanism $\mathcal{M}$ with an

[1]All error bounds stated in the introduction holds with constant probability.

error $\mathrm{Err}_{\mathcal{M}}(D, \varepsilon)$ on D, our framework almost achieves an error $\tilde{O}(\mathrm{Err}_{\mathcal{M}}(D, \varepsilon_{\min}(D)))$ [2] (see Theorem 6.1 for more details).

- **Problems under local-DP setting.** In Section 7, we extend our PrDP framework to the local DP (LDP) setting—speci-fically, per-record local DP (PrLDP)—by introducing a novel technique called *privacy-specified query augmentation*. Our framework achieves results comparable to those in the central DP setting. Given an LDP protocol $\mathcal{M}$ with error $\mathrm{Err}_{\mathcal{M}}(D, \varepsilon)$ on dataset D, our PrLDP framework achieves an error of approximately $\tilde{O}(\mathrm{Err}_{\mathcal{M}}(D, \varepsilon_{\min}(D)))$ (see Theorem 7.4 for details).

For error bounds on specific queries within our general framework, refer to Table 1. To the best of our knowledge, this is the first work to achieve PrDP even for fundamental queries. As a baseline, we compare our results with those from [51], the state-of-the-art (SOTA) PDP mechanisms for counting and sum estimation. This comparison is not entirely fair since, as previously noted, PDP is a special case of PrDP with a relaxed privacy guarantee. Surprisingly, we even improved their PDP error bounds, reducing the polylogarithmic factor to a doubly logarithmic one. Moreover, in Section 8, we demonstrate our framework yields 2× to 165× reduction in error compared to prior PDP methods in practice [3]. Its generalizability is also verified under three different privacy budget functions: inverse, logarithmic, and square root.

2 Related Work

Providing record-specific privacy protection is a highly relevant and important problem, driven by the diverse backgrounds of users and their varying privacy requirements. Such a challenge was explored even before the concept of DP. For instance, Xiao and Tao [54] introduced the concept of personalized privacy within the framework of k-anonymity, which was subsequently extended to other privacy models [32, 50, 58].

Under the context of DP [25, 26], which provides a rigorous definition of privacy and has become the dominant privacy model, several approaches offer record-specific privacy protection. PrDP was introduced by [2] and formalized in [49]. They studied some basic queries under specific privacy functions. However, they do not support arbitrary privacy functions, even for their selected queries. Besides PrDP, Jorgensen *et al.* [35] propose the PDP model, where each user specifies a (public) personalized privacy parameter. As we show in Section 3.4, this can be regarded as a special case of PrDP with relaxed privacy protection. Significant efforts have been made to achieve PDP under various settings [7, 13, 28, 40, 42, 46, 48, 57]. Most recently, Sun *et al.* [51] provided the first formal error bounds for PDP on fundamental queries, including basic counting and sum estimation.

Moving towards the LDP setting, existing studies have primarily focused on PDP [5, 10, 41, 56, 59], while no prior work has addressed PrDP. In contrast, our work is the first to investigate PrDP in the local setting.

3 Preliminary

3.1 Notation

Let $[U] = \{0, 1, \ldots, U\}$. A dataset $D = \{r_1, r_2, \ldots, r_n\}$ contains $n \in \mathbb{N}$ records, where each record $r \in [U]^d$ is a d-dimensional vector. We say that $D' \subseteq D$ if $D' = D$ or if D' can be obtained by deleting some records from D. Let $|\cdot|$ denote the ℓ_1 norm for any vector. As in the bank example provided in the introduction, the dataset D contains the account information of a bank's customers, with each record r represented as

$$r = (v_{\mathrm{Bal}}, v_{\mathrm{PC}}, \ldots).$$

[2] We use $\tilde{O}(\cdot)$ to hide polylogarithmic factors.

[3] Code is available at https://github.com/XChen1998/A-General-Framework-for-Per-record-Differential-Privacy.

where each coordinate represents a different attribute of the customer such as account balance (v_{Bal}) and mailing postcode (v_{PC}). Note that the integer domain is a standard and widely adopted setting in the DP literature [16, 24, 33], as fractional domains can be easily mapped to integers. For example, a bank account balance, typically represented as a decimal number, can be converted to an integer by defining a base unit and applying quantization.

3.2 DP Basics

In this paper, we define two datasets, D and D', as neighbors if they differ by the insertion or deletion of a single record, denoted as $D \sim D'$. We use $D \sim_r D'$ to indicate that D and D' differ specifically on the record r. We define $d(D, D')$ as the distance between D and D', measured by the minimum number of single-record additions or deletions needed to transform D into D'. The standard DP model is defined as follows:

Definition 3.1. (Differential Privacy). For $\varepsilon > 0$, a mechanism $\mathcal{M}$ is ε-DP if for any pair of neighboring datasets $D \sim_r D'$ and any possible output y, the following holds:

$$\Pr[\mathcal{M}(D) = y] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(D') = y].$$

3.2.1 Laplace mechanism. A commonly used one-dimensional DP mechanism is the *Laplace mechanism*. Given any query $Q : [U]^{n\times d} \to \mathbb{R}$, its *global sensitivity* (GS_Q) is defined as

$$\text{GS}_Q = \max_{D, D' : D \sim D'} |Q(D) - Q(D')|.$$

The Laplace mechanism is given by

Lemma 3.2. (Laplace Mechanism). *Given a query $Q : [U]^{n\times d} \to \mathbb{R}$, the Laplace mechanism*

$$\mathcal{M}(D) = Q(D) + \eta$$

satisfies ε-DP, where η is drawn from a Laplace distribution with scale GS_Q/ε, i.e., $\eta \sim Lap\left(GS_Q/\varepsilon\right)$.

The following Laplace tail bound shows that the error of the Laplace mechanism is at most $\text{GS}_Q/\varepsilon \cdot \log(1/\beta)$ with probability at least $1 - \beta$.

Lemma 3.3. (Laplace Tail Bound). *Let $\eta \sim Lap(1)$ be a random variable drawn from the unit Laplace distribution. For any $\beta \in (0, 1)$, the tail probability of η satisfies*

$$\Pr(|\eta| \geq \ln \frac{1}{\beta}) \leq \beta.$$

Therefore, for a basic counting problem under ε-DP, where $\text{GS}_Q = 1$, the Laplace mechanism achieves an error of $O\left(1/\varepsilon\right)$ with probability at least $1 - \beta$. However, for more complicated problems like one-dimensional sum estimation, where $\text{GS}_Q = U$, this mechanism incurs an error of $O(U/\varepsilon)$. This is undesirable, as U is a predefined global parameter that must be set conservatively large to accommodate all possible datasets. For example, in the bank deposit estimation problem, U is set to 10^{12} to handle the largest possible deposit, but such a value rarely appears in most banks.

In contrast, recent studies have achieved an instance-specific error bound of $\tilde{O}\left(\max(D)/\varepsilon\right)$ [17], where $\max(D)$ is the largest value in D and $\max(D) \ll U$ for most datasets. In the bank total deposit problem, $\text{Max}(D) = \max\{v_{\text{Bal}} \mid r \in D\}$ represents the highest balance in D considered in this study, which is significantly smaller than 10^{12}.

3.2.2 Optimality of DP Errors. The Laplace mechanism is known to achieve *worst-case optimal error* [18]. However, worst-case optimality often introduces an unnecessarily large error in most instances. To mitigate this, *down-neighborhood optimality* [30] was introduced as a more meaningful criterion for defining optimality in standard DP error.

Definition 3.4. (Down-neighborhood Optimality) [30]. Let $\mathbb{M}$ be the class of all ε-DP mechanisms. Given any query Q, the ρ-down neighborhood lower bound on dataset D is defined as

$$\mathcal{L}(D,\rho) := \inf_{\mathcal{M}'\in\mathbb{M}} \sup_{\substack{D'\subseteq D,\\ d(D,D')\le\rho}} \inf\left\{\xi : \Pr\left[|\mathcal{M}'(D') - Q(D')| \le \xi\right] \ge \frac{2}{3}\right\},$$

and a mechanism $\mathcal{M}$ is (ρ, c)-down neighborhood optimal if, for any dataset D,

$$\Pr\left[|\mathcal{M}(D) - Q(D)| \le c\cdot\mathcal{L}(D,\,\rho)\right] \ge \frac{2}{3},$$

where c is a constant quantifying the level of optimality.

Roughly speaking, a mechanism $\mathcal{M}$ is (ρ, c)-down neighborhood optimal if, for every D, it performs as well as the optimal mechanism $\mathcal{M}'$ specifically designed for D and its ρ-down neighborhood. Clearly, smaller ρ and c imply stronger optimality. Moreover, $\mathcal{L}(G,\rho)$ is further lower-bounded by the $(\rho-1)$*-downward query difference.*

Theorem 3.5 ([30]). *For any query Q, dataset D, $\varepsilon \le \ln 2$ and $\rho \ge 1$, we have*

$$\mathcal{L}(D,\rho) \ge \frac{\Delta Q^{(\rho-1)}(D)}{2\rho},$$

where

$$\Delta Q^{(\rho-1)}(D) = \max_{D'\subseteq D,\, d(D,D')\le\rho} \left|Q(D') - Q(D)\right|$$

is the $(\rho-1)$-downward query difference.

Therefore, if a DP mechanism achieves an error of $c\cdot\Delta Q^{(\rho-1)}(D)$, then it is $(\rho,\, 2c\rho)$-down neighborhood optimal. It is widely accepted that when $\rho = \tilde{O}(1/\varepsilon)$, such an error is good enough [24, 30]. Moreover, to the best of our knowledge, any existing DP mechanism incurs an error of $\Omega(\Delta Q^{\rho}(D))$ with $\rho = \Omega(1/\varepsilon)$.

3.3 Per-record Differential Privacy

The standard DP model assumes that all records have equal privacy requirements. However, in reality, privacy concerns may vary across records depending on their content, as discussed in the introduction.

To accommodate diverse real-world needs, the per-record DP framework [2, 49] was introduced, allowing each record to have a privacy requirement based on its content. In this framework, a record-dependent privacy budget function $\mathcal{E}(\cdot)$ is provided to define the privacy requirement of each record r.

Definition 3.6. (Record-Dependent Privacy Budget Function). A record-dependent privacy budget function, or simply a privacy budget function, is a mapping $\mathcal{E} : [U]^d \to [\check{\varepsilon}, \hat{\varepsilon}]$, where $\check{\varepsilon}, \hat{\varepsilon} \in \mathbb{R}_+$ are predefined lower and upper bounds on the privacy budgets, respectively.

For the example of estimating a bank's total deposits, a possible privacy budget function is defined as

$$\mathcal{E}(r) = \begin{cases} \alpha/v_{\text{Bal}}, & \text{if } \alpha/\hat{\varepsilon} \le v_{\text{Bal}}, \\ \hat{\varepsilon}, & \text{if } v_{\text{Bal}} > \alpha/\hat{\varepsilon} \end{cases} \tag{1}$$

where α is a constant factor and $\hat{\varepsilon}$ is a predefined upper bound on the privacy budget. Since v_{Bal} has a domain of $[U]$, there is no need to explicitly specify $\check{\varepsilon}$; instead, we use $\check{\varepsilon} = \alpha/U$. This privacy budget function ensures stronger protection for records with larger values.

