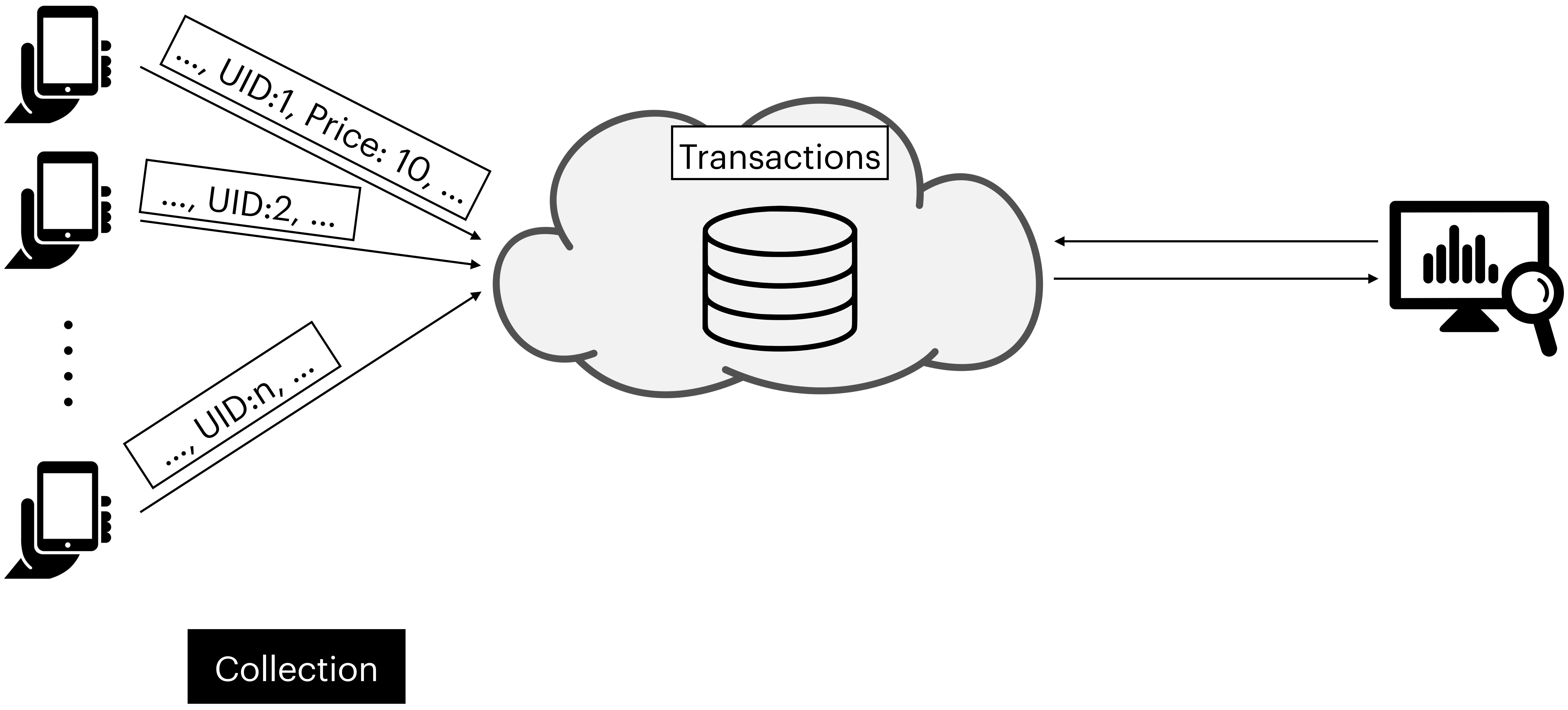
Collecting and analyzing data over multiple services under local differential privacy

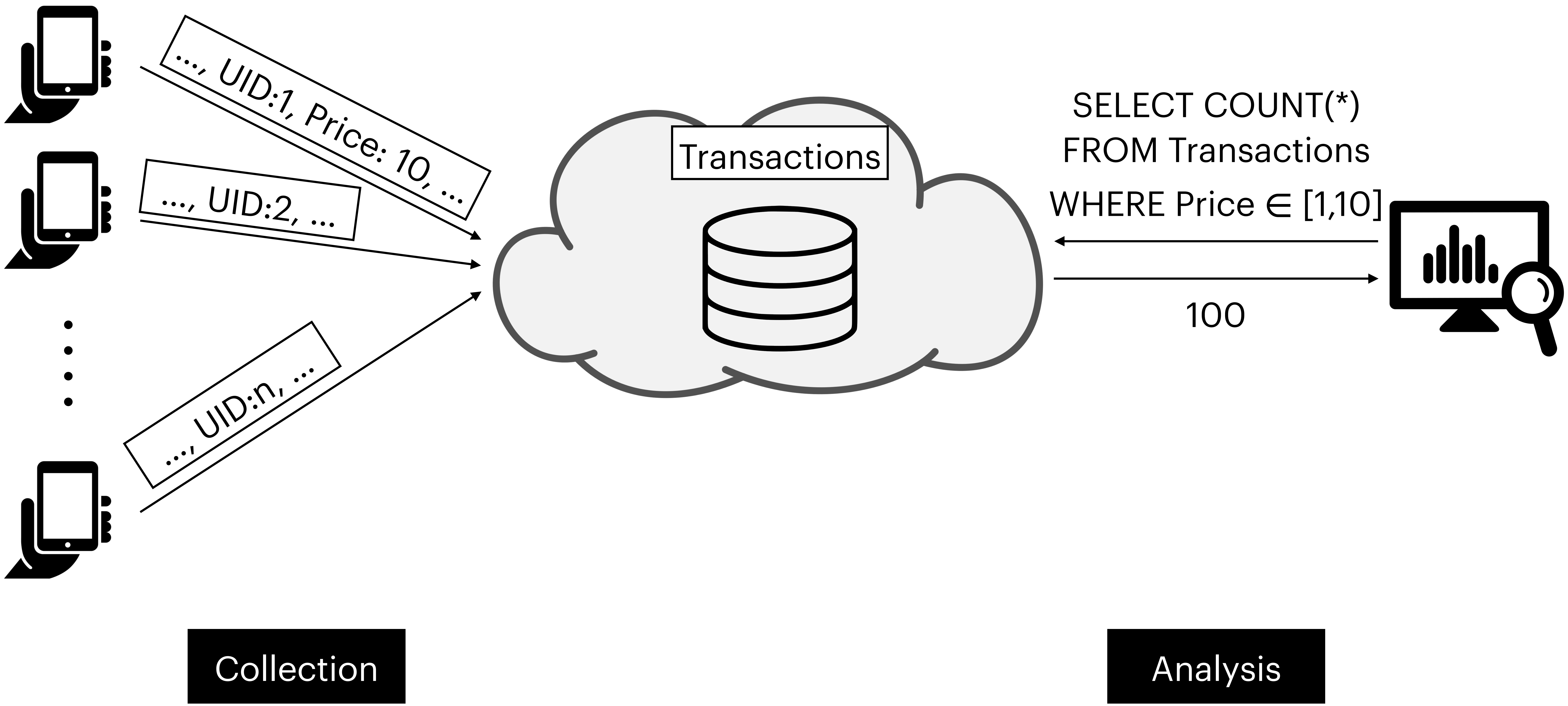
Min Xu, Bolin Ding, Tianhao Wang, Jingren Zhou

Data Collection and Analysis in the Cloud

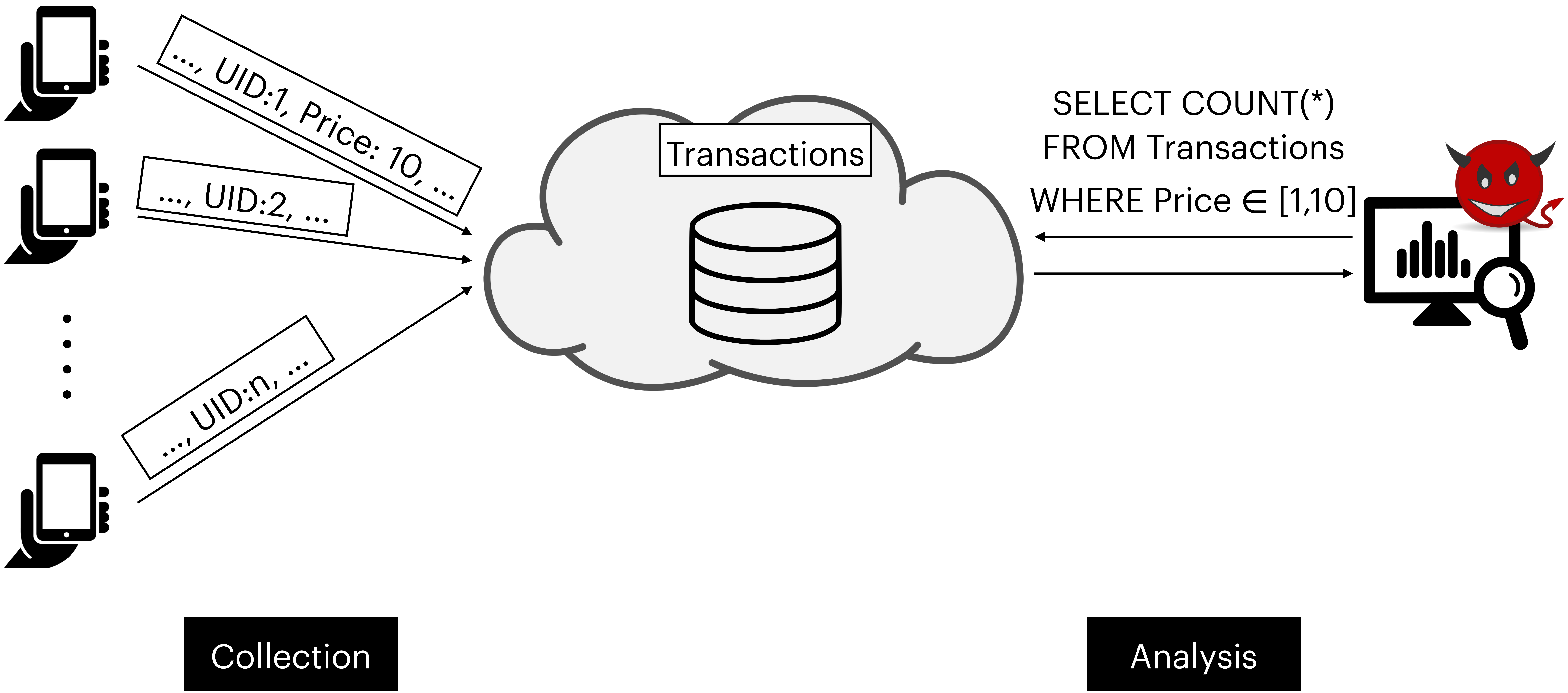
Data Collection and Analysis in the Cloud



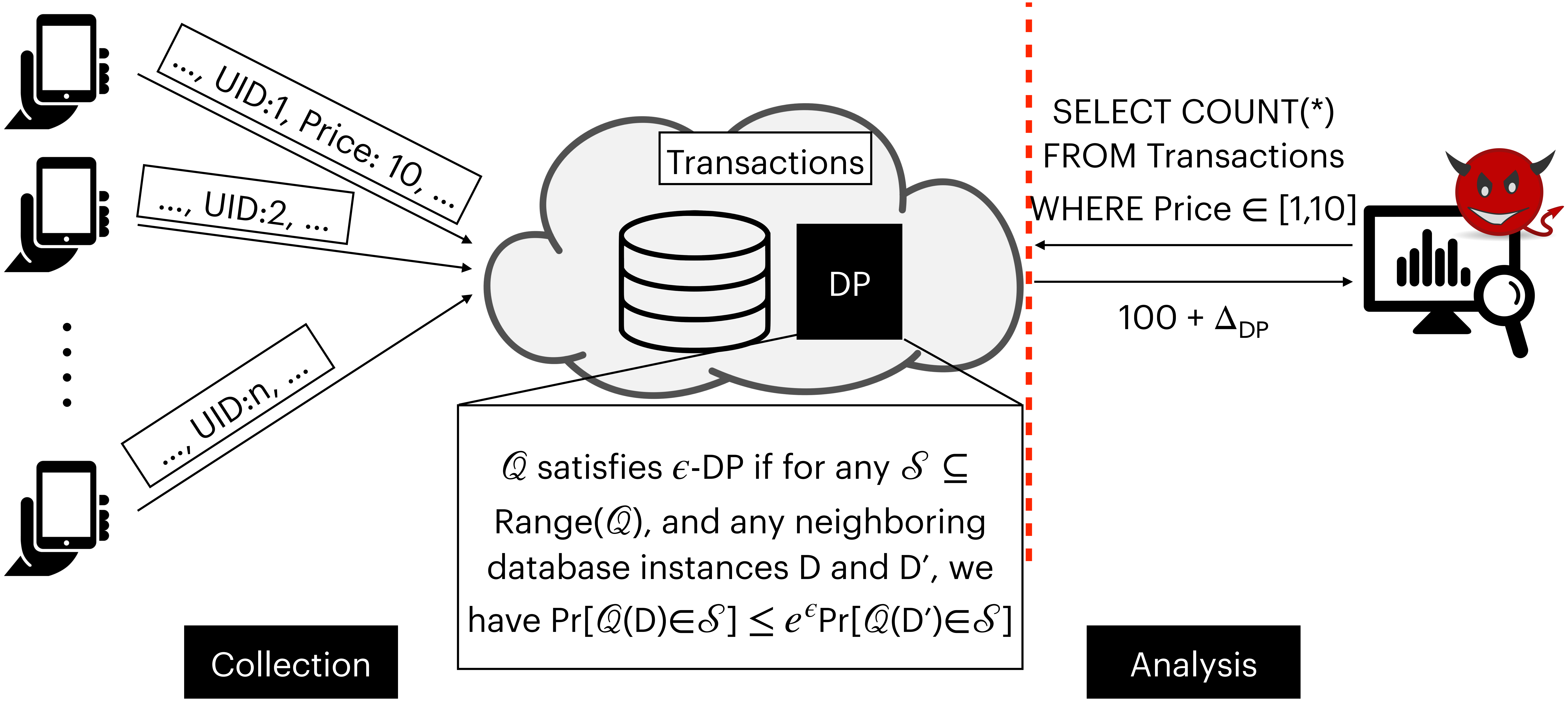
Data Collection and Analysis in the Cloud



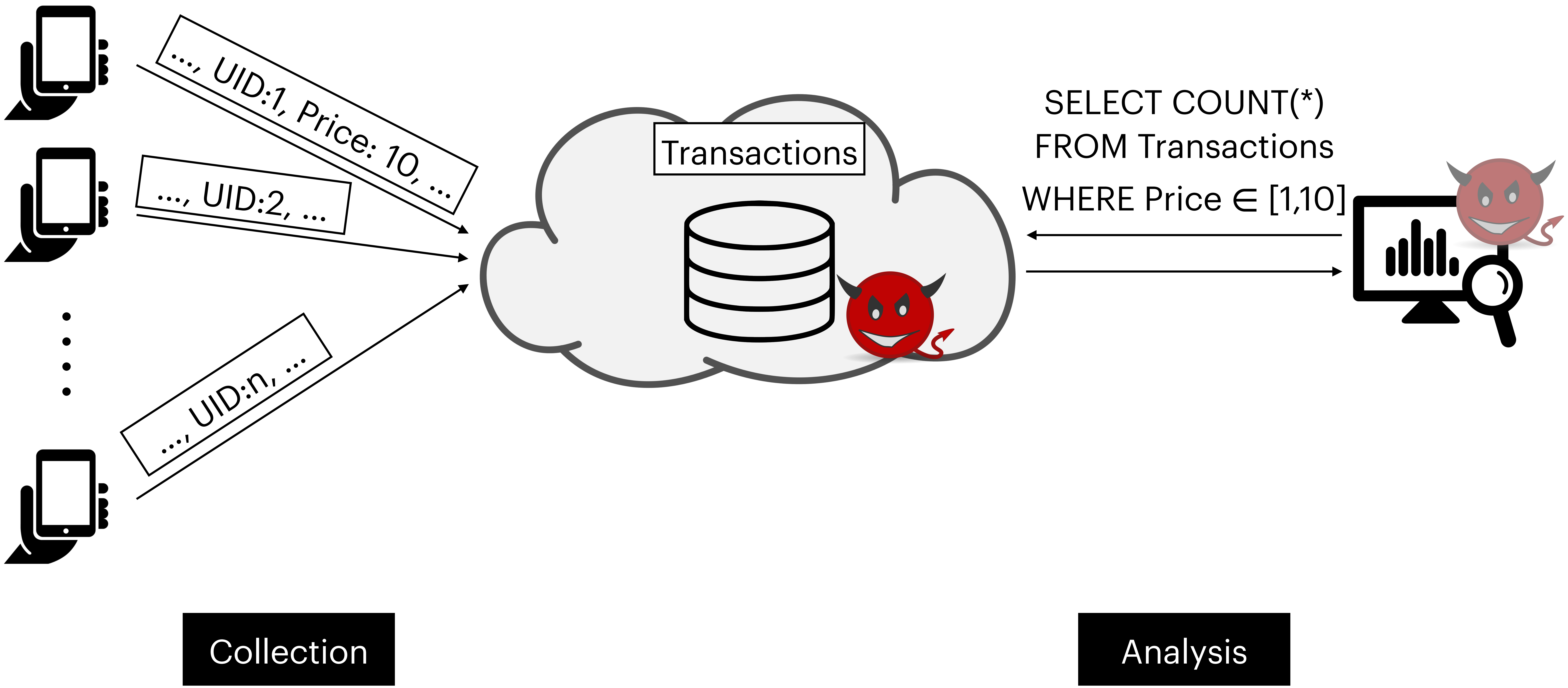
Data Collection and Analysis in the Cloud



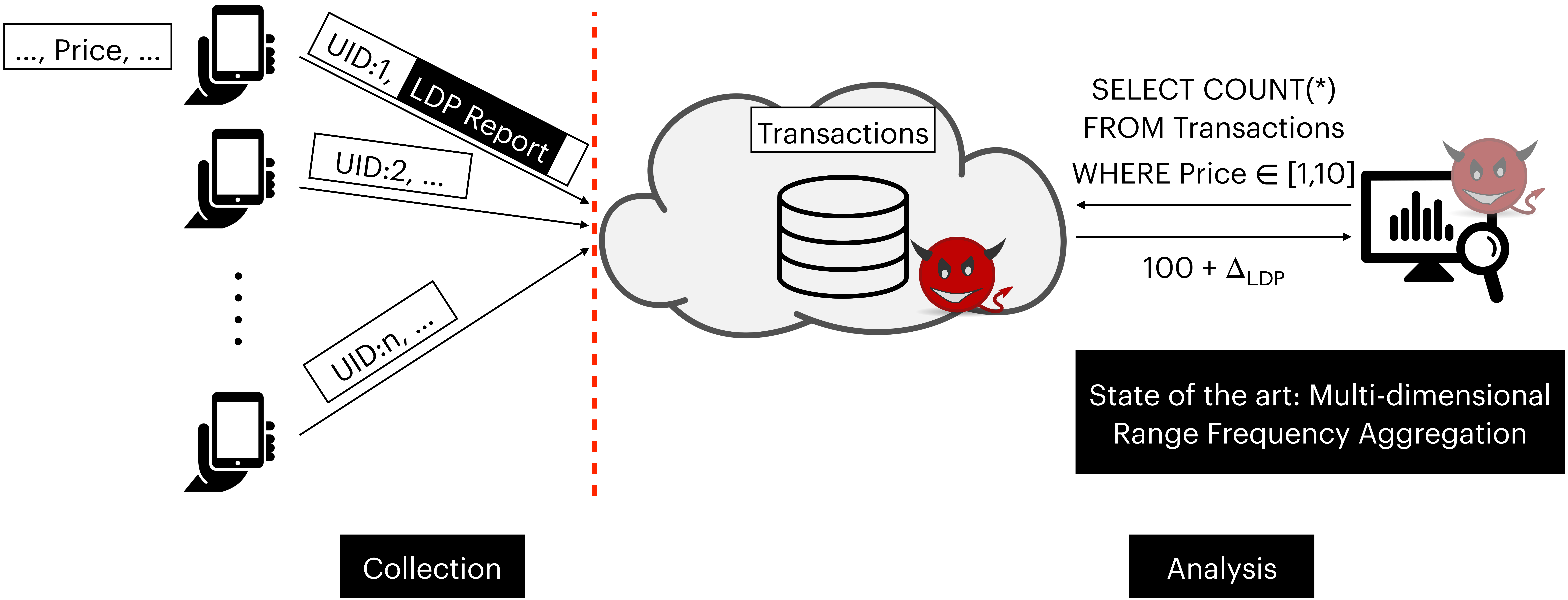
Data Collection and Analysis in the Cloud



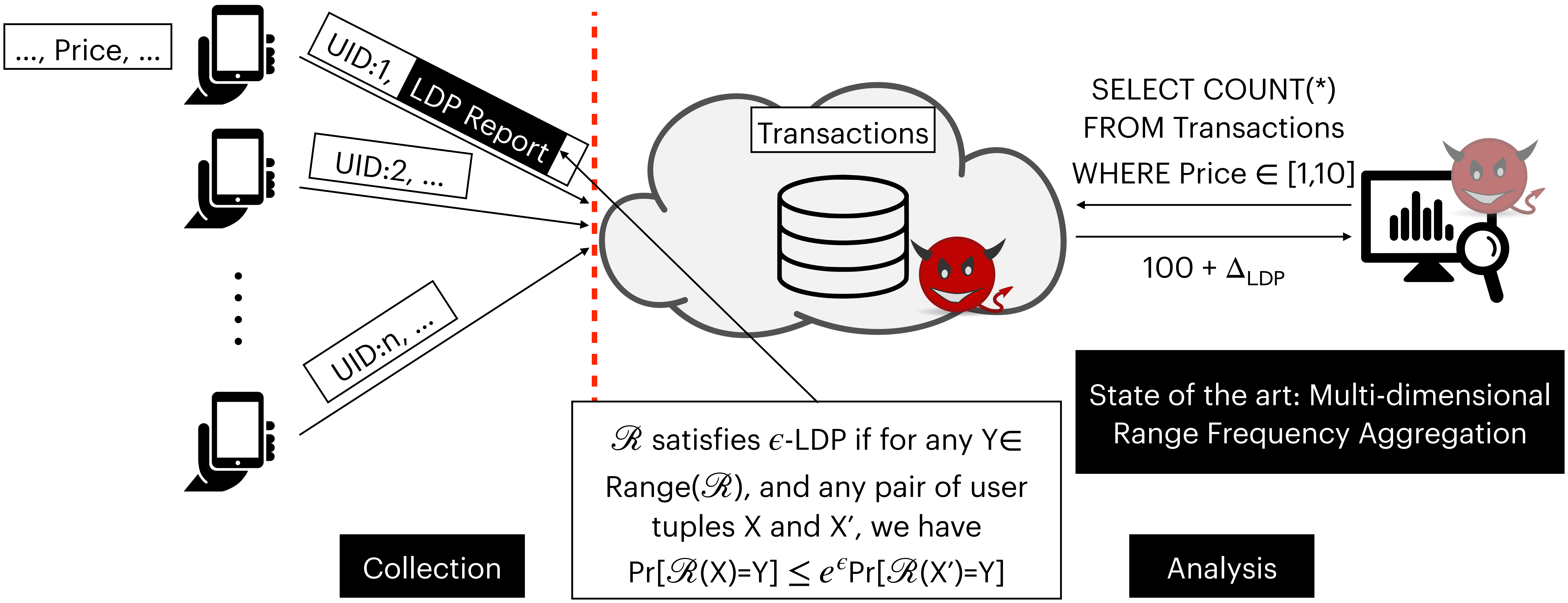
Data Collection and Analysis in the Cloud