With the privacy budget function, we can define per-recond DP:

Definition 3.7. (Per-record Privacy (PrDP)). Given a record-dependent privacy budget function $\mathcal{E} : [U]^d \to [\check{\varepsilon}, \hat{\varepsilon}]$, a mechanism $\mathcal{M}$ is $\mathcal{E}$-per-record differentially private, or simply $\mathcal{E}$-PrDP, if for any pair of neighboring datasets $D \sim_r D'$ and for all possible outputs y, the following holds:

$$\Pr[\mathcal{M}(D) = y] \le e^{\mathcal{E}(r)} \cdot \Pr[\mathcal{M}(D') = y].$$

As required by our methodology, we define the concept of the *privacy-specified domain* as follows:

Definition 3.8. (Privacy-Specified Domain). For a given $\mathcal{E} : [U]^d \to \mathbb{R}_+$ and a privacy budget range $[\varepsilon_a, \varepsilon_b]$, the privacy-specified domain is defined as

$$\mathcal{X}_{[\varepsilon_a, \varepsilon_b]} = \{r \in [U]^d \mid \mathcal{E}(r) \in [\varepsilon_a, \varepsilon_b]\}.$$

For example, if the privacy budget function is given by (1) with $\alpha = 10^4$, then $\mathcal{X}_{[0.002, 0.004]}$ includes records with $v_{\text{Bal}} \in [2.5 \times 10^6, 5 \times 10^6]$.

PrDP enjoys the following general properties:

Lemma 3.9. (Post-Processing). *If a mechanism $\mathcal{M}$ is considered $\mathcal{E}$-PrDP, then for any mechanism $\mathcal{M}'$, $\mathcal{M}'(\mathcal{M}(D))$ is still $\mathcal{E}$-PrDP.*

Lemma 3.10. (PrDP Sequential Composition). *Let $\{\mathcal{M}_1, \ldots, \mathcal{M}_k\}$ be a series of $\mathcal{E}_i$-PrDP mechanisms. Then $\mathcal{M}_{[k]}$, defined as*

$$\mathcal{M}_{[k]}(D) \coloneqq (\mathcal{M}_1(D), \mathcal{M}_2(D), \ldots, \mathcal{M}_k(D)),$$

satisfies $\mathcal{E}'$-PrDP, where $\mathcal{E}'(r) = \sum_{i=1}^{k} \mathcal{E}_i(r)$ for all $r \in [U]^d$.

Lemma 3.11. (PrDP Parallel Composition). *Let $\mathcal{R}_1, \ldots, \mathcal{R}_k$ be a set of disjoint subdomains. Let $\{\mathcal{M}_1, \ldots, \mathcal{M}_k\}$ be a series of $\mathcal{E}_i$-PrDP mechanisms. Then their parallel composition*

$$\mathcal{M}(D) \coloneqq (\mathcal{M}_1(D \cap \mathcal{R}_1), \ldots, \mathcal{M}_k(D \cap \mathcal{R}_k))$$

satisfies $\mathcal{E}'$-PrDP, where $\mathcal{E}'(r) = \max\{\mathcal{E}_i(r)\}_{i=1}^{k}$ for all $r \in [U]^d$.

3.4 PrDP and PDP

PDP assumes that each record r_i has a user-defined privacy budget ε_i that is independent of its value. In contrast, PrDP defines the privacy budget of r_i as a function of its value. Furthermore, PDP considers all record values r_i as public information, whereas PrDP only reveals the privacy budget function $\mathcal{E}(\cdot)$ while keeping the privacy budget of each individual record private.

At first glance, both PDP and PrDP provide record-specific privacy protection but follow different models and are thus incomparable. PDP defines the privacy protection level based on user-specified budgets, whereas PrDP determines it as a function of each record's value. In the following example, we show that PDP is, in fact, just a special case of PrDP with a relaxed privacy guarantee.

Recall that each record r_i is a high-dimensional vector. We introduce an additional user-specified attribute, v_{bud}, and define the privacy budget function as $\mathcal{E}(r) = c \cdot v_{\text{bud}}$, where c is a predefined

parameter. This design enables a user-specific privacy budget for each record, achieving the functionality of PDP. Furthermore, in this setting, PrDP maintains the privacy of each record's budget, whereas PDP requires it to be publicly disclosed.

4 Straw-man Framework

We begin by exploring several straightforward approaches to achieve $\mathcal{E}$-PrDP. While these methods ultimately fail to achieve $\mathcal{E}$-PrDP, they highlight key challenges and offer valuable insights for our final solution.

4.1 Utilize the Global Minimal Privacy Budget

The most intuitive way to satisfy $\mathcal{E}$-PrDP is to utilize the global minimal privacy budget $\check{\varepsilon}$, thereby reducing the problem to $\check{\varepsilon}$-DP. In the example of bank deposit estimation, the smallest possible privacy budget is $\check{\varepsilon} = \alpha/U$. The best error for one-dimensional sum estimation under standard DP is $O\left(\log\log U \cdot \mathrm{Max}(D)/\varepsilon\right)$ [24]. Substituting $\check{\varepsilon} = \alpha/U$ yields an error of $\tilde{O}(\mathrm{Max}(D) \cdot U)$. Clearly, using $\check{\varepsilon}$ directly leads to excessively large errors, as demonstrated in the introduction. This occurs because the approach always considers the record with the strongest global protection, even if such records may not appear in most datasets.

Therefore, achieving an error dependency on the instance-specific minimal privacy budget, i.e., $\varepsilon_{\min}(D)$, is much more preferable. In the previous example, using $\varepsilon_{\min}(D)$ as the privacy budget results in an error of $\tilde{O}\left(\mathrm{Max}(D)^2\right)$, which can be significantly smaller since $\mathrm{Max}(D) \ll U$. However, directly utilizing $\varepsilon_{\min}(D)$ violates PrDP, since it depends on $\mathrm{Max}(D)$, a highly sensitive statistic. This motivates our development of alternative solutions.

4.2 Clipping Mechanism in PrDP

A natural idea to reduce the error dependence on the global minimum privacy budget $\check{\varepsilon}$ to an instance-specific budget, $\varepsilon_{\min}(D)$, is the clipping mechanism. Such a technique is widely used in standard DP sum estimation to reduce error dependence from a predefined domain upper bound U to $\mathrm{Max}(D)$. The high-level idea is to first estimate a threshold $\tau \sim \mathrm{Max}(D)$ and then clip values over τ to τ. By adding noise proportional to τ to the result over the clipped data, we can achieve an error depending on $\mathrm{Max}(D)$.

Inspired by the clipping mechanism under standard DP, except for finding a value threshold τ, our idea is to also privately find a suitable estimation of $\varepsilon_{\min}(D)$, denoted as ε_τ. Then, we clip the records with a privacy requirement stronger than ε_τ and run an ε_τ-DP mechanism on the clipped dataset. In the example of bank deposit estimation, since the privacy budget is inversely proportional to v_{Bal}, estimating ε_τ is equivalent to estimating $\mathrm{Max}(D)$. Thus, we first find a close approximation of $\mathrm{Max}(D)$, denoted as τ, then clip all records beyond this threshold. Finally, we invoke an ε_τ-DP mechanism over the clipped data with $\varepsilon_\tau = \mathcal{E}(\tau)$.

At first glance, this method appears to achieve $\mathcal{E}$-PrDP, as clipping ensures that all records are bounded by τ, with the privacy budgets bounded by ε_τ. However, we will show below that this approach neither satisfies PrDP nor guarantees an error dependency on $\varepsilon_{\min}(D)$.

- **Challenge 1 (Privacy)**: *Invoking a DP mechanism with ε_τ over clipped data is insufficient to guarantee $\mathcal{E}$-PrDP.* By definition, the privacy budget of a record r should be $\mathcal{E}(r)$. Thus, even if we clip its value to τ, we must still adhere to its original privacy budget rather than the privacy budget after clipping. Therefore, ε_τ is insufficient to guarantee privacy.
- **Challenge 2 (Utility)**: *Accurately estimating $\varepsilon_{\min}(D)$ under $\mathcal{E}$-PrDP is challenging.* To estimate $\varepsilon_{\min}(D)$ under $\mathcal{E}$-PrDP, the foremost issue is determining the appropriate privacy budget to use. The most straightforward approach is to use $\check{\varepsilon}$, which leads to an error dependency

on $\check{\varepsilon}$. Alternatively, using ε_τ to estimate itself results in a circular dependency, creating a chicken-and-egg problem.

5 Our First Attempt in PrDP Protocol Design: Basic Counting

In this section, we present our first attempt at designing an $\mathcal{E}$-PrDP protocol by studying the basic counting problem. This study provides key insights into our design and serves as a foundational building block for our general PrDP framework. The standard DP protocol for bit counting achieves an error of $O(1/\varepsilon)$ with the Laplace mechanism. Moving toward $\mathcal{E}$-PrDP, utilizing the global minimum privacy budget $\check{\varepsilon}$ yields an error $O(1/\check{\varepsilon})$. Our proposed protocol improves this by reducing the error to $\tilde{O}(1/\varepsilon_{\min}(D))$.

5.1 Privacy-Specified Domain Partitioning

Fundamentally, the first challenge in applying the clipping mechanism to achieve $\mathcal{E}$-PrDP arises from the fact that records with strong privacy requirements retain their original privacy constraints even after clipping. Therefore, a natural idea is to directly exclude those records with small privacy budgets.

This can be achieved through privacy-specified domain partitioning, where the entire domain is divided into sub-domains such that values r within each sub-domain have similar privacy budgets $\mathcal{E}(r)$. When we perform count estimation, given a threshold ε_τ, we only add up the counting results for those sub-domains with privacy budgets greater than or equal to ε_τ. This approach ensures that records with small privacy budgets (i.e., less than ε_τ) do not incur any privacy leakage, thereby addressing the first challenge associated with the clipping mechanism.

To address the second challenge—accurately estimating $\varepsilon_{\min}(D)$ under PrDP—we extend our domain partitioning approach to identify a threshold ε_τ. The goal is to ensure that the count derived from selected sub-domains (i.e., those with privacy budgets greater than ε_τ) can serve as a reliable estimate for the final counting result. Since the error consists of two components—the number of records excluded and the noise introduced in the selected sub-domains—achieving an error of $\widetilde{Q}(1/\varepsilon_{\min}(D))$ requires two objectives:

(a) At most $\tilde{O}(1/\varepsilon_{\min}(D))$ records are excluded.
(b) The total noise introduced in the counting results of the selected sub-domains stays within $\tilde{O}(1/\varepsilon_{\min}(D))$.

With these ideas, we now proceed with the design of our protocol. For domain partitioning, we begin by segmenting the privacy budget range into $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ disjoint sub-intervals, where all sub-intervals are left-open and right-closed except for the first one:

$$[2^0 \cdot \check{\varepsilon},\ 2^1 \cdot \check{\varepsilon}],\ (2^1 \cdot \check{\varepsilon},\ 2^2 \cdot \check{\varepsilon}],\ \ldots,\ (2^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil - 1} \cdot \check{\varepsilon},\ \hat{\varepsilon}].$$

By leveraging the concept of a privacy-specified domain, as introduced in Definition 3.8, we can partition the input domain into $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ disjoint sub-domains:

$$\mathcal{X}_{[2^0 \cdot \check{\varepsilon},\, 2^1 \cdot \check{\varepsilon}]},\ \mathcal{X}_{(2^1 \cdot \check{\varepsilon},\, 2^2 \cdot \check{\varepsilon}]},\ \ldots,\ \mathcal{X}_{(2^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil - 1} \cdot \check{\varepsilon},\, \hat{\varepsilon}]}.$$

For simplicity, we denote the i-th domain $\mathcal{X}_{(2^{i-1} \cdot \check{\varepsilon},\, 2^i \cdot \check{\varepsilon}]}$ as $\mathcal{X}_i$. The primary advantage of this partitioning is that for each sub-domain $\mathcal{X}_i$, we can use $2^{i-1} \cdot \check{\varepsilon}$ for all records within that domain. Such a design upgrades the privacy protection for each record by at most a factor of two, thereby avoiding the use of an excessively small privacy budget like $\check{\varepsilon}$ for all records.