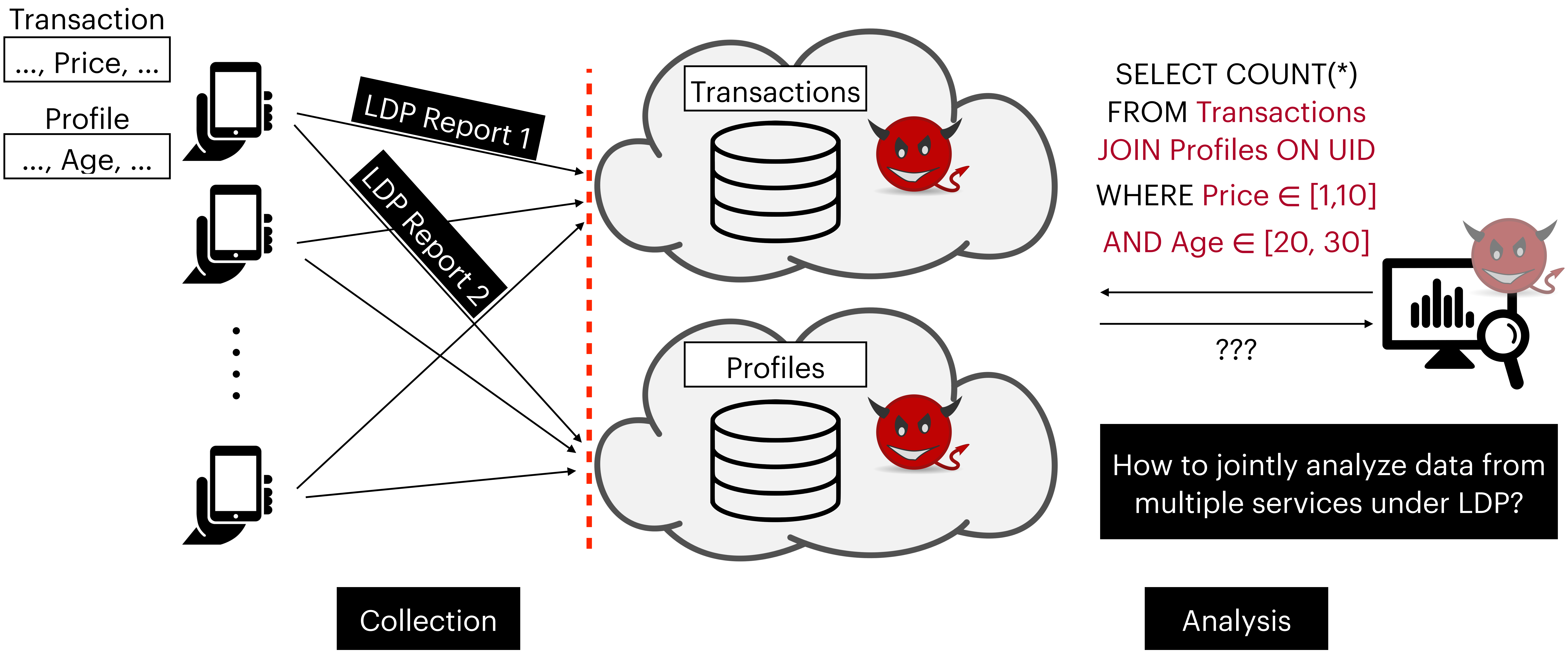
Data Collection and Analysis in the Cloud



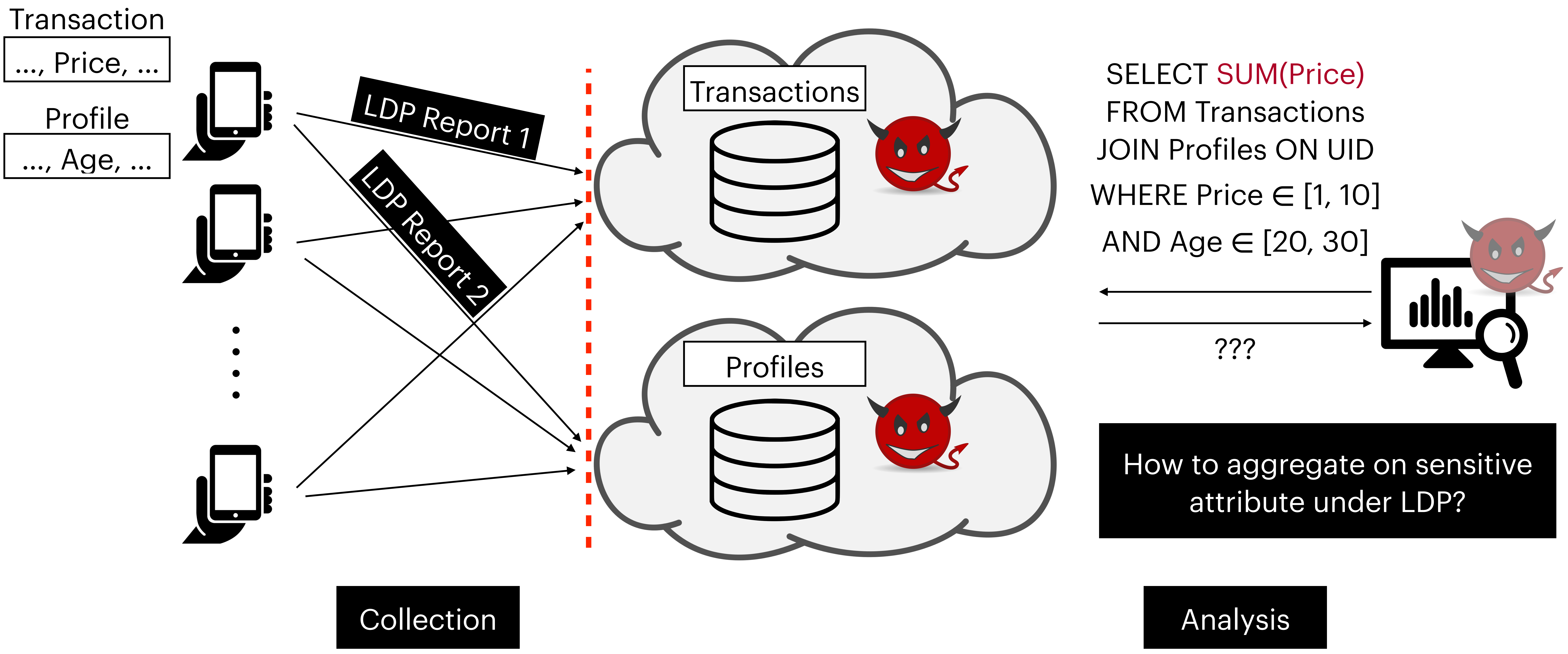
Data Collection and Analysis in the Cloud



Data Collection and Analysis in the Cloud - This Work



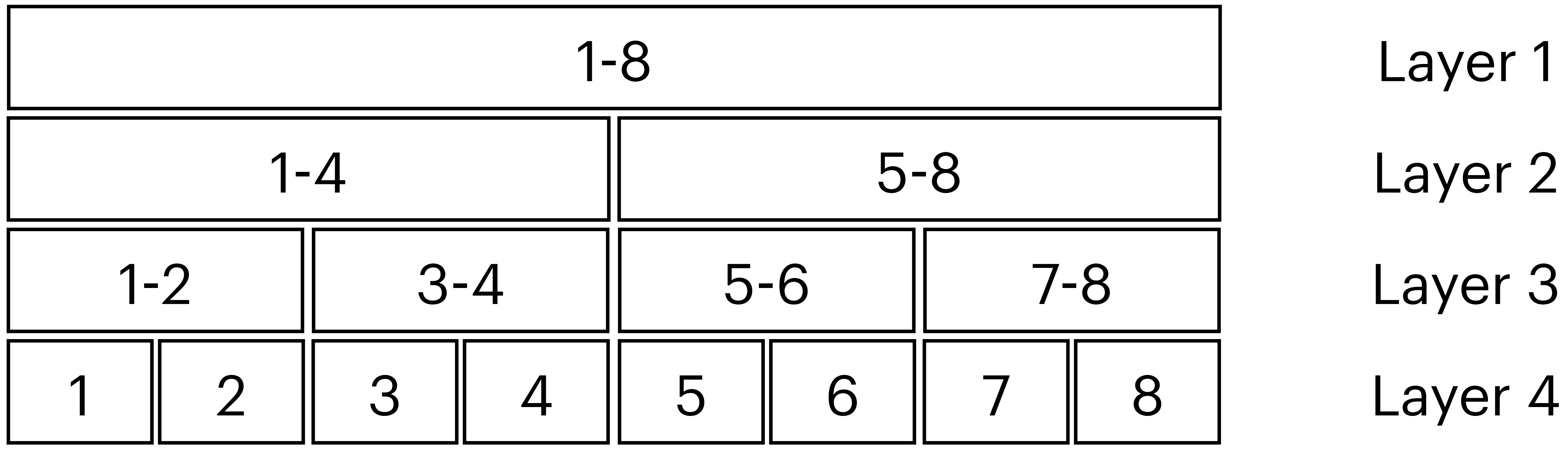
Data Collection and Analysis in the Cloud - This Work



Recap: Hierarchical Interval Optimized (HIO)

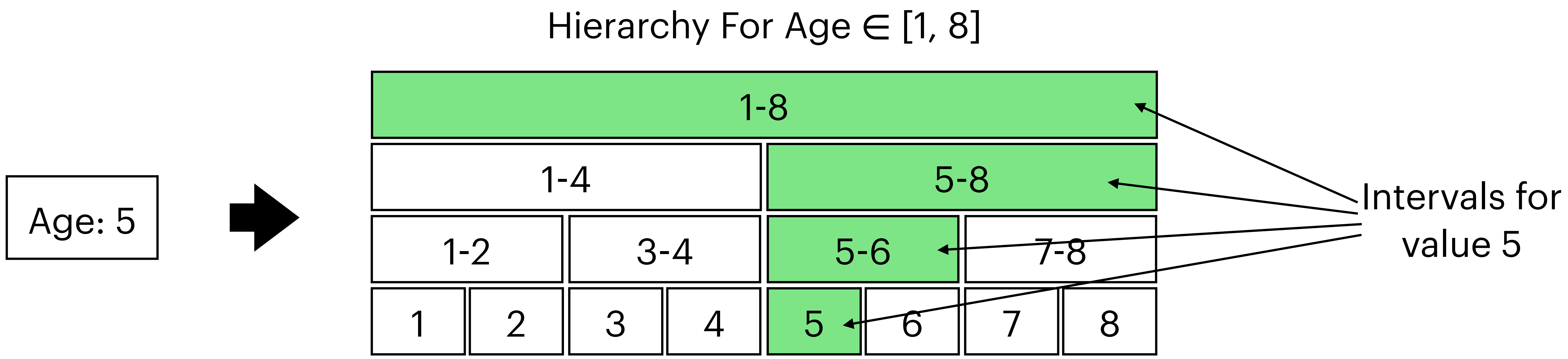
Collection

Hierarchy For Age ∈ [1, 8]

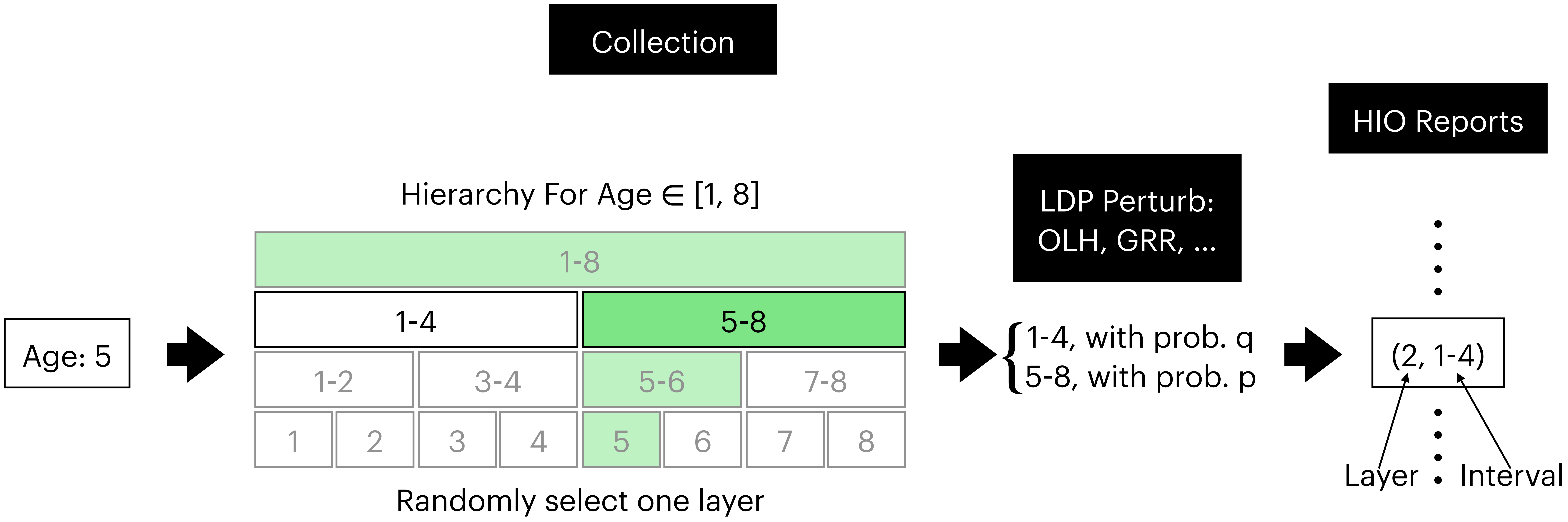


Recap: Multi-dimensional Range Frequency Oracles using Hierarchical Interval Optimized (HIO)

Collection

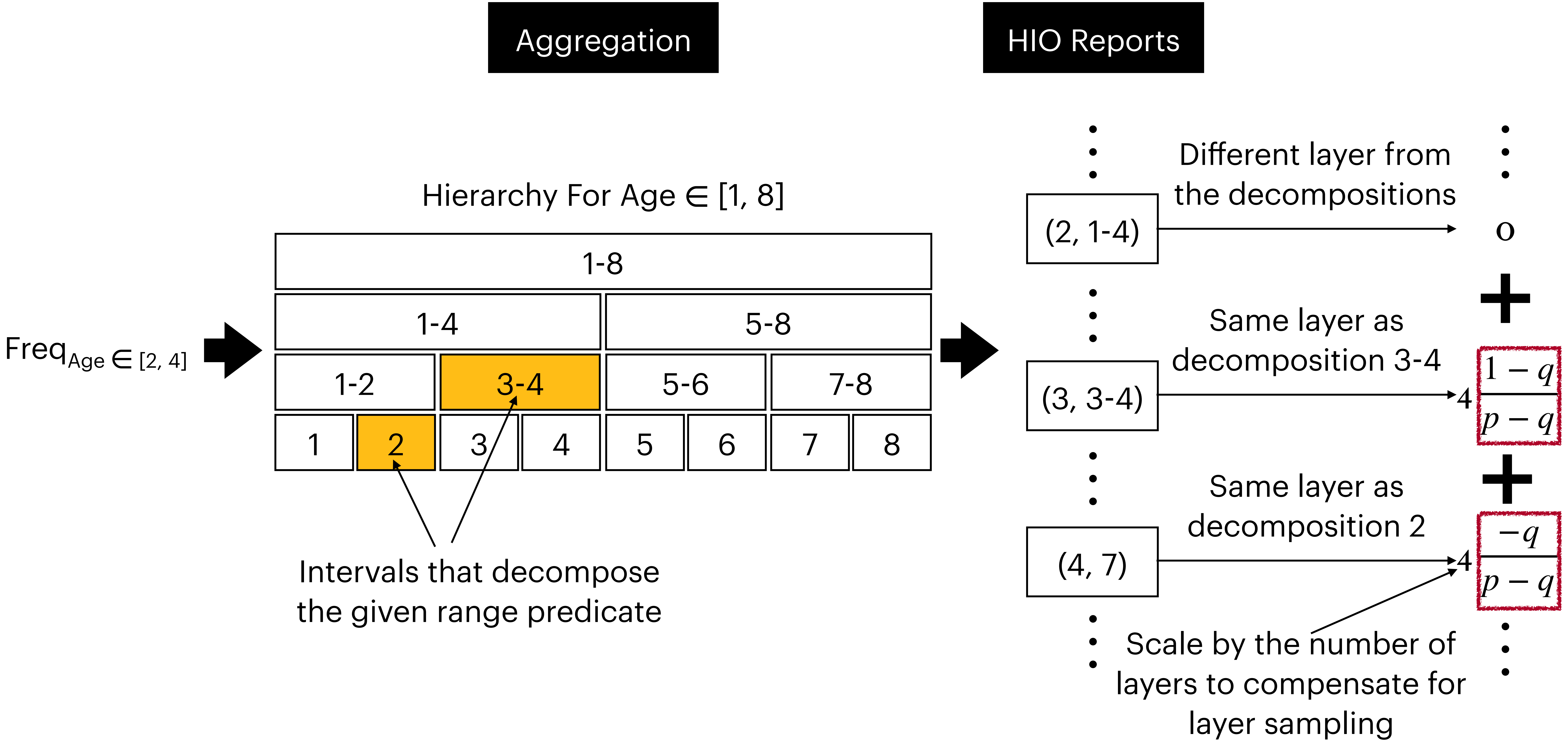


Recap: Multi-dimensional Range Frequency Oracles using Hierarchical Interval Optimized (HIO)