Then, for each sub-domain $\mathcal{X}_i$, we estimate its counting result with

$$\widetilde{Q}_{\text{count}}(D \cap \mathcal{X}_i) := Q_{\text{count}}(D \cap \mathcal{X}_i) + \text{Lap}\left(\frac{1}{2^{i-1} \cdot \check{\varepsilon}}\right). \tag{2}$$

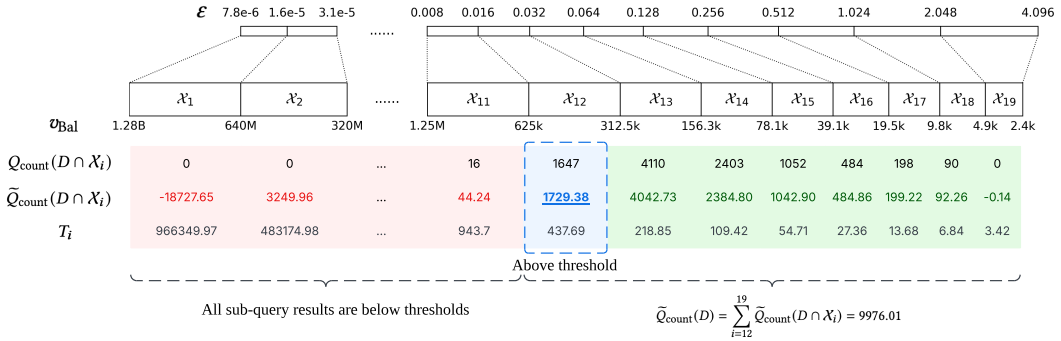


Fig. 1. An illustration of Algorithm 1 for a counting problem with the privacy budget function $\mathcal{E}(r) = \alpha / v_{\text{bal}}$, where $\alpha = 10^4$, $U = 1,280,000,000$ \$, and $\hat{\varepsilon} = 4.096$.

Next, we aim to identify the ℓ-th domain $\mathcal{X}_\ell$ with its minimum privacy budget $2^{\ell-1} \cdot \check{\varepsilon}$, setting it as ε_τ, such that ε_τ is very close to $\varepsilon_{\min}(D)$. Then, we discard all domains with a privacy budget below ε_τ and sum up all the noisy counts $\widetilde{\mathcal{Q}}_{\text{count}}$ for the remaining domains. To achieve our targets (a) and (b), ε_τ should satisfy the following two properties:

(i) $\left|D \cap \left(\bigcup_{i=1}^{\ell-1} \mathcal{X}_i\right)\right| \le \tilde{O}(1/\varepsilon_{\min}(D))$
(ii) $\varepsilon_\tau \ge \frac{1}{2}\varepsilon_{\min}(D)$

Property (i) is to ensure objective (a), that no more than $\tilde{O}(1/\varepsilon_{\min}(D))$ records are excluded. Property (ii) satisfies (b), ensuring that our estimated ε_τ is not overly small, thereby preventing excessive noise.

To achieve these two sub-goals, we identify the first domain $\mathcal{X}_\ell$ such that it contains enough records. More precisely, from $i \in \{1, 2, \ldots, \lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil\}$, we find the first i satisfying

$$\widetilde{\mathcal{Q}}_{\text{count}}(D \cap \mathcal{X}_i) \ge \frac{1}{2^{i-1} \cdot \check{\varepsilon}} \ln\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right),$$

where β is a predefined failure rate.

The right-hand side serves as the threshold for determining whether there are enough records in $\mathcal{X}_i$. These thresholds are calibrated based on Lemma 3.2 to satisfy the desired error bound, which is formalized later in Theorem 5.1. This prevents an "early stop", where the process halts at an empty $\mathcal{X}_i$, thereby satisfying property (ii). At the same time, the "late stop" is also avoided. All sub-domains we passed should be very "light" meaning there are very few records. Otherwise they would pass the thresholds. Through a careful calculation, we can show that all passed domains have at most $\tilde{O}(1)$ records thus satisfying property (i).

To summarize, our final protocol works as follows: First, it identifies the first $\mathcal{X}_\ell$ that contains sufficient records by comparing the query results with carefully calibrated thresholds. Then, we sum all sub-query results starting from the ℓ-th privacy-specified domain. The detailed algorithm is presented in Algorithm 1. The runtime complexity of this algorithm is $O(n)$. Our algorithm also returns $\varepsilon_\tau = 2^{\ell-1} \cdot \check{\varepsilon}$ as an estimate of $\varepsilon_{\min}(D)$, which will be further utilized in our general PrDP framework.

Now, we analyze the privacy and utility of Algorithm 1. For privacy, on any given domain $\mathcal{X}_i$, the algorithm is executed with the minimum privacy budget within the entire domain, ensuring $\frac{1}{2^{i-1} \cdot \check{\varepsilon}}$-DP. Therefore, the mechanism also satisfies $\mathcal{E}$-PrDP on $\mathcal{X}_i$. Notably, any individual record

Algorithm 1: PrDP Count

Input: Privacy budget function $\mathcal{E}(\cdot)$, dataset D, failure rate β
Output: $\widetilde{Q}_{\text{count}}(D)$ under $\mathcal{E}$-PrDP

$\ell \leftarrow \lceil \log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \rceil$
; // Obtain results
for $i \leftarrow \{1, 2, \ldots, \lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil\}$ **do**
  $\widetilde{Q}_{\text{count}}(D \cap \mathcal{X}_i) \leftarrow Q_{\text{count}}(D \cap \mathcal{X}_i) + \text{Lap}\left(\frac{1}{2^{i-1} \cdot \check{\varepsilon}}\right)$
end
; // Estimate ℓ, i.e., ε_τ
for $i \leftarrow \{1, 2, \ldots, \lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil\}$ **do**
  $\mathcal{X}_i \leftarrow \mathcal{X}_{(2^{i-1} \cdot \check{\varepsilon}, \min(2^i \cdot \check{\varepsilon}, \hat{\varepsilon})]}$
  $T_i \leftarrow \frac{1}{2^{i-1} \cdot \check{\varepsilon}} \ln\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right)$
  if $\widetilde{Q}_{\text{count}}(D \cap \mathcal{X}_i) \geq T_i$ **then**
    $\ell \leftarrow i$
    break
  end
end
$\varepsilon_\tau \leftarrow 2^{\ell-1} \cdot \check{\varepsilon}$
; // Aggregate results from the remaining domains
$\widetilde{Q}_{\text{count}}(D) \leftarrow \sum_{i=\ell}^{\lceil \log \frac{\hat{\varepsilon}}{\check{\varepsilon}} \rceil} \widetilde{Q}_{\text{count}}(D \cap \mathcal{X}_i)$
return ε_τ, $\widetilde{Q}_{\text{count}}(D)$

can only affect a single $\mathcal{X}_i$ due to the disjoint partitioning. By the properties of parallel composition (Lemma 3.11), the entire mechanism satisfies $\mathcal{E}$-PrDP.

For utility, Theorem 5.1 provides the error bound for both the counting results and ℓ. We present the proof of the utility of Algorithm 1, as stated in Theorem 5.1.

Theorem 5.1. *For any given privacy budget function $\mathcal{E}(\cdot)$ and $\beta \in (0, 1)$, Algorithm 1 returns ε_τ and $\widetilde{Q}_{count}(D)$, such that with probability at least $1 - \beta$, we have $\varepsilon_\tau \geq \frac{1}{2}\varepsilon_{\min}(D)$, and*

$$\left|\widetilde{Q}_{\text{count}}(D) - Q_{\text{count}}(D)\right| = O\left(\frac{1}{\varepsilon_{\min}(D)} \log\left(\frac{\log(\hat{\varepsilon}/\check{\varepsilon})}{\beta}\right)\right).$$

Proof. We analyze the error introduced by the algorithm, which consists of two components.

The first component arises from omitting the results of the first $\ell - 1$ privacy-specified domains. According to Lemma 3.3 and the union bound over $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ domains, with probability at least $1 - \beta$, we have

$$\begin{aligned} |\tilde{Q}_{\text{count}}(D \cap \mathcal{X}_i) - Q_{\text{count}}(D \cap \mathcal{X}_i)| &\leq T_i \\ &= \frac{1}{2^{i-1} \cdot \check{\varepsilon}} \ln\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right), \end{aligned} \tag{3}$$

which holds for all domains. This means Algorithm 1 will not stop at empty domains with $Q_{\text{count}}(D \cap \mathcal{X}_i) = 0$, to be more specific, we have

$$\varepsilon_\tau \geq \frac{1}{2}\varepsilon_{\min}(D). \tag{4}$$

(3) also indicates that for each domain with $i < \ell$, the number of elements within those domains is very limited even if they are discarded

$$Q_{\text{count}}(D \cap \mathcal{X}_i) \leq 2 \cdot \frac{1}{2^{i-1} \cdot \check{\varepsilon}} \ln\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right),$$

That is, for domains preceding ℓ, they are either empty—if they are before the domain containing $\varepsilon_{\min}(D)$—or very "light" if they coincide with or follow the domain containing $\varepsilon_{\min}(D)$:

$$\begin{aligned}\sum_{i=1}^{\ell-1} Q_{\text{count}}(D \cap \mathcal{X}_i) &= O\left(\frac{1}{\varepsilon_{\min}(D)} \log\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right)\right) \\ &= \tilde{O}\left(\frac{1}{\varepsilon_{\min}(D)}\right).\end{aligned} \tag{5}$$

This part achieves sub-goal (a).

The second component comes from the intrinsic error of the Laplace mechanism. According to (4), the cumulative error over the retained domains is bounded by

$$\begin{aligned}&\sum_{i=\ell}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil} |\tilde{Q}_{\text{count}}(D \cap \mathcal{X}_i) - Q_{\text{count}}(D \cap \mathcal{X}_i)| \\ \leq &\sum_{i=\ell}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil} \frac{1}{2^{i-1} \cdot \check{\varepsilon}} \ln\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right) \\ \leq &O\left(\frac{1}{\varepsilon_\tau} \log\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right)\right) \\ \leq &O\left(\frac{1}{\varepsilon_{\min}(D)} \log\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right)\right) \\ = &\tilde{O}\left(\frac{1}{\varepsilon_{\min}(D)}\right).\end{aligned} \tag{6}$$

From the second to the third line, we observe that the series follows an exponential pattern. Therefore, the summation is on the order of its first term. This bound satisfies sub-goal (b) discussed earlier.