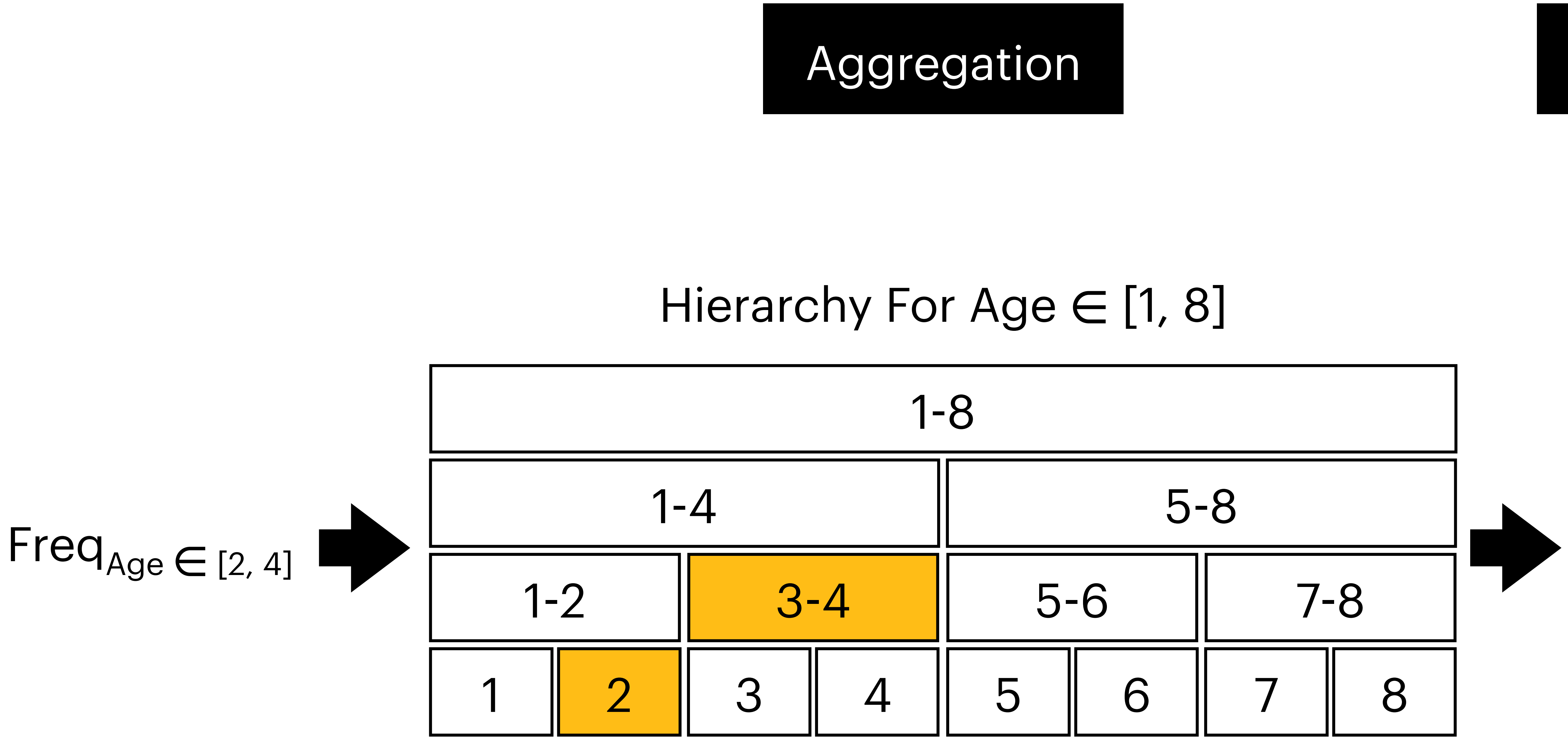


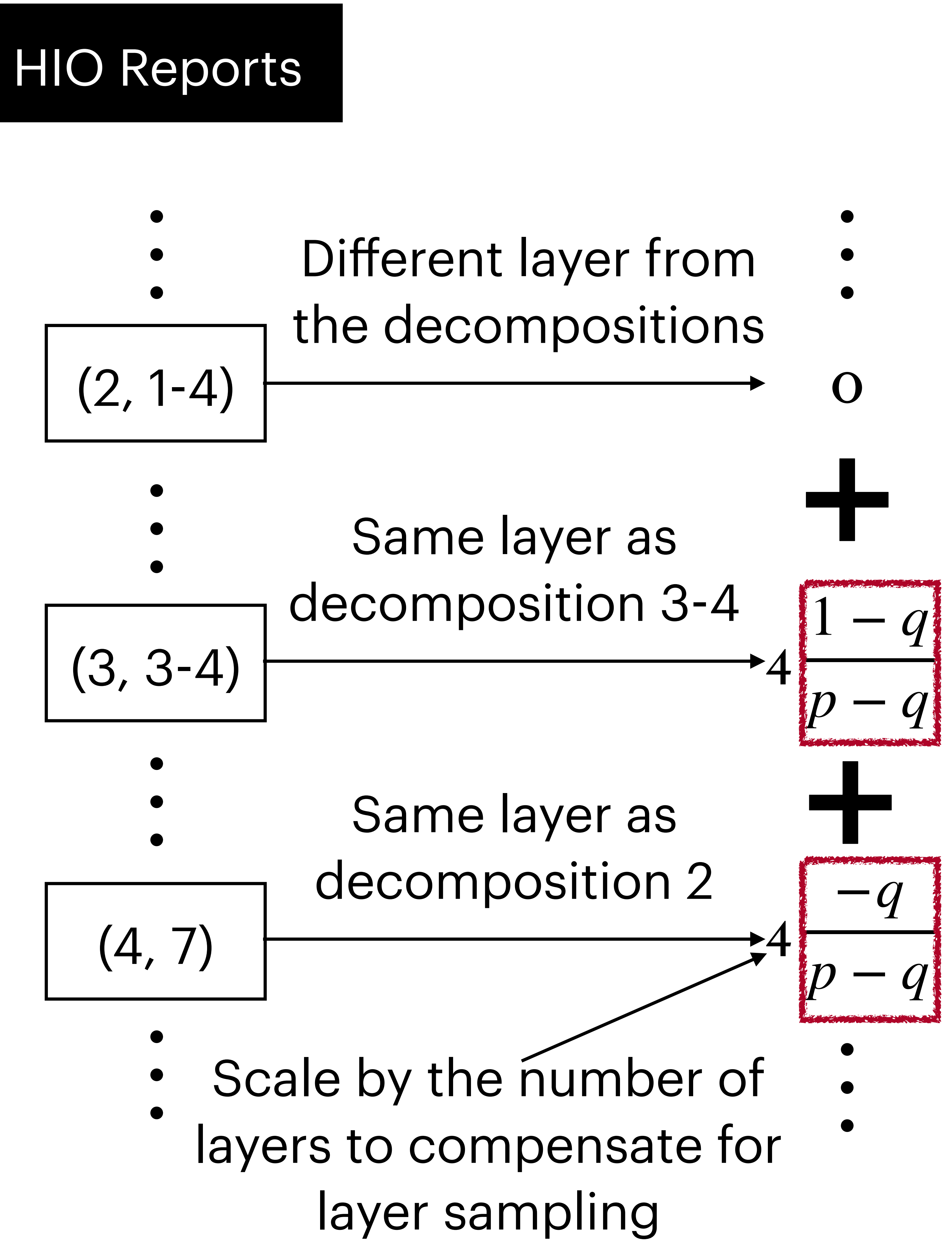
For GRR, $p = \frac{e^{\epsilon}}{e^{\epsilon} + m - 1}$, $q = \frac{1}{e^{\epsilon} + m - 1}$, where m is the cardinality of the domain

Recap: Multi-dimensional Range Frequency Oracles using Hierarchical Interval Optimized (HIO)



Recap: Multi-dimensional Range Frequency Oracles using Hierarchical Interval Optimized (HIO)





Mean Squared Error: $O\left(\frac{n \log^{d+d_q} m}{\epsilon^2}\right)$, where n is the number of users, d and d_q are the numbers of attributes in the schema and predicate

Data Collection with Two Independent Services using HIO

Data Collection with Two Independent Services using HIO

$\text{Freq}_{\text{Age} \in [5, 8]} = \dfrac{-q}{p-q}$

$\text{Freq}_{\text{Price} \in [1, 2]} = \dfrac{1-q}{p-q}$

Data Collection with Two Independent Services

$\mathrm{Freq}_{\mathrm{Age}\in[5,\,8]} = \dfrac{-q}{p-q}$

$\mathrm{Freq}_{\mathrm{Age}\in[5,\,8]\,\wedge\,\mathrm{Price}\in[1,\,2]}?$

$\mathrm{Freq}_{\mathrm{Price}\in[1,\,2]} = \dfrac{1-q}{p-q}$

Data Collection with Two Independent Services

$\text{Freq}_{\text{Age}\in[5,\,8]} = \dfrac{-q}{p-q}$

$\text{Freq}_{\text{Age}\in[5,\,8]\,\wedge\,\text{Price}\in[1,\,2]}?$

$\text{Freq}_{\text{Price}\in[1,\,2]} = \dfrac{1-q}{p-q}$

State Inversion with Two Independent Hierarchies

Target frequency estimation

$\mathbf{X[11] := Freq_{X1 \in [5,8] \wedge X2 \in [1,2]}}$

$X[10] := Freq_{X1 \in [5,8] \wedge X2 \notin [1,2]}$

$X[01] := Freq_{X1 \notin [5,8] \wedge X2 \in [1,2]}$

$X[00] := Freq_{X1 \notin [5,8] \wedge X2 \notin [1,2]}$

$Y[11] := Freq_{Y1 \in [5,8] \wedge Y2 \in [1,2]}$

$Y[10] := Freq_{Y1 \in [5,8] \wedge Y2 \notin [1,2]}$

$Y[01] := Freq_{Y1 \notin [5,8] \wedge Y2 \in [1,2]}$

$Y[00] := Freq_{Y1 \notin [5,8] \wedge Y2 \notin [1,2]}$

State Inversion with Two Independent Hierarchies

$\mathbf{X[11] := Freq_{X1 \in [5,8] \wedge X2 \in [1,2]}}$

$X[10] := Freq_{X1 \in [5,8] \wedge X2 \notin [1,2]}$

$X[01] := Freq_{X1 \notin [5,8] \wedge X2 \in [1,2]}$

$X[00] := Freq_{X1 \notin [5,8] \wedge X2 \notin [1,2]}$

$Y[11] := Freq_{Y1 \in [5,8] \wedge Y2 \in [1,2]}$

$Y[10] := Freq_{Y1 \in [5,8] \wedge Y2 \notin [1,2]}$

$Y[01] := Freq_{Y1 \notin [5,8] \wedge Y2 \in [1,2]}$

$Y[00] := Freq_{Y1 \notin [5,8] \wedge Y2 \notin [1,2]}$

$$\mathbf{E}\begin{bmatrix} Y[11] \\ Y[10] \\ Y[01] \\ Y[00] \end{bmatrix} = \begin{bmatrix} Pr[Y11|X11] & Pr[Y11|X10] & Pr[Y11|X01] & Pr[Y11|X00] \\ Pr[Y10|X11] & Pr[Y10|X10] & Pr[Y10|X01] & Pr[Y10|X00] \\ Pr[Y01|X11] & Pr[Y01|X10] & Pr[Y01|X01] & Pr[Y01|X00] \\ Pr[Y00|X11] & Pr[Y00|X10] & Pr[Y00|X01] & Pr[Y00|X00] \end{bmatrix} * \begin{bmatrix} \mathbf{X[11]} \\ X[10] \\ X[01] \\ X[00] \end{bmatrix}$$

State Inversion with Two Independent Hierarchies

$X[11] := \mathrm{Freq}_{X1\in[5,8]\,\wedge\,X2\in[1,2]}$

$X[10] := \mathrm{Freq}_{X1\in[5,8]\,\wedge\,X2\notin[1,2]}$

$X[01] := \mathrm{Freq}_{X1\notin[5,8]\,\wedge\,X2\in[1,2]}$

$X[00] := \mathrm{Freq}_{X1\notin[5,8]\,\wedge\,X2\notin[1,2]}$

$Y[11] := \mathrm{Freq}_{Y1\in[5,8]\,\wedge\,Y2\in[1,2]}$

$Y[10] := \mathrm{Freq}_{Y1\in[5,8]\,\wedge\,Y2\notin[1,2]}$

$Y[01] := \mathrm{Freq}_{Y1\notin[5,8]\,\wedge\,Y2\in[1,2]}$

$Y[00] := \mathrm{Freq}_{Y1\notin[5,8]\,\wedge\,Y2\notin[1,2]}$

$$\mathbf{E}\underbrace{\begin{bmatrix} Y[11] \\ Y[10] \\ Y[01] \\ Y[00] \end{bmatrix}}_{\mathbf{Y}} = \underbrace{\begin{bmatrix} \Pr[Y11|X11] & \Pr[Y11|X10] & \Pr[Y11|X01] & \Pr[Y11|X00] \\ \Pr[Y10|X11] & \Pr[Y10|X10] & \Pr[Y10|X01] & \Pr[Y10|X00] \\ \Pr[Y01|X11] & \Pr[Y01|X10] & \Pr[Y01|X01] & \Pr[Y01|X00] \\ \Pr[Y00|X11] & \Pr[Y00|X10] & \Pr[Y00|X01] & \Pr[Y00|X00] \end{bmatrix}}_{\mathbf{M}} * \underbrace{\begin{bmatrix} \mathbf{X[11]} \\ X[10] \\ X[01] \\ X[00] \end{bmatrix}}_{\mathbf{X}} \Rightarrow \mathbf{X} \leftarrow \mathbf{M}^{-1}\mathbf{Y}$$

Unbiased estimation

- How to calculate Pr[Y11|X11]?

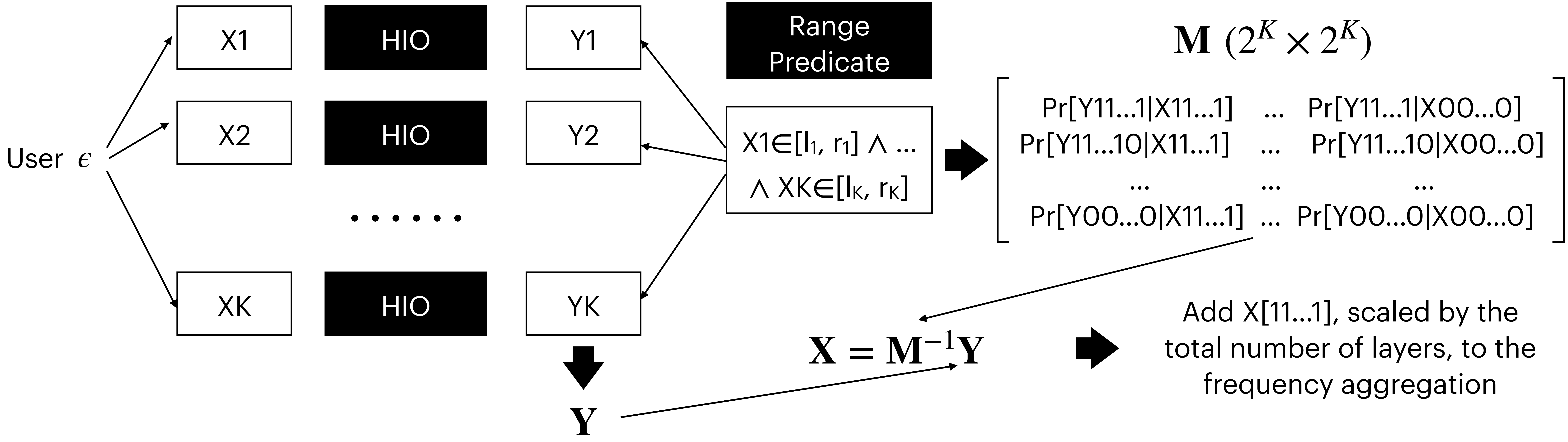
  Because X1 and X2 are independently perturbed into Y1 and Y2, Pr[Y11|X11] = Pr[Y1∈[2, 4]|X1∈[2, 4]]*Pr[Y2∈[1, 4]|X2∈[1, 4]] = p*p. We can derive other conditional prob. similarly

Joint Frequency Oracles with Two Independent Services

$$\text{Freq}_{\text{Age}\in[5,\,8]\,\wedge\,\text{Price}\in[1,\,2]} \leftarrow (\mathbf{M}^{-1}\mathbf{Y})[11\ldots1]$$

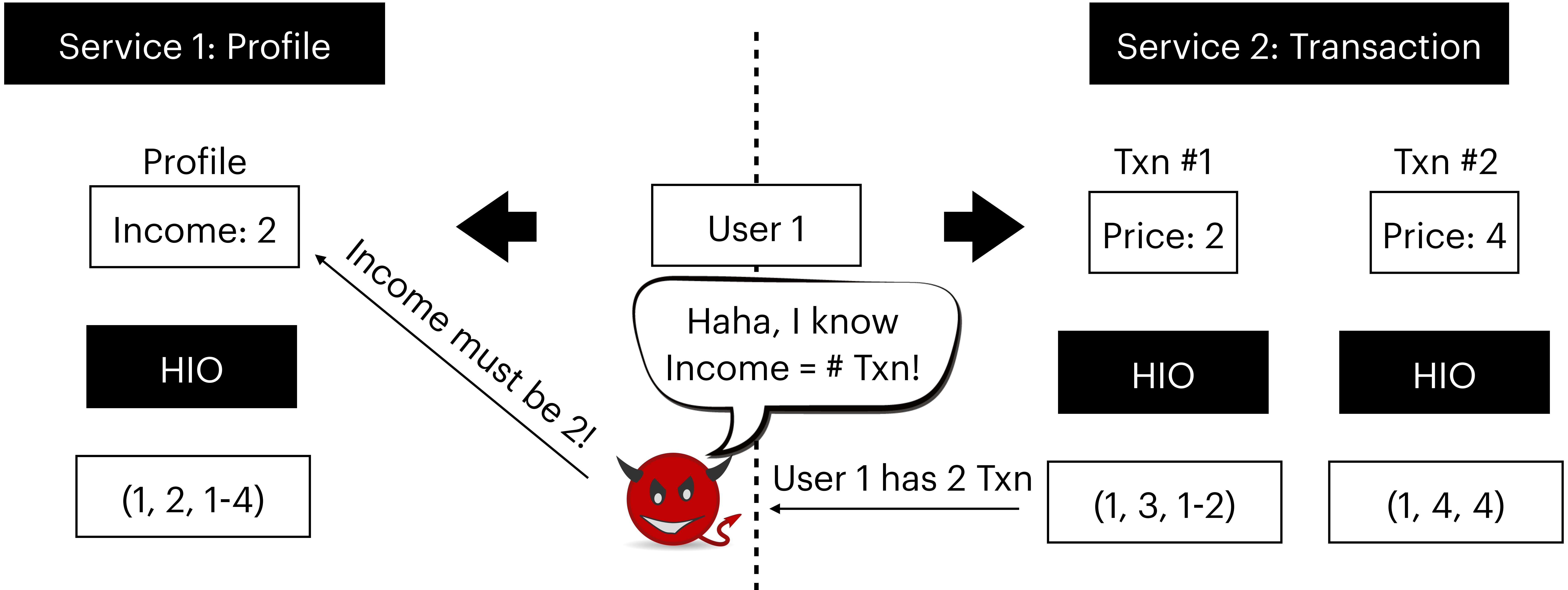
Joint Frequency Oracles with Two Independent Services

General Joint Frequency Oracles with K Services (HIO-JOIN)



- Mean Squared Error: $O\left(\frac{n \log^{K(d+d_q)} m}{(\epsilon/K)^{2K}}\right)$, where K is the number of services, d and d_q are the numbers of attributes in the schema and predicate for each service
- In fact, it can handle range predicate on attributes of arbitrary subset of the K services

1-Many join: Primary-Foreign-Key JOIN with Two Services



Frequency-Based Attack: Infer sensitive information based on the leaked number of tuples of a user collected by each service

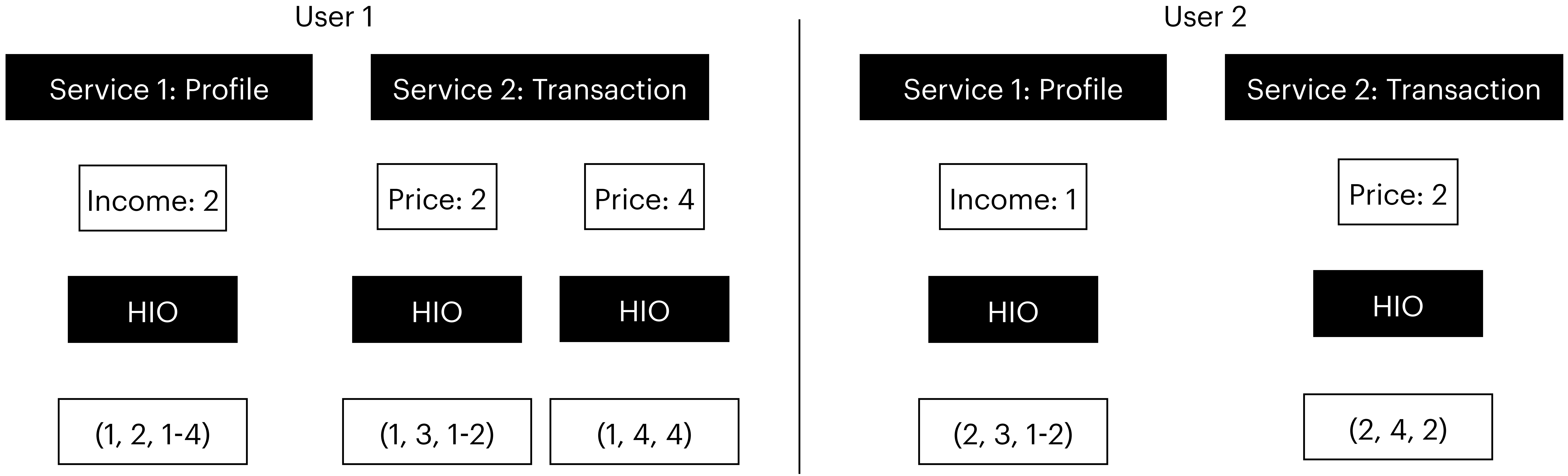
- The information that user 1 made 2 transactions with service 2 is leaked at join
- If such information is correlated with the sensitive information, say, the income, attacker can infer the income of the user based on the number of transactions she made

User-Level LDP (uLDP)

ϵ-uLDP: Intuitively, for any two users, the difference between the distributions of their collected data is bounded by e^{ϵ}

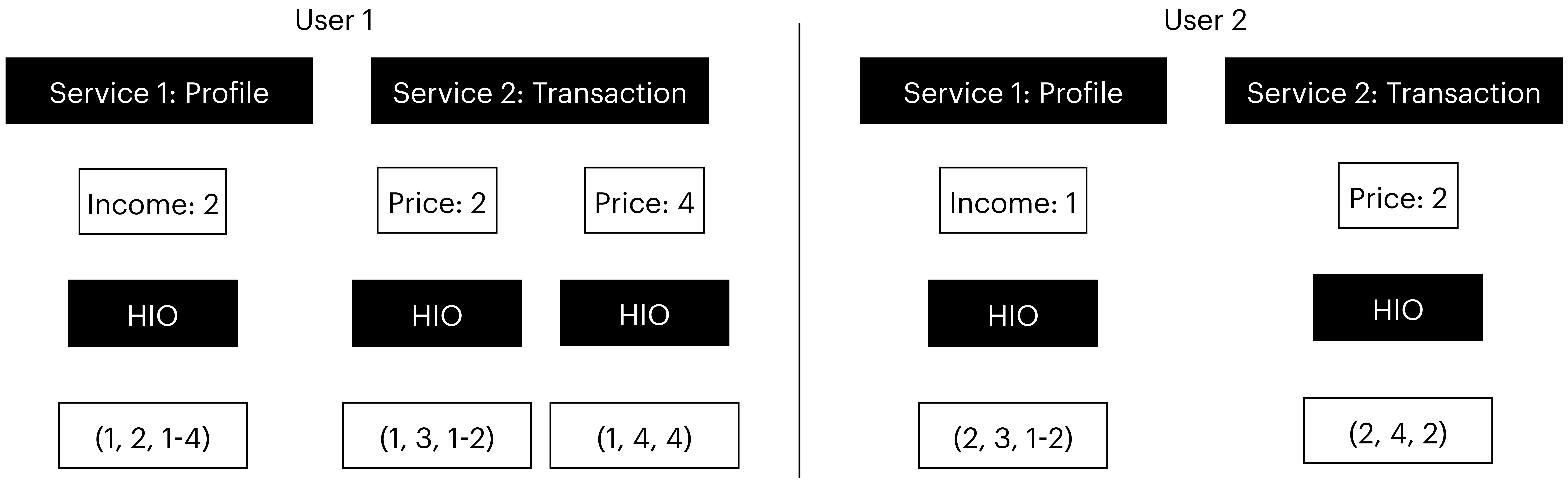
User-Level LDP (uLDP)

ϵ-uLDP: Intuitively, for any two users, the difference between the distributions of their collected data is bounded by e^{ϵ}