By combining (5) and (6), we obtain the desired error bound stated in Theorem 5.1. □

Example 5.2. Suppose we have a bank dataset with a million records and aim to count customers with a specific mailing postcode. Given the privacy budget function in (1) with $\alpha = 10^4$, $\check{\varepsilon} = 7.8125 \times 10^{-6}$, and $\hat{\varepsilon} = 4.096$, i.e., we set $U = 1,280,000,000$ \$. Figure 1 illustrates how Algorithm 1 works. It first computes noisy counts over all domains and compares them with their respective thresholds. Then, it identifies $\mathcal{X}_{12}$ as the first sub-domain with $\widetilde{Q}_{\text{count}}(D \cap \mathcal{X}_{12}) = 1729.38 > T_{12} = 437.69$. Therefore, the final result is

$$\begin{aligned}\widetilde{Q}_{\text{count}}(D) &= \sum_{i=12}^{19} \widetilde{Q}_{\text{count}}(D \cap \mathcal{X}_i) \\ &= 1729.38 + 4042.73 + 2384.80 \\ &\quad + 1042.90 + 484.86 + 199.22 + 92.26 - 0.14 \\ &= 9976.01.\end{aligned}$$

The ground truth result for this dataset is $\sum_{i=1}^{19} Q_{\text{count}}(D \cap \mathcal{X}_i) = 10,000$. Additionally, we return an $\varepsilon_\tau = 0.016$. □

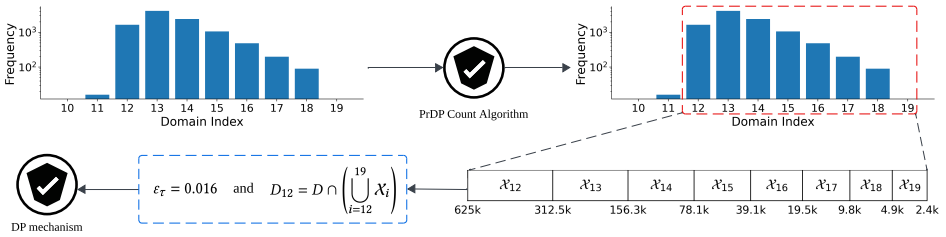


Fig. 2. Illustration of the workflow of Algorithm 2.

5.2 Extensions beyond Basic Counting

In this section, we extend Algorithm 1 to two classic problems: sum estimation and distinct count. These extensions offer deeper insights into the algorithm's strengths and limitations, inspiring a more general framework that eliminates the need for query-specific designs and supports diverse queries and privacy budget functions.

Sum Estimation. For sum estimation, the SOTA mechanism under standard ε-DP achieves an error of $\tilde{O}(\mathrm{Max}(D)/\varepsilon)$. Our goal is to achieve $\tilde{O}(\mathrm{Max}(D)/\varepsilon_{\min}(D))$ in the PrDP setting. To extend Algorithm 1 for sum estimation, we perform a privacy-specified domain partition and estimate the sum within each sub-domain by adding Laplace noise with a scale proportional to $\max_{r\in\mathcal{X}_i}\left(\frac{v_{\mathrm{Bal}}}{\mathcal{E}(r)}\right)$. We then identify the first sub-domain where $\widetilde{Q}_{\mathrm{sum}}(D\cap\mathcal{X}_i) \geq \max_{r\in\mathcal{X}_i}\left(\frac{v_{\mathrm{Bal}}}{\mathcal{E}(r)}\right)\ln\left(\frac{\lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil}{\beta}\right)$. Finally, we return the sum over the domains from the ℓ-th to the $\lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil$-th.

For specific cases, this approach can achieve the desired error bound. For bank deposit estimation with $\mathcal{E}(r) = \alpha/v_{\mathrm{Bal}}$, the error bound is $\tilde{O}(\mathrm{Max}(D)/\varepsilon_{\min}(D))$ with constant probability. However, this approach does not guarantee good utility for all privacy budget functions. For example, when we use $\mathcal{E}(r) = \alpha/\log(v_{\mathrm{Bal}})$, the error bound changes to $\tilde{O}(\mathrm{Max}(D)^2/\varepsilon_{\min}(D))$. This is because, when we partition the domains according to privacy budget, each privacy-specified sub-domain $\mathcal{X}_\ell$ now covers a much wider value range. For example, when we consider the sub-domain that $\mathrm{Max}(D)$ is located in, the maximum balance in that sub-domain can be as large as $\mathrm{Max}(D)^2$. Then, the noise added to the sum result of that sub-domain must be at the scale of $\tilde{O}(\mathrm{Max}(D)^2/\varepsilon_{\min}(D))$. For a detailed discussion and proof, see Appendix A of the full version [11].

Non-Union Preserving Queries. More generally, the idea of Algorithm 1 cannot handle non-union-preserving queries, namely, those for which $Q\left(\bigcup_i D_i\right) \neq \sum_i Q(D_i)$, which is a common scenario in practice. Examples include distinct count, median, maximum, mode estimation, or join counting queries in relational databases. Such queries cannot be decomposed into sub-queries over privacy-specified domains $\mathcal{X}_i$ and summed. To be more precise, consider a bank determining the number of districts its customers reside in, identified by v_{PC}. This reduces to counting distinct v_{PC} in D. If the privacy budget depends on multiple factors, i.e., $\mathcal{E}(r) = \mathcal{E}(v_{\mathrm{Bal}}, v_{\mathrm{PC}}, \ldots)$, the same postcode may appear in multiple privacy-specified domains, leading to duplicate counts and non-additive results, making partitioning inapplicable.

Despite these challenges, the following sections will show how insights from the basic counting algorithm remain valuable. These insights form the foundation of a general framework that supports all query types under PrDP without constraining the privacy budget function's form.

6 A General Framework for PrDP

We have shown that the basic counting problem under PrDP can be effectively solved through privacy-specified domain partitioning, achieving the desired error bound of $\tilde{O}(1/\varepsilon_{\min(D)})$. In this section, we propose a general framework that handles both union and non-union-preserving queries while accommodating any given privacy budget function. Roughly speaking, the general PrDP framework first uses our Laplace-based PrDP counting mechanism to accurately estimate $\varepsilon_{\min}(D)$. It then employs any existing standard DP mechanism with the estimated privacy budget to answer the desired query. Moreover, our framework achieves an instance-specific error bound by reducing the error dependency on the global parameter $\check{\varepsilon}$ to $\varepsilon_{\min}(D)$ for D.

We follow the intuition of Algorithm 1 but split the process into two steps, each consuming half of the privacy budget. In the first step, we estimate a reasonable threshold ε_τ that is not too small while ensuring that most records' privacy budgets fall within the interval $[\varepsilon_\tau, \hat{\varepsilon}]$. This is achieved by invoking Algorithm 1 with the privacy function

$$\mathcal{E}'(r) := \frac{1}{2} \cdot \mathcal{E}(r). \tag{7}$$

After this step, we obtain $\varepsilon_\tau = 2^{\ell-1} \cdot \check{\varepsilon}$ under $\mathcal{E}'$-PrDP, satisfying the following properties:

(i) $\left|D \cap \left(\bigcup_{i=1}^{\ell-1} \mathcal{X}i\right)\right| \leq \tilde{O}(1/\varepsilon_{\min}(D))$
(ii) $\varepsilon_\tau \geq \frac{1}{2}\varepsilon_{\min}(D)$

Then, instead of answering the query on each privacy-specified domain separately and summing up the results, in the second step, we treat the remaining data as a whole. Specifically, we union all sub-domains with domain index larger than ℓ, then invoke a $\frac{\varepsilon_\tau}{2}$- (standard) DP mechanism $\mathcal{M}$ over $\bigcup_{i=\ell}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon})\rceil} \mathcal{X}_i$. For simplicity, we denote $D \cap \left(\bigcup_{i=\ell}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon})\rceil} \mathcal{X}_i\right)$ as D_ℓ. Excluding those "light" domains with overly strong privacy protection needs is critical to our general framework, as it balances utility and privacy to avoid error dominated by $\check{\varepsilon}$, and instead depends only on $\varepsilon_{\min}(D)$. Since the remaining privacy budget is at least $\frac{\varepsilon_\tau}{2}$ across the entire unioned privacy-specified domain, the second step also preserves $\mathcal{E}'$-PrDP.

Please note that the choice of $\mathcal{M}$ is at the discretion of the data curator. They can choose any suitable existing DP mechanism or design a custom one to meet diverse real-world requirements.

This applies to all types of SJA queries, meaning that any standard DP mechanism for such queries can benefit from our framework. However, due to the complex error characteristics of these mechanisms [8, 20–22, 34, 52] (see a survey paper for a comprehensive study [23]), it is not straightforward to derive formal error guarantees for the resulting PrDP solution. In conclusion, our framework serves as a transparent, glass-box transformation that adapts standard DP mechanisms to PrDP settings. It fails to answer a query only when no standard DP mechanism exists for that query.

The detailed algorithm is presented in Algorithm 2. Additionally, Figure 2 illustrates the workflow of the proposed general PrDP framework. The runtime complexity of the framework is primarily determined by the complexity of the selected standard DP mechanism, since estimating ε_τ consumes only $O(n)$.

For privacy, by the sequential composition rule in Lemma 3.10, the entire framework satisfies $2 \cdot \mathcal{E}'$-PrDP, which simplifies to $\mathcal{E}$-PrDP. For the utility, property (i) means at most $\tilde{O}(1/\varepsilon_{\min}(D))$ records will be excluded. Property (ii) implies when we call the $\frac{\varepsilon_\tau}{2}$- (standard) DP mechanism $\mathcal{M}$ in the second step, the error will scale with $\varepsilon_{\min}(D)$.

Algorithm 2: General PrDP Framework

Input: Privacy budget function $\mathcal{E}(\cdot)$, dataset D, failure rate β, DP mechanism $\mathcal{M}$
Output: $\tilde{Q}(D)$ under $\mathcal{E}$-PrDP
; `// Part 1: Estimate` ε_τ
$\varepsilon_\tau \leftarrow$ Algorithm 1, PrDP Count $\left(\mathcal{E}'(r),\, D,\, \frac{\beta}{2}\right)$
; `// Part 2: Execute the DP mechanism`
$\tilde{Q}(D) \leftarrow \mathcal{M}\left(D \cap \left(\bigcup_{i=\ell}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon})\rceil} \mathcal{X}_i\right),\, \frac{1}{2}\,\varepsilon_\tau,\, \frac{\beta}{2}\right)$
return $\tilde{Q}(D)$

Theorem 6.1. *For any given privacy budget function $\mathcal{E}(\cdot)$ and $\beta \in (0,1)$, as well as any DP mechanism $\mathcal{M}$ for query Q, Algorithm 2 returns $\widetilde{Q}(D)$, such that with probability at least $1-\beta$, we have*

$$|\widetilde{Q}(D) - Q(D)| \le \underbrace{|Q(D) - Q(D')|}_{I} + \underbrace{\mathrm{Err}_{\mathcal{M}}\left(D',\, \varepsilon_{\min}(D)/4,\, \beta/2\right)}_{II},$$

where D' is obtained by removing $O\left(\frac{1}{\varepsilon_{\min}(D)} \log \frac{\log(\hat{\varepsilon}/\check{\varepsilon})}{\beta}\right)$ records from D and $Err_{\mathcal{M}}(\cdot,\cdot,\cdot)$ denotes the error introduced by the DP mechanism $\mathcal{M}$, which depends on the input dataset, privacy budget, and failure rate.

Proof. Firstly, according to Theorem 5.1, we know that

$$\begin{aligned}\left|D \setminus D_\ell\right| &\le O\left(\frac{1}{\frac{1}{2}\varepsilon_{\min}(D)} \log \frac{2\lceil \log(\hat{\varepsilon}/\check{\varepsilon})\rceil}{\beta}\right)\\ &= O\left(\frac{1}{\varepsilon_{\min}(D)} \log \frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon})\rceil}{\beta}\right).\end{aligned}$$

Thus, D_ℓ itself satisfies the requirement for D'.

As for the error of the selected mechanism $\mathcal{M}$ over the remaining records D', its magnitude is primarily determined by $\varepsilon_{\min}(D)$. According to Theorem 5.1, we have $\varepsilon_\tau \ge \frac{1}{2}\varepsilon_{\min}(D)$. Therefore, part II holds.

Combining these two parts, we derive the stated error bound in Theorem 6.1.

Finally, the probability that the bound holds with at least $1-\beta$ follows from the union bound over the two consecutive algorithms, each of which guarantees utility with probability at least $1-\frac{\beta}{2}$. □

As shown in the above theorem, the error consists of two components:

I comes from the bias of excluding $\tilde{O}(1/\varepsilon_{\min}(D))$ records.

II comes from the error of invoking the DP mechanism on the remaining records with $\varepsilon_\tau/2$.

We now provide intuition showing that, in most cases, these two components are not significantly larger than

$$\mathrm{Err}_{\mathcal{M}}\left(D,\, \varepsilon_{\min}(D)/4,\, \beta/2\right), \tag{8}$$

which is our target. While this is not a formal guarantee, we will further demonstrate it holds in most common queries such as sum estimation and distinct count.

First, following the notation in Theorem 3.5, component I is exactly $\Delta Q^\rho(D)$ with $\rho = \tilde{O}(1/\varepsilon_{\min}(D))$. As discussed in Section 3.2.2, under standard DP, most existing mechanisms have an error at least

$\Omega(\Delta Q^{\rho}(D))$ with $\rho = \tilde{O}(1/\varepsilon_{\min}(D))$ if we use a privacy budget $\varepsilon = \varepsilon_{\min}(D)$. Therefore, component I will not be significantly larger than (8), at the very least.

Moving towards component II, first, many standard DP mechanisms achieve an error that is independent of the input dataset. Additionally, for common queries such as sum estimation and join counting, existing standard DP mechanisms achieve an instance-specific error that is supermodular with respect to the input dataset D. Intuitively, adding more records does not reduce query sensitivity and, therefore, does not facilitate achieving DP, implying that component II should not exceed (8). Moreover, in other cases where the error function is instance-specific but not supermodular, the difference between datasets D and D' remains limited, ensuring that the errors incurred on D and D' are still very similar.

Sum estimation. As mentioned in Section 5.2, the naive extension fails to attain the desired error bound $\tilde{O}(\mathrm{Max}(D)/\varepsilon_{\min}(D))$ for certain privacy budget functions like $\mathcal{E}(r) = \alpha'/v_{\mathrm{bal}}$. Instead, for Algorithm 2, component I is $\tilde{O}\left(\mathrm{Max}(D)/\varepsilon_{\min}(D)\right)$. For component II, invoking [24] with a privacy budget over D' results in an error of $\tilde{O}(\mathrm{Max}(D')/\varepsilon_{\min}(D))$, which is further bounded by $\tilde{O}(\mathrm{Max}(D)/\varepsilon_{\min}(D))$. Overall, the final error is $\tilde{O}\left(\mathrm{Max}(D)/\varepsilon_{\min}(D)\right)$, aligning with our target.

Distinct count. For distinct count, the standard DP mechanism has an error of $O(1/\varepsilon_{\min}(D))$, therefore, our target under PrDP is $\tilde{O}(1/\varepsilon_{\min}(D))$. For Algorithm 2, component I is $\tilde{O}(1/\varepsilon_{\min}(D))$. Meanwhile, component II is $O(1/\varepsilon_{\min}(D))$. Combining them together gives us the desired $\tilde{O}(1/\varepsilon_{\min}(D))$.

7 Per-Record Local Differential Privacy

In this section, we extend our framework to the *local differential privacy* (LDP) model [6, 45], where we achieve an error with a dependency on $\varepsilon_{\min}(D)$, similar to that in Section 6.

7.1 Per-record Local Differential Privacy

Let us define per-record LDP (PrLDP) first. To begin with, we revisit the definition of LDP. Recall that under central DP, there is a trusted data curator to collect and process data. Under LDP, there is no trusted curator. Instead, each party processes their record locally and then sends the result to the analyzer directly. During this process, each party invokes an LDP protocol to protect the privacy of their record. The formal definition is given below:

Definition 7.1. (ε-Local Differential Privacy). For a given $\varepsilon > 0$, $\mathcal{M}$ is ε-local differential privacy (ε-LDP) if for any pair r, $r' \in [U]^d$ and for all possible outputs y, the following holds:

$$\Pr[\mathcal{M}(r) = y] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(r') = y].$$

This definition ensures that the outputs of the mechanism $\mathcal{M}$ for any pair of records are ε-indistinguishable, thereby providing strong privacy protection for each individual record. Consider the bank deposit estimation scenario in which customers no longer trust the bank to process their records. Each customer therefore performs local processing to privatize their data before sending them to the bank. Because privacy requirements may vary across customers, it is natural to extend Definition 7.1 to a per-record privacy budget, whereby the privacy budget for each customer is determined by a shared privacy budget function—e.g., the function shown in (1). This introduces an additional challenge: determining whether the privacy function should take r or r'. Such an issue does not exist in our PrDP model, as the central DP model defines neighboring datasets using the add-delete-one paradigm, where one dataset is obtained by inserting or deleting a single record from the other. In contrast, the LDP model uses the change-one paradigm, where the record can be changed to an arbitrary one. Under standard DP, it is well established that these two notions provide equivalent levels of privacy protection up to small constant factors. Specifically, any change-one

pair of datasets can be seen as differing by a distance of two under the add-delete-one definition: deleting one record and inserting another. Conversely, by introducing a dummy record $\perp$ that has no influence on the computation, one can simulate the insertion or deletion of a record as a change: replacing a real record with $\perp$, or vice versa.

Following this idea, we define PrLDP under the add-delete-one paradigm using the dummy record $\perp$:

Definition 7.2. (Per-record Local Differential Privacy). Given a record-dependent privacy budget function $\mathcal{E} : [U]^d \to [\check{\varepsilon}, \hat{\varepsilon}]$, a mechanism $\mathcal{M}$ is $\mathcal{E}$-per-record local DP, or simply $\mathcal{E}$-PrLDP, if for arbitrary input record $r \in [U]^d$ and for all possible outputs y, the following holds:

$$e^{-\mathcal{E}(r)} \Pr[\mathcal{M}(\perp) \in y] \le \Pr[\mathcal{M}(r) \in y] \le e^{\mathcal{E}(r)} \Pr[\mathcal{M}(\perp) \in y].$$

For specific queries, we may replace $\perp$ with a meaningful default value. For example, in counting and summation queries, we can set $Q(\perp) := 0$.

Unique challenge of PrLDP. Compared with PrDP, the main challenge of PrLDP is that we need to add noise to each data record. However, since the privacy budget used depends on the value of the record, the scale of the added noise can inadvertently leak information about the record itself. This also highlights the challenge of PrDP over PDP. As previously mentioned, a fundamental distinction between PrDP and PDP lies in that, under PDP, the privacy budgets assigned to individual records are public, making it safe to release record-specific noise. In contrast, PrDP treats these budgets as private.

7.2 Privacy-Specified Query Augmentation for Counting under PrLDP

We first consider the counting problem. Existing LDP protocol achieves an error of $O(\sqrt{n}/\varepsilon)$, where n is the number of parties and each party adds noise with scale $O(1/\varepsilon)$ to their count. Moving to PrLDP, as previously mentioned, utilizing record-specified noise will lead to privacy leakage. To address this issue, we propose a *privacy-specified query augmentation* method. More specifically, we partition the interval $[\check{\varepsilon}, \hat{\varepsilon}]$ into $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ sub-intervals, which correspond to $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ privacy-specified domains. For the i-th domain, we define the query on a data record r as

$$Q_{\text{count},i}(r) = \mathbb{I}(r \in \mathcal{X}_i),$$

where $\mathbb{I}(\cdot)$ denotes the indicator function. Note that $\perp$ does not belong to any domain. Thus, $Q_{\text{count},i}(\perp) \equiv 0$.

Then, the local randomizer further privatizes the $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ real counts by adding Laplace noise with scale $1/(2^{i-1} \cdot \check{\varepsilon})$

$$\widetilde{Q}_{\text{count},i}(r) = Q_{\text{count},i}(r) + \text{Lap}\left(\frac{1}{2^{i-1} \cdot \check{\varepsilon}}\right).$$

Overall, each party sends $\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil$ noisy counting results to augment their response. After receiving responses from all n parties, the data analyzer aggregates the results for each privacy-specified domain and compares them against the threshold

$$T_i := \sqrt{8n} \cdot \frac{1}{2^{i-1} \cdot \check{\varepsilon}} \ln\left(\frac{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}{\beta}\right).$$

Finally, similar to the process in Algorithm 1, it identifies the first domain, $\mathcal{X}_l$, whose result exceeds the corresponding threshold, and sums the responses over all domains with $i \ge l$ as the final result. The detailed algorithms for the local randomizer and the data analyzer are presented in Algorithm 3 and Algorithm 4, respectively.

Algorithm 3: PrLDP Count (Randomizer)

Input: Privacy budget function $\mathcal{E}(r)$, record r_j
Output: $\{\widetilde{Q}_{\text{count},i}(r_j)\}_{i=1}^{\lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil}$ under PrLDP

for $i \leftarrow \{1, 2, \ldots, \lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil\}$ **do**
  $\widetilde{Q}_{\text{count},i}(r_j) \leftarrow Q_{\text{count},i}(r_j) + \text{Lap}\left(\frac{1}{2^{i-1}\cdot\check{\varepsilon}}\right)$
end
Send: $\{\widetilde{Q}_{\text{count},i}(r_j)\}_{i=1}^{\lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil}$

For privacy, each data record affects only a single domain, which is protected by noise calibrated to the strongest privacy requirement of that domain. By using parallel composition across the privacy-specified domains, our privacy-specified query augmentation approach satisfies PrLDP. Regarding utility, Theorem 7.3 provides theoretical guarantees for both $\widetilde{Q}_{\text{count}}(D)$ and ε_τ.