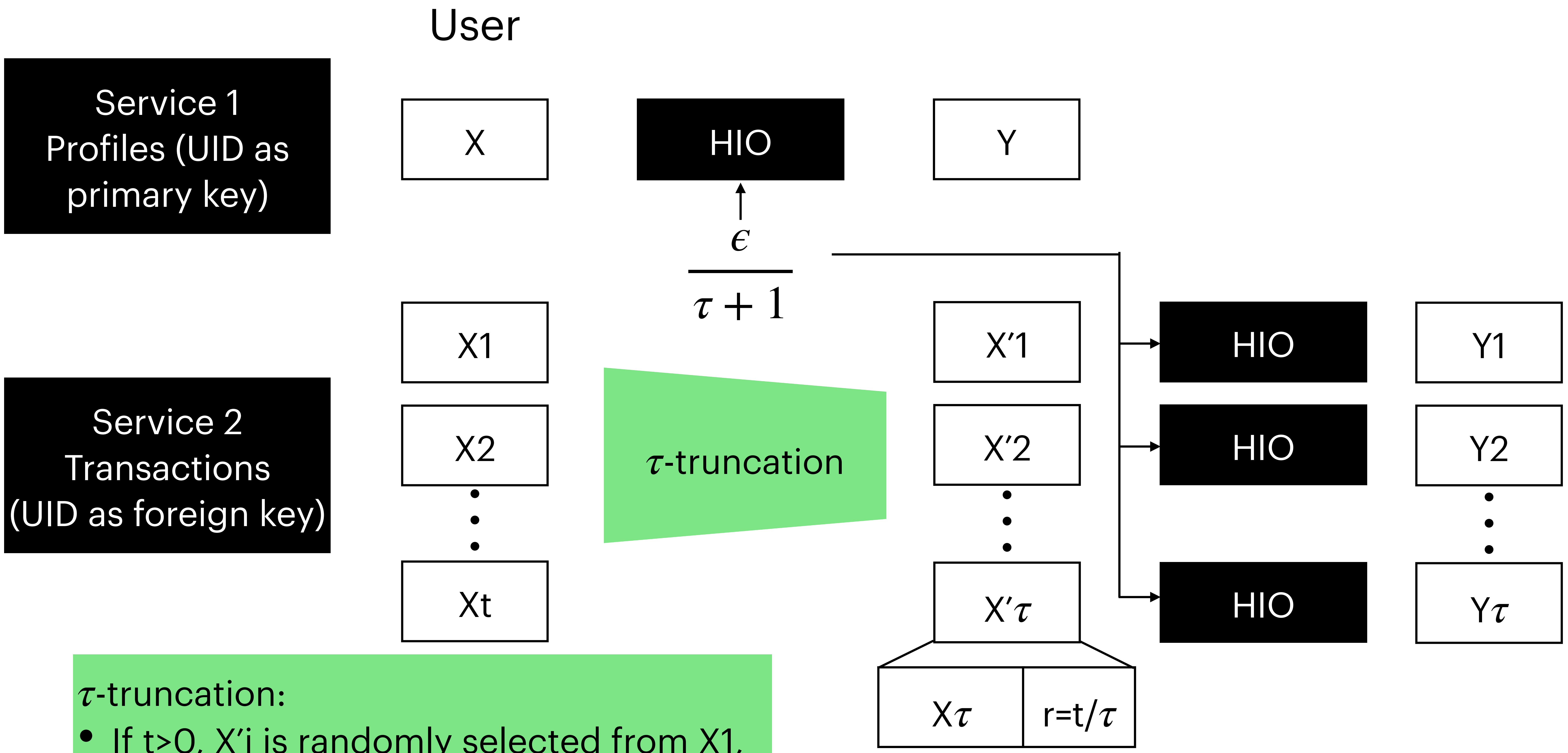
User-Level LDP (uLDP)

ϵ-uLDP: Intuitively, for any two users, the difference between the distributions of their collected data is bounded by e^{ϵ}



Impossible to have the same collected data for user 1 and 2! => ∞-uLDP => no privacy at all!

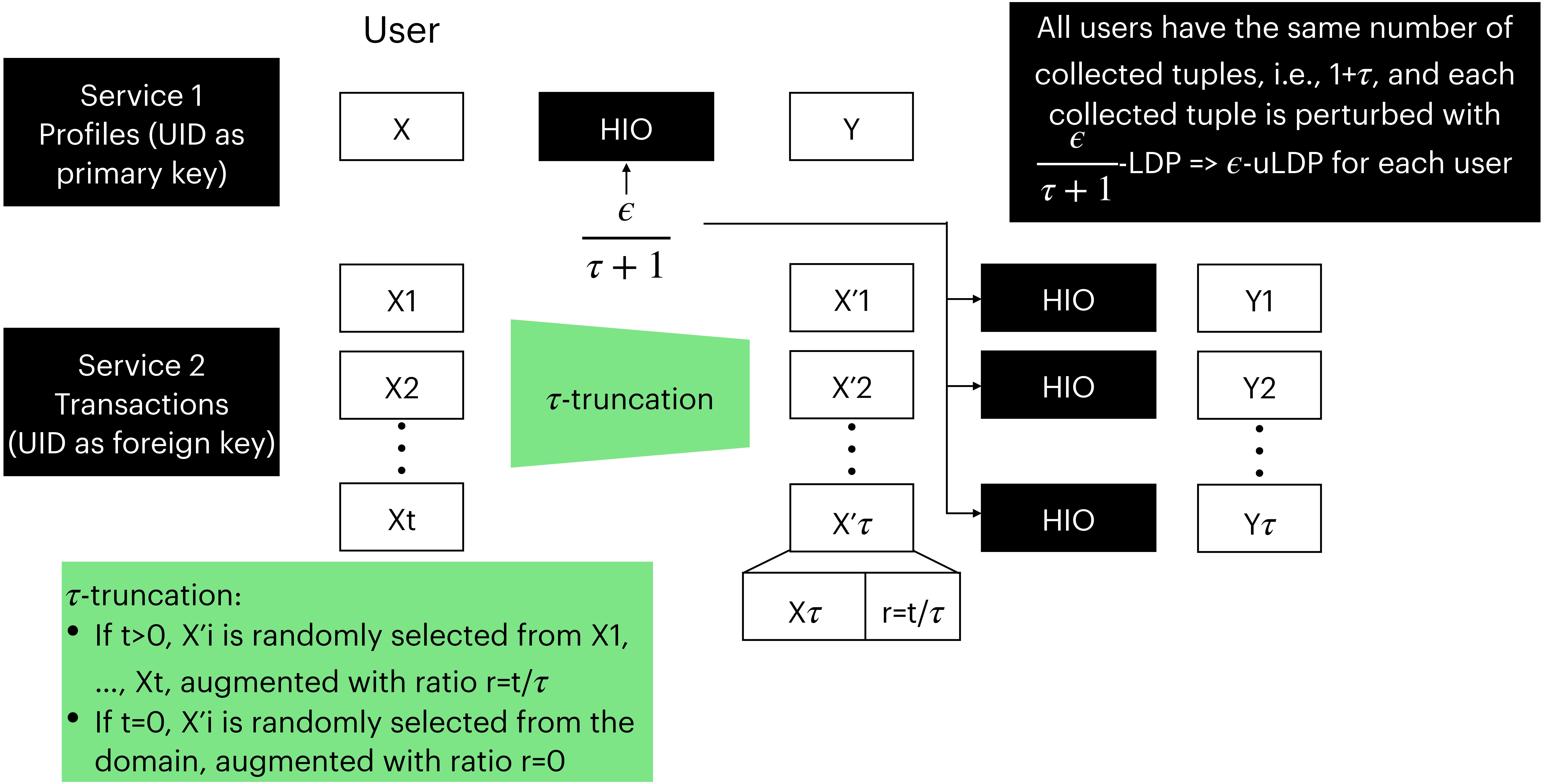
1-Many JOIN under ϵ-uLDP



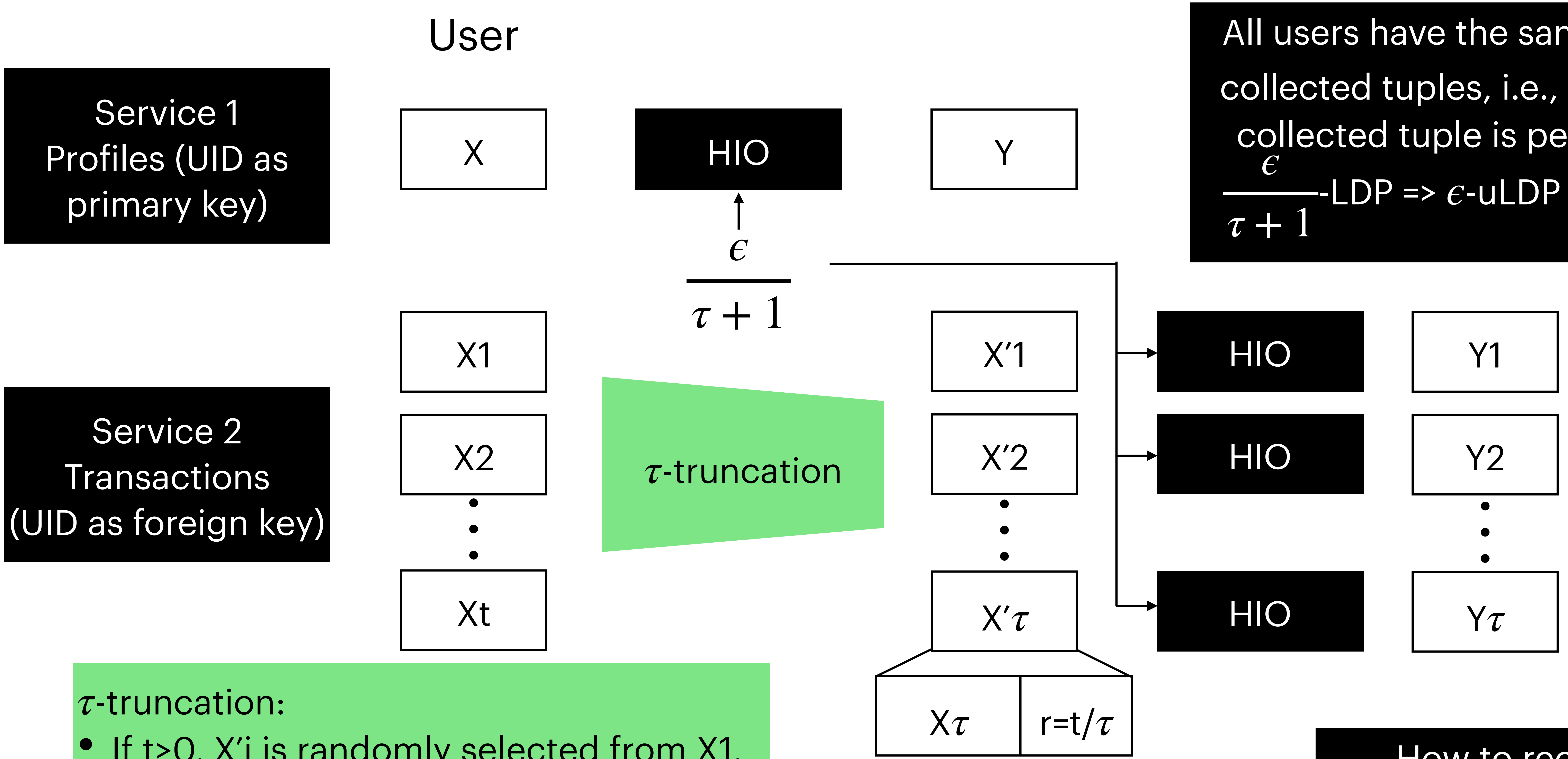
τ-truncation:

- If t>0, X'i is randomly selected from X1, …, Xt, augmented with ratio r=t/τ
- If t=0, X'i is randomly selected from the domain, augmented with ratio r=0

1-Many JOIN under ϵ-uLDP



1-Many JOIN under ϵ-uLDP



All users have the same number of collected tuples, i.e., 1+τ, and each collected tuple is perturbed with $\frac{\epsilon}{\tau+1}$-LDP => ϵ-uLDP for each user

τ-truncation:

- If t>0, X'i is randomly selected from X1, …, Xt, augmented with ratio r=t/τ
- If t=0, X'i is randomly selected from the domain, augmented with ratio r=0

How to recover the unbiased aggregations from the truncated tuples?