Algorithm 4: PrLDP Count (Data Analyzer)

Input: Noisy answers $\{\{\widetilde{Q}_{\text{count},i}(r_j)\}\}_{i=1}^{\lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil}\}_{j=1}^{n}$, failure rate β
Output: $\widetilde{Q}_{\text{count}}(D)$ under PrLDP

$\ell \leftarrow \left\lceil \log\left(\frac{\hat{\varepsilon}}{\check{\varepsilon}}\right)\right\rceil$
; `// Obtain results`
for $i \leftarrow \{1, 2, \ldots, \lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil\}$ **do**
  $\widetilde{Q}_{\text{count},i} \leftarrow \sum_{j=1}^{n} \widetilde{Q}_{\text{count},i}(r_j)$
end
; `// Estimate` ℓ`, i.e.,` ε_τ
for $i \leftarrow \{1, 2, \ldots, \lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil\}$ **do**
  $T_i \leftarrow \sqrt{8n} \cdot \frac{1}{2^{i-1}\check{\varepsilon}} \ln\left(\frac{\lceil\log(\hat{\varepsilon}/\check{\varepsilon})\rceil}{\beta}\right)$
  if $\widetilde{Q}_{\text{count},i} \geq T_i$ **then**
    $\ell \leftarrow i$
    Break
  end
end
$\varepsilon_\tau \leftarrow 2^{l-1} \cdot \check{\varepsilon}$
; `// Aggregate results from the remaining domains`
$\widetilde{Q}_{\text{count}}(D) \leftarrow \sum_{i=\ell}^{\lceil\log(\frac{\hat{\varepsilon}}{\check{\varepsilon}})\rceil} \widetilde{Q}_{\text{count},i}$
return ε_τ, $\widetilde{Q}_{\text{count}}(D)$

Theorem 7.3. *For any given privacy budget function $\mathcal{E}(\cdot)$ and $\beta \in (0,1)$, the combination of Algorithm 3 and Algorithm 4 returns ε_τ and $\widetilde{Q}_{count}(D)$, such that with probability at least $1-\beta$, we have $\varepsilon_\tau \geq \frac{1}{2}\varepsilon_{\min}(D)$, and*

$$\left|\widetilde{Q}_{\text{count}}(D) - Q_{\text{count}}(D)\right| = O\left(\frac{\sqrt{n}}{\varepsilon_{\min}(D)} \log\left(\frac{\log(\hat{\varepsilon}/\check{\varepsilon})}{\beta}\right)\right).$$

Proof. Algorithm 4 satisfies PrLDP by Lemma 3.9. For utility, the analysis follows similarly to the proof of Theorem 5.1, where the $\sqrt{n}$ factor arises from aggregating n noisy responses—standard in LDP settings. More precisely, for n independent and identically distributed Laplace noises, the variance of their sum increases by a factor of n, resulting in a larger error magnitude proportional to $\sqrt{n}$. □

7.3 A General Framework for PrLDP

In the previous section, we introduced a counting algorithm under PrLDP based on the privacy-specified query augmentation technique. Following a similar strategy as the centralized setting, it is not hard to build a general framework for extending existing LDP protocol to the PrLDP model.

Our protocol involves two rounds. Specifically, in the first round, we use Algorithm 3 and Algorithm 4 to estimate ε_τ with privacy budget function $\mathcal{E}'(r) := \frac{1}{2}\mathcal{E}(r)$. In the second round, we exclude records whose privacy budgets fall below ε_τ. More precisely, each party will replace its data record with $\perp$ if that record corresponding to a privacy budget less than ε_τ. Then, each party invokes a LDP protocol with a privacy budget of $\frac{\varepsilon_\tau}{2}$. The details are provided in Algorithm 5. Roughly speaking, similar to the general framework for PrDP, the PrLDP general framework first uses our PrLDP counting protocol to obtain ε_τ, and then sets the records with privacy budgets below this threshold to dummy records. Finally, it applies an existing LDP protocol to answer the desired queries. Therefore, it supports any query for which an LDP protocol exists, regardless of whether it is union-preserving or non-union-preserving. For example, since no existing LDP protocol supports distinct count, our framework does not support this type of query.

For privacy, by Lemma 3.11, the first round of computation preserves $\frac{1}{2}\mathcal{E}$-PrLDP. In the second round, if a party's record corresponds to a privacy budget less than ε_τ, setting it to $\perp$ prevents any privacy leakage. Otherwise, the privacy loss is bounded by $\frac{\varepsilon_\tau}{2}$. Therefore, the second round also satisfies $\frac{1}{2}\mathcal{E}$-PrLDP. By the composition theorem, the overall framework satisfies $\mathcal{E}$-PrLDP.

Algorithm 5: General PrLDP Framework

Input: Privacy budget function $\mathcal{E}(\cdot)$, records $\{r_j\}_{j=1}^n$, failure rate β, LDP mechanism $\mathcal{M}_{\text{local}}$, $\mathcal{M}_{\text{curator}}$

Output: $\widetilde{Q}(D)$ under PrLDP

; // Round I: estimate ε_τ

for $j \leftarrow \{1, 2, \ldots, n\}$ **do**

  $\{\widetilde{Q}_{\text{count},i}(r_j)\}_{i=1}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil} \leftarrow \text{randomizer}\left(\mathcal{E}'(r),\ r_j\right)$

end

$\varepsilon_\tau \leftarrow \text{analyzer}\left(\{\{\widetilde{Q}_{\text{count},i}(r_j)\}\}_{i=1}^{\lceil \log(\hat{\varepsilon}/\check{\varepsilon}) \rceil}\}_{j=1}^n,\ \frac{\beta}{2}\right)$

; // Round II: execute the LDP protocal

for $j \leftarrow 1$ ***to*** n **do**

  ; // r with small privacy budget respond as $\perp$

  $\widetilde{Q}_j \leftarrow \mathcal{M}_{\text{local}}\left(\frac{\varepsilon_\tau}{2}, \text{ if } \mathcal{E}(r_j) \geq \varepsilon_\tau \text{ then } r_j \text{ else } \perp\right)$

end

$\widetilde{Q}(D) \leftarrow \mathcal{M}_{\text{curator}}(\{\widetilde{Q}_j\}_{j=1}^n,\ \frac{\beta}{2})$

return $\widetilde{Q}(D)$

For utility, we have

Theorem 7.4. *For any privacy budget function $\mathcal{E}(\cdot)$, $\beta \in (0,1)$, and any LDP protocol $\mathcal{M}$, Algorithm 5 returns an estimate $\widetilde{Q}(D)$ such that, with probability at least $1-\beta$,*

$$|\widetilde{Q}(D) - Q(D)| \le \underbrace{|Q(D) - Q(D')|}_{(I)} + \underbrace{\mathrm{Err}_{\mathcal{M}}(D',\ \varepsilon_{\min}(D)/4,\ \beta/2)}_{(II)},$$

where D' is obtained by removing $O\left(\frac{\sqrt{n}}{\varepsilon_{\min}(D)} \log \frac{\log(\hat{\varepsilon}/\check{\varepsilon})}{\beta}\right)$ records from D, and $\mathrm{Err}_{\mathcal{M}}(\cdot,\cdot,\cdot)$ denotes the error introduced by the LDP protocol, which depends on the dataset, privacy budget, and failure rate.

Proof. For privacy, the result follows from the sequential composition of the two components in the framework. For utility, the argument mirrors that of Theorem 6.1, with the $\sqrt{n}$ term carried over from Theorem 7.3. In particular, at most $\tilde{O}(\sqrt{n}/\varepsilon_{\min}(D))$ users return $\perp$, and the remaining users contribute through a $\frac{\varepsilon_\tau}{2}$-LDP mechanism. □

The error consists of two components, both of which depend on $\varepsilon_{\min}(D)$. Below, we use sum estimation as an example to demonstrate that our mechanism achieves the target error of $O(\mathrm{Err}_{\mathcal{M}}(D, \varepsilon_{\min}(D), \beta))$. First, in sum estimation, the SOTA LDP mechanism achieves an error of $\tilde{O}(\sqrt{n} \cdot \mathrm{Max}(D)/\varepsilon_{\min}(D))$ [33]. In our mechanism, component I is $\tilde{O}(\sqrt{n} \cdot \mathrm{Max}(D)/\varepsilon_{\min}(D))$, and component II is also $\tilde{O}(\sqrt{n} \cdot \mathrm{Max}(D)/\varepsilon_{\min}(D))$. This matches exactly the error of the LDP protocol $\mathcal{M}$ in [33] on D under a privacy budget of $\varepsilon_{\min}(D)$.

8 Experiments

In this section, we evaluate the proposed PrDP algorithms and compare them with existing methods. For empirical results of PrLDP, please refer to Appendix B [11]. We also include experiments with our PrDP framework on an SJA query, two-line path counting, in Appendix C [11]

Basic Counting. We evaluate our PrDP basic counting algorithm (Algorithm 1) against two baselines: the naive mechanism using $\check{\varepsilon}$ and the SOTA PDP method presented in [51]. The PDP method is selected for the same reasons discussed earlier in Section 3.4. For clarification, in this section, the basic counting query aims to privately report the number of records n in the dataset D.

Sum Estimation. For sum estimation, we evaluate our PrDP count (Algorithm 1) extension and PrDP framework (Algorithm 2), where we use the Laplace-based standard DP sum estimation mechanism from [24] to serve as $\mathcal{M}$. We compare them against two baselines: a naive use of $\check{\varepsilon}$ and the SOTA PDP method in [51].

Max. For the max problem, our PrDP framework (Algorithm 2) is compared with a naive use of $\check{\varepsilon}$, where the exponential-based max-selection mechanism from [3, 24] is used as $\mathcal{M}$.

8.1 Setup

Datasets. Both real-world and synthetic datasets are used in this work. The synthetic datasets are generated from two distributions: the Normal distribution ($f(x) \propto \exp\left(-((x-\mu)^2)/(2\sigma^2)\right)$) and the Zipf distribution ($f(x) \propto (x+a)^{-b}$). For the Normal distribution, we set $\mu = 50$k, $\sigma = 50$k and $\mu = 500$k, $\sigma = 500$k. For the Zipf distribution, we use $a = 1$, $b = 3$ and $a = 1$, $b = 5$. This results in a total of four synthetic bank datasets. We use four real-world datasets representing practical needs across various financial scenarios, including a real bank dataset [39], salary payment datasets—San Francisco Salary (SF-Salary) [36] and Ontario Salary (Ont-Salary) [38]—and a market trade dataset, Japan Trade (JP-Trade) [37]. For clarity and simplicity, all datasets are preprocessed to exclude

negative or abnormal values [4]. Since the JP-Trade dataset contains 100 million records, we select only trades to a specific country as a representative subset. Table 2 presents the statistics of these real-world datasets. As previously discussed, U is consistently set to 10^{12} in all these financial settings, except for the scaling experiments where U varies. This conservatively large value assumes no prior knowledge of the datasets, thereby ensuring stronger privacy.

Dataset	n	Max(D)	mean D	median D
Bank	3.27×10^4	1.02×10^5	1.91×10^3	8.34×10^2
SF-Salary	1.48×10^5	5.68×10^5	7.52×10^4	7.17×10^4
Ont-Salary	5.75×10^5	1.75×10^6	1.27×10^5	1.16×10^5
JP-Trade	2.03×10^5	1.71×10^6	2.20×10^3	3.20×10^2

Table 2. Statistics of real-world datasets.

Privacy Specification. We evaluate three different types of privacy specifications in total. For the experiments presented in the tables in this section, we use $\mathcal{E}(r) = 10^4/v_{\text{Bal}}$ (*inverse*) as the privacy budget function. For the subsequent figures, where we evaluate performance under varying dataset sizes, U, and data distribution, we additionally consider two alternative functions to enrich the privacy specification: $\mathcal{E}(r) = 500/\log^4 v_{\text{Bal}}$ (*log*) and $\mathcal{E}(r) = 8/\sqrt{v_{\text{Bal}}}$ (*sqrt*). We set $\check{\varepsilon} = \mathcal{E}(U)$ and $\hat{\varepsilon} = 100$ consistently across all experiments.

Experimental Parameters. All experiments are conducted on a PC equipped with an Apple M4 Max CPU (16 cores) and 128 GB of memory. The failure rate is consistently set to $\beta = 10\%$, and each experiment is repeated 50 times. We use absolute error for basic counting and sum estimation, and rank error for the max query. The average relative error (RE) and running time are reported after discarding the top 20% and bottom 20% of the results, thereby conservatively reflecting the 10% failure rate.

8.2 Results

8.2.1 Utility and Runtime. Table 3 presents the utility and runtime of our algorithms and corresponding baselines for basic counting, sum estimation, and max queries. The results clearly demonstrate that our PrDP mechanisms improve utility across all query types on both synthetic and real-world datasets, while reducing runtime by up to 16×. For count and sum estimation, our PrDP extension achieves 2× to 50× lower RE, and the PrDP framework achieves 2× to 165× lower RE compared to SOTA PDP methods. Despite their relaxed privacy guarantees, our mechanisms provide superior utility.

Regarding our two proposed methods, since we use the *inverse* privacy budget function, both methods theoretically share the same asymptotic error bound. However, the PrDP framework allocates part of the privacy budget to estimate ε_τ, resulting in a higher error than the PrDP extension by a constant factor, as confirmed by our experiments on synthetic datasets. However, in real-world datasets, we observe that the PrDP framework outperforms the PrDP extension by some small constant factors. This is because real-world data tends to exhibit more skewed distributions than synthetic datasets, resulting in a greater number of domain partitions with sparse data. For

[4]Negative values can be supported by shifting the domain by a sufficiently large constant, executing the mechanism over the shifted records, and finally shifting the query results back. The privacy budget function must also be defined to accommodate negative inputs—for example, by using absolute values.

Dataset			Synthetic Data				Real-world Data			
			Normal (200k)		Zipf (200k)		Bank	SF-Salary	Ont-Salary	JP-Trade
			μ = 50k σ = 50k	μ = 500k μ = 500k	a=1 b=3	a=1 b=5				
Basic Counting	Naive	RE (%)	> 100	> 100	> 100	> 100	> 100	> 100	> 100	> 100
		Time (s)	0.018	0.020	0.019	0.025	0.0017	0.013	0.035	0.017
	SOTA PDP	RE (%)	2.27	13.69	1.28	0.641	1.34	3.29	1.17	0.686
		Time (s)	1.83	2.22	1.74	1.74	0.24	1.34	5.82	1.25
	PrDP Count (Ours)	RE (%)	0.0138	0.279	0.00941	0.0196	0.0357	0.0390	0.0505	0.0647
		Time (s)	1.06	0.95	1.29	1.33	0.23	0.78	2.86	1.41
Sum Estimation	Naive	RE (%)	100	100	100	100	100	100	100	100
		Time (s)	0.13	0.12	0.24	0.24	0.035	0.089	0.35	0.12
	SOTA PDP	RE (%)	10.68	48.80	17.44	7.18	21.25	14.92	5.40	38.25
		Time (s)	15.17	15.25	15.12	14.91	2.32	10.64	43.94	14.80
	PrDP Extension (Ours)	RE (%)	0.0358	0.967	0.198	0.321	3.54	0.984	0.244	18.91
		Time (s)	1.13	0.98	1.40	1.43	0.25	0.85	3.30	1.53
	PrDP Framework (Ours)	RE (%)	0.187	1.65	1.04	0.544	2.25	0.527	0.225	18.23
		Time (s)	1.41	1.31	1.86	1.95	0.33	1.06	3.98	1.73
Max	Naive	RE (%)	> 100	> 100	> 100	> 100	> 100	> 100	> 100	> 100
		Time (s)	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.	N.A.
	PrDP Framework (Ours)	RE (%)	0.738	6.15	0.734	0.387	1.32	2.02	0.642	0.757
		Time (s)	1.30	1.14	1.61	1.73	0.29	0.92	3.41	1.60

Table 3. Comparison of different mechanisms under PrDP for basic counting, sum estimation, and max. RE stands for relative error.

these sparse partitions, the PrDP framework estimates counts, while the PrDP extension estimates sums. Since count estimation generally involves a lower threshold, results are more likely to pass the threshold, leading to fewer discarded records under the PrDP framework.

For the Max problem, the naive use of $\check{\varepsilon}$ violates the requirement on n in the base standard DP mechanism of [3, 24], which requires $n > \tilde{O}(1/\check{\varepsilon})$ thus rendering the method ineffective. This further highlights the superiority of our PrDP framework, which achieves a maximum relative rank error of 6.15% across various datasets while maintaining efficiency.

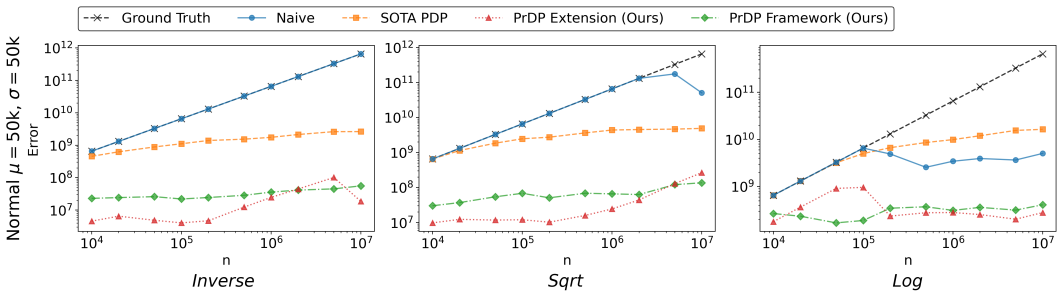


Fig. 3. Comparison of errors in sum estimation under PrDP across different mechanisms, with varying dataset sizes n. All aixs are in log scale.

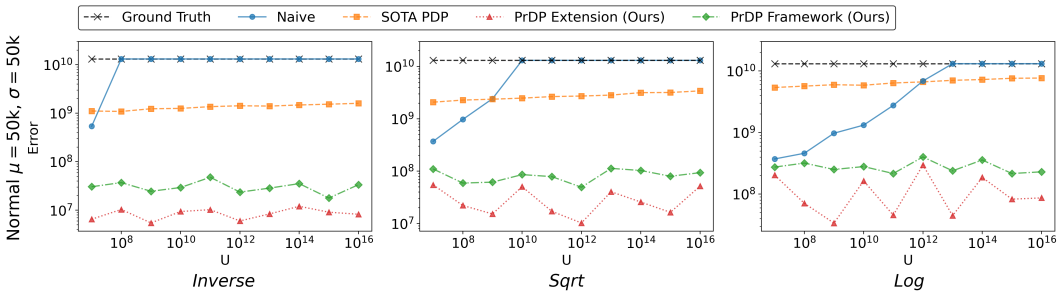


Fig. 4. Comparison of errors in sum estimation under PrDP across different mechanisms, with varying U. All aixs are in log scale.

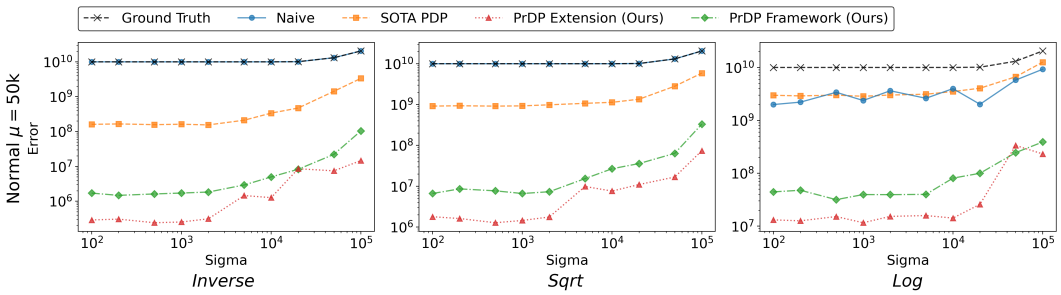


Fig. 5. Comparison of errors in sum estimation under PrDP across different mechanisms, with varying σ. All aixs are in log scale.

8.2.2 Dataset Size and Privacy Budget Functions. To evaluate the performance of our algorithms across different dataset sizes and privacy functions, we conduct sum estimation experiments using synthetic datasets drawn from a normal distribution with $\mu = \sigma = 50$k, with data sizes from 10^4 to 10^7, and apply all three privacy budget functions. The results are shown in Figure 3. Here, both the PrDP general framework and the extension exhibit high utility across all dataset sizes and outperform the baseline methods. Under the *inverse* and *Sqrt* budget function, PrDP extension outperforms the framework by a constant factor in most cases as the framework requires to divide the privacy budget to estimate ε_τ. However, with *Log* budget functions, the framework outperforms the extension in certain scenarios, as the PrDP extension does not satisfy our target utility guarantee, whereas the framework holds for arbitrary privacy function, as discussed in Section 5.2.

8.2.3 Global bound U and Data Skewness. To examine the impact of U and data skewness, we perform two experiments using 200,000-record synthetic datasets. In Figure 4, we fix $\mu = \sigma = 50$k and vary U from 10^4 to 10^{16}. The results show that our two proposed methods consistently perform better than the baselines, whereas the baselines exhibit utility degradation as U increases.

In Figure 5, we fix $\mu = 50$k and vary σ from 10^2 to 10^5. The absolute error increases with data skewness due to two factors: (1) larger σ values imply higher $\mathrm{Max}(D)$, necessitating stronger protection (i.e., lower $\varepsilon_{\min}(D)$), and (2) extreme skewness leads to sparse data, making threshold selection more difficult and increasing error. In all cases, both the PrDP extension and framework consistently outperform baselines. Note that even under stronger data skewness (larger σ), the framework outperforms the extension in only a few cases. This is because synthetic datasets exhibit

smooth tails and therefore cannot replicate the sudden spikes present in real-world data, leading to different results compared to those in Table 3 for real-world datasets.

9 CONCLUSION & FUTURE WORK

In this study, we propose the first general framework for per-record privacy requirements in both central (PrDP) and local (PrLDP) settings. Our framework accommodates any type of privacy requirement, provided that a corresponding standard DP or LDP mechanism exists. This serves as a transparent foundation for addressing per-record privacy needs across a wide range of DP research. A current limitation is that the framework supports only static databases. An important direction for future work is to extend it to dynamic settings [9, 15, 19, 27, 55], where records with different privacy requirements arrive in a stream and query results are released continuously. This scenario introduces new challenges in preserving privacy while maintaining utility, as $\varepsilon_{\min}(D)$ evolves over time. Another direction is to extend the framework to support the shuffle-DP model [4, 12, 43], a distributed setting that improves utility by introducing a shuffler between the parties and the analyzer to break the linkage between each record and its corresponding party. Existing work on shuffle-DP has not yet considered the case where the privacy budget is allocated on a per-record basis, which represents a more general and broadly relevant setting.

Acknowledgments

This research is supported by the NTU–NAP Startup Grant (024584-00001) and the Singapore Ministry of Education Tier 1 Grant (RG19/25). We would also like to thank the anonymous reviewers who have made valuable suggestions on improving the presentation of the paper.

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. 2016. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*. 308–318.

[2] Mohammad Alaggan, Sébastien Gambs, and Anne-Marie Kermarrec. 2016. Heterogeneous Differential Privacy. *Journal of Privacy and Confidentiality* 7, 2 (2016).

[3] Hilal Asi and John C Duchi. 2020. Instance-optimality in differential privacy via approximate inverse sensitivity mechanisms. *Advances in neural information processing systems* 33 (2020), 14106–14117.

[4] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The privacy blanket of the shuffle model. In *Annual International Cryptology Conference*. Springer, 638–667.

[5] Ting Bao, Lei Xu, Liehuang Zhu, Lihong Wang, and Tielei Li. 2021. Successive point-of-interest recommendation with personalized local differential privacy. *IEEE Transactions on Vehicular Technology* 70, 10 (2021), 10477–10488.

[6] Raef Bassily and Adam Smith. 2015. Local, private, efficient protocols for succinct histograms. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*. 127–135.

[7] Franziska Boenisch, Christopher Mühl, Adam Dziedzic, Roy Rinberg, and Nicolas Papernot. 2023. Have it your way: Individualized Privacy Assignment for DP-SGD. *Advances in Neural Information Processing Systems* 36 (2023), 19073–19103.

[8] Kuntai Cai, Xiaokui Xiao, and Graham Cormode. 2023. Privlava: synthesizing relational data with foreign keys under differential privacy. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–25.

[9] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. 2011. Private and Continual Release of Statistics. *ACM Transactions on Information and System Security* (2011).

[10] Rui Chen, Haoran Li, A Kai Qin, Shiva Prasad Kasiviswanathan, and Hongxia Jin. 2016. Private spatial data aggregation in the local setting. In *2016 IEEE 32nd International Conference on Data Engineering (ICDE)*. IEEE, 289–300.

[11] Xinghe Chen, Dajun Sun, Quanqing Xu, and Wei Dong. 2025. A General Framework for Per-record Differential Privacy [Full Version]. (2025). https://drive.google.com/drive/folders/1U1HDfh8FgyFq1ddsrRJFx8tU9uMgKK0a?usp=sharing

[12] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed differential privacy via shuffling. In *Annual international conference on the theory and applications of cryptographic techniques*. Springer, 375–403.

[13] Lei Cui, Youyang Qu, Mohammad Reza Nosouhi, Shui Yu, Jian-Wei Niu, and Gang Xie. 2019. Improving data utility through game theory in personalized differential privacy. *Journal of Computer Science and Technology* 34 (2019), 272–286.

[14] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting Telemetry Data Privately. In *NeurIPS*.

[15] Wei Dong, Zijun Chen, Qiyao Luo, Elaine Shi, and Ke Yi. 2024. Continual observation of joins under differential privacy. *Proceedings of the ACM on Management of Data* 2, 3 (2024), 1–27.

[16] Wei Dong, Juanru Fang, Ke Yi, Yuchao Tao, and Ashwin Machanavajjhala. 2022. R2T: Instance-optimal Truncation for Differentially Private Query Evaluation with Foreign Keys. In *Proc. ACM SIGMOD International Conference on Management of Data*.

[17] Wei Dong, Juanru Fang, Ke Yi, Yuchao Tao, and Ashwin Machanavajjhala. 2024. Instance-optimal Truncation for Differentially Private Query Evaluation with Foreign Keys. *ACM Transactions on Database Systems* 49, 4 (2024), 1–40.

[18] Wei Dong, Qiyao Luo, Giulia Fanti, Elaine Shi, and Ke Yi. 2024. Almost instance-optimal clipping for summation problems in the shuffle model of differential privacy. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 1939–1953.

[19] Wei Dong, Qiyao Luo, and Ke Yi. 2023. Continual Observation under User-level Differential Privacy. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE Computer Society, 2190–2207.

[20] Wei Dong, Dajun Sun, and Ke Yi. 2023. Better than Composition: How to Answer Multiple Relational Queries under Differential Privacy. *Proceedings of the ACM on Management of Data* 1, 2 (2023), 1–26.

[21] Wei Dong and Ke Yi. 2021. Residual Sensitivity for Differentially Private Multi-Way Joins. In *Proc. ACM SIGMOD International Conference on Management of Data*.

[22] Wei Dong and Ke Yi. 2022. A Nearly Instance-optimal Differentially Private Mechanism for Conjunctive Queries. In *Proc. ACM Symposium on Principles of Database Systems*.

[23] Wei Dong and Ke Yi. 2023. Query Evaluation under Differential Privacy. *ACM SIGMOD Record* 52, 3 (2023), 6–17.

[24] Wei Dong and Ke Yi. 2023. Universal private estimators. In *Proceedings of the 42nd ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 195–206.

[25] Cynthia Dwork and Jing Lei. 2009. Differential Privacy and Robust Statistics. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. 371–380.

[26] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*. Springer, 265–284.

[27] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. 2010. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*. 715–724.
[28] Hamid Ebadi, David Sands, and Gerardo Schneider. 2015. Differential privacy: Now it's getting personal. *Acm Sigplan Notices* 50, 1 (2015), 69–81.
[29] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC conference on computer and communications security*. 1054–1067.
[30] Juanru Fang, Wei Dong, and Ke Yi. 2022. Shifted Inverse: A General Mechanism for Monotonic Functions under User Differential Privacy. (2022).
[31] Vitaly Feldman and Tijana Zrnic. 2021. Individual privacy accounting via a renyi filter. *Advances in Neural Information Processing Systems* 34 (2021), 28080–28091.
[32] Bugra Gedik and Ling Liu. 2007. Protecting location privacy with personalized k-anonymity: Architecture and algorithms. *IEEE Transactions on Mobile Computing* 7, 1 (2007), 1–18.
[33] Ziyue Huang, Yuting Liang, and Ke Yi. 2021. Instance-optimal Mean Estimation Under Differential Privacy. In *NeurIPS*.
[34] Noah Johnson, Joseph P Near, and Dawn Song. 2018. Towards practical differential privacy for SQL queries. *Proceedings of the VLDB Endowment* 11, 5 (2018), 526–539.
[35] Zach Jorgensen, Ting Yu, and Graham Cormode. 2015. Conservative or liberal? Personalized differential privacy. In *2015 IEEE 31St international conference on data engineering*. IEEE, 1023–1034.
[36] Kaggle. 2014. San Francisco City Employee Salary Data. https://www.kaggle.com/datasets/kaggle/sf-salaries/data.
[37] Kaggle. 2020. Japan's 100 million customs trade statistics since 1988. https://www.kaggle.com/datasets/zanjibar/100-million-data-csv.
[38] Kaggle. 2020. Ontario Public Sector Salary 2019. https://www.kaggle.com/datasets/rajacsp/ontario.
[39] Kaggle. 2023. Banking Dataset - Marketing Targets. https://www.kaggle.com/datasets/prakharrathi25/banking-dataset-marketing-targets.
[40] Haoran Li, Li Xiong, Zhanglong Ji, and Xiaoqian Jiang. 2017. Partitioning-based mechanisms under personalized differential privacy. In *Advances in Knowledge Discovery and Data Mining: 21st Pacific-Asia Conference, PAKDD 2017, Jeju, South Korea, May 23-26, 2017, Proceedings, Part I 21*. Springer, 615–627.
[41] Xiaoguang Li, Haonan Yan, Zelei Cheng, Wenhai Sun, and Hui Li. 2022. Protecting regression models with personalized local differential privacy. *IEEE Transactions on Dependable and Secure Computing* 20, 2 (2022), 960–974.
[42] Junxu Liu, Jian Lou, Li Xiong, Jinfei Liu, and Xiaofeng Meng. 2024. Cross-silo federated learning with record-level personalized differential privacy. In *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. 303–317.
[43] Qiyao Luo, Jianzhe Yu, Wei Dong, Quanqing Xu, Chuanhui Yang, and Ke Yi. 2025. RM2: Answer Counting Queries Efficiently under Shuffle Differential Privacy. *Proceedings of the ACM on Management of Data* 3, 3 (2025), 1–24.
[44] Ashwin Machanavajjhala, Daniel Kifer, John Abowd, Johannes Gehrke, and Lars Vilhuber. 2008. Privacy: Theory meets practice on the map. In *2008 IEEE 24th international conference on data engineering*. IEEE, 277–286.
[45] Andrew McGregor, Ilya Mironov, Toniann Pitassi, Omer Reingold, Kunal Talwar, and Salil Vadhan. 2010. The limits of two-party differential privacy. In *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*. IEEE, 81–90.
[46] Ben Niu, Yahong Chen, Boyang Wang, Zhibo Wang, Fenghua Li, and Jin Cao. 2021. AdaPDP: Adaptive personalized differential privacy. In *IEEE INFOCOM 2021-IEEE conference on computer communications*. IEEE, 1–10.
[47] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Ulfar Erlingsson. 2018. Scalable Private Learning with PATE. In *International Conference on Learning Representations*.
[48] Rachel Redberg and Yu-Xiang Wang. 2021. Privately publishable per-instance privacy. *Advances in Neural Information Processing Systems* 34 (2021), 17335–17346.
[49] Jeremy Seeman, William Sexton, David Pujol, and Ashwin Machanavajjhala. 2024. Privately Answering Queries on Skewed Data via Per-Record Differential Privacy. *Proc. VLDB Endow.* 17, 11 (July 2024), 3138–3150.
[50] Yanguang Shen, Hui Shao, and Yan Li. 2009. Research on the personalized privacy preserving distributed data mining. In *2009 Second International Conference on Future Information Technology and Management Engineering*. IEEE, 436–439.
[51] Dajun Sun, Wei Dong, Yuan Qiu, and Ke Yi. 2024. Personalized Truncation for Personalized Privacy. *Proceedings of the ACM on Management of Data* 2, 6 (2024), 1–25.
[52] Yuchao Tao, Xi He, Ashwin Machanavajjhala, and Sudeepa Roy. 2020. Computing Local Sensitivities of Counting Queries with Joins. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*. 479–494.
[53] Apple Differential Privacy Team. 2017. Learning with Privacy at Scale. https://machinelearning.apple.com/research/learning-with-privacy-at-scale.
[54] Xiaokui Xiao and Yufei Tao. 2006. Personalized privacy preservation. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*. 229–240.

[55] Dongdong Xie, Pinghui Wang, Quanqing Xu, Chuanhui Yang, and Rundong Li. 2025. Efficient and Accurate Differentially Private Cardinality Continual Releases. *Proceedings of the ACM on Management of Data* 3, 3 (2025), 1–27.
[56] Qiao Xue, Youwen Zhu, and Jian Wang. 2022. Mean estimation over numeric data with personalized local differential privacy. *Frontiers of Computer Science* 16 (2022), 1–10.
[57] Ruilin Yang, Hui Yang, Jiluan Fan, Changyu Dong, Yan Pang, Duncan S Wong, and Shaowei Wang. 2023. Personalized Differential Privacy in the Shuffle Model. In *International Conference on Artificial Intelligence Security and Privacy*. Springer, 468–482.
[58] Xiaojun Ye, Yawei Zhang, and Ming Liu. 2008. A personalized (a, k)-anonymity model. In *2008 The Ninth International Conference on Web-Age Information Management*. IEEE, 341–348.
[59] NIE Yiwen, Wei Yang, Liusheng Huang, Xike Xie, Zhenhua Zhao, and Shaowei Wang. 2018. A utility-optimized framework for personalized private histogram estimation. *IEEE Transactions on Knowledge and Data Engineering* 31, 4 (2018), 655–669.

Received April 2025; revised July 2025; accepted August 2025