Sensitive Weight Frequency Oracles (SWFO)

- So far, we can handle frequency oracles on the collected data with range predicate over arbitrary subset of attributes from the multiple services
- It is necessary to further support oracles for the aggregated frequency, weighted by some sensitive attribute of each tuple. We call such oracles the *sensitive weight frequency oracles*:
  - **Attribute Aggregation**

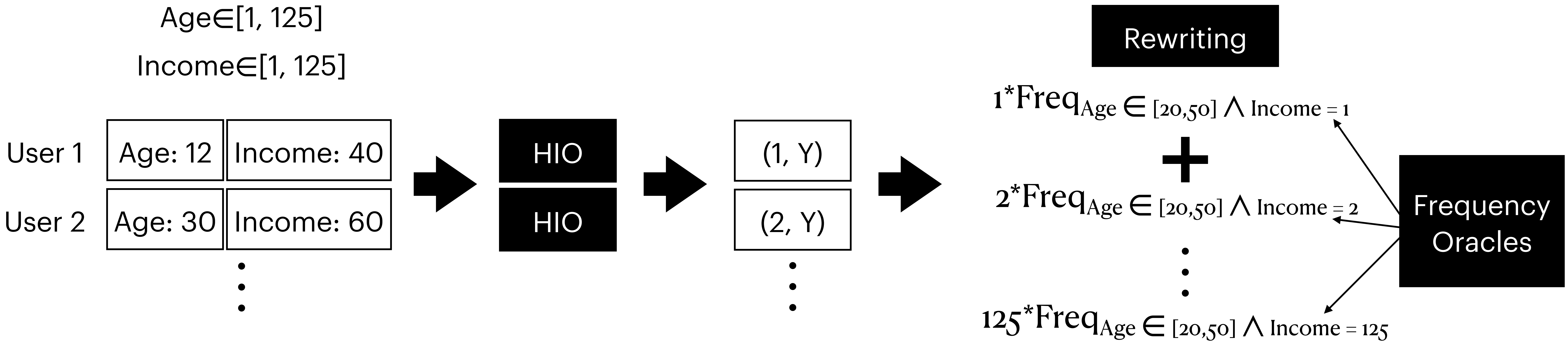
    SELECT SUM(Income) FROM Profiles WHERE Age $\in$ [20,50]
  - **Recover Unbiased Aggregation with τ-Truncation** (weighing each tuple by r=t/τ)

    SELECT SUM(r) FROM Profiles JOIN Transactions ON UID WHERE Age $\in$ [20,50]

    SELECT SUM(r*Price) FROM Profiles JOIN Transactions ON UID WHERE Age $\in$ [20,50]
  - Previous works leave such aggregation analysis as open problems

(Joint) Frequency Oracles to SWFO (Straw-man)

SELECT SUM(Income) FROM Profiles WHERE Age ∈ [20,50]



Mean Squared Error: $O\left(\boxed{\sum_{v \in A} v^2} \frac{n \log^{d+d_q} m}{\epsilon^2}\right)$

The multiplicative factor of sum of squares of the domain values in the error bound is prohibitive for aggregation on large domains

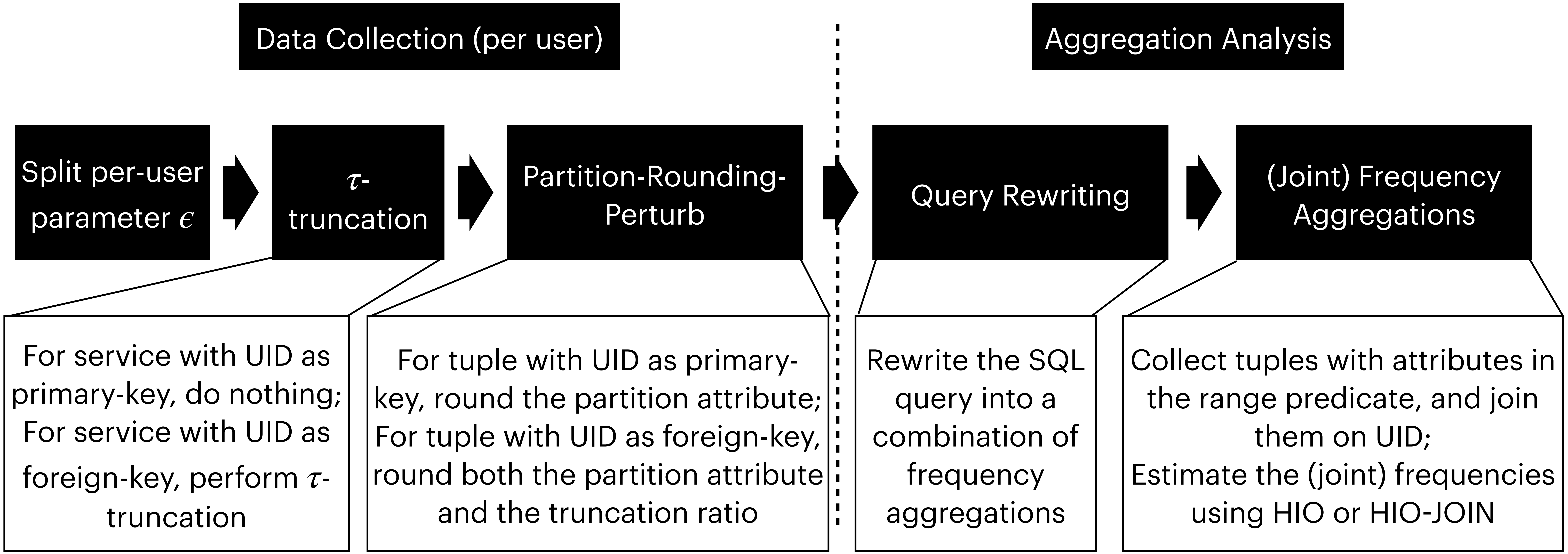
Partition-Rounding-Perturb Framework for SWFO

SELECT SUM(Income) FROM Profiles WHERE Age ∈ [20,50]

Mean Squared Error: $O\left(\frac{\boxed{2(A_{\min}^2 + A_{\max}^2)d} \cdot n \log^{d+d_q} m}{\epsilon^2}\right)$

The multiplicative factor is linear to the number of attributes times the sum of squares of the min/max

End-to-End Data Collection and Analysis Pipeline



- Query independent - can handle arbitrary frequency and attribute aggregation with range predicate over any subset of attributes
- Minimal coordination among services: i) value of τ; and ii) the total number of tuples to be collected for each user

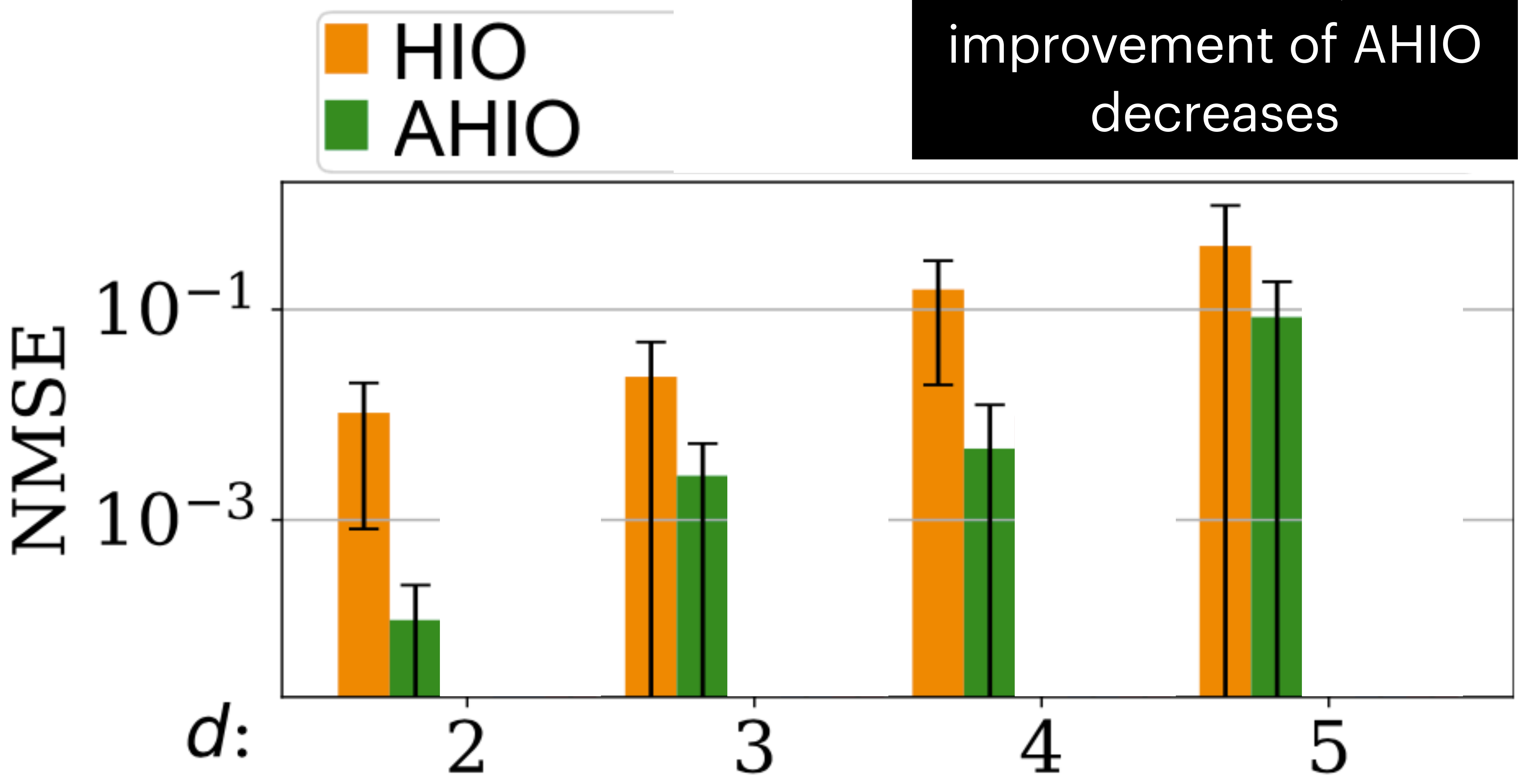
Experimental Results

- We setup a single-node SparkSQL cluster for the experiments, and register the LDP perturb and aggregation estimation primitives as UDFs of SparkSQL
- Datasets
  - SYN-1: Single-table synthetic data , with 1M records and configurable number of attributes d and attribute cardinality m
  - SYN-2: Two-table synthetic data, with two join configurations: 1-1; and 1-many with join degree in range [1, 10]
- We focus on the utilities for COUNT, SUM and AVG aggregations of the following schemes
  - AHIO: Partition-Rounding-Perturb with the rounding value stored in separate attribute
  - HIO-JOIN: no truncation for 1-1 join; with τ-truncation for 1-many join

Attribute Aggregation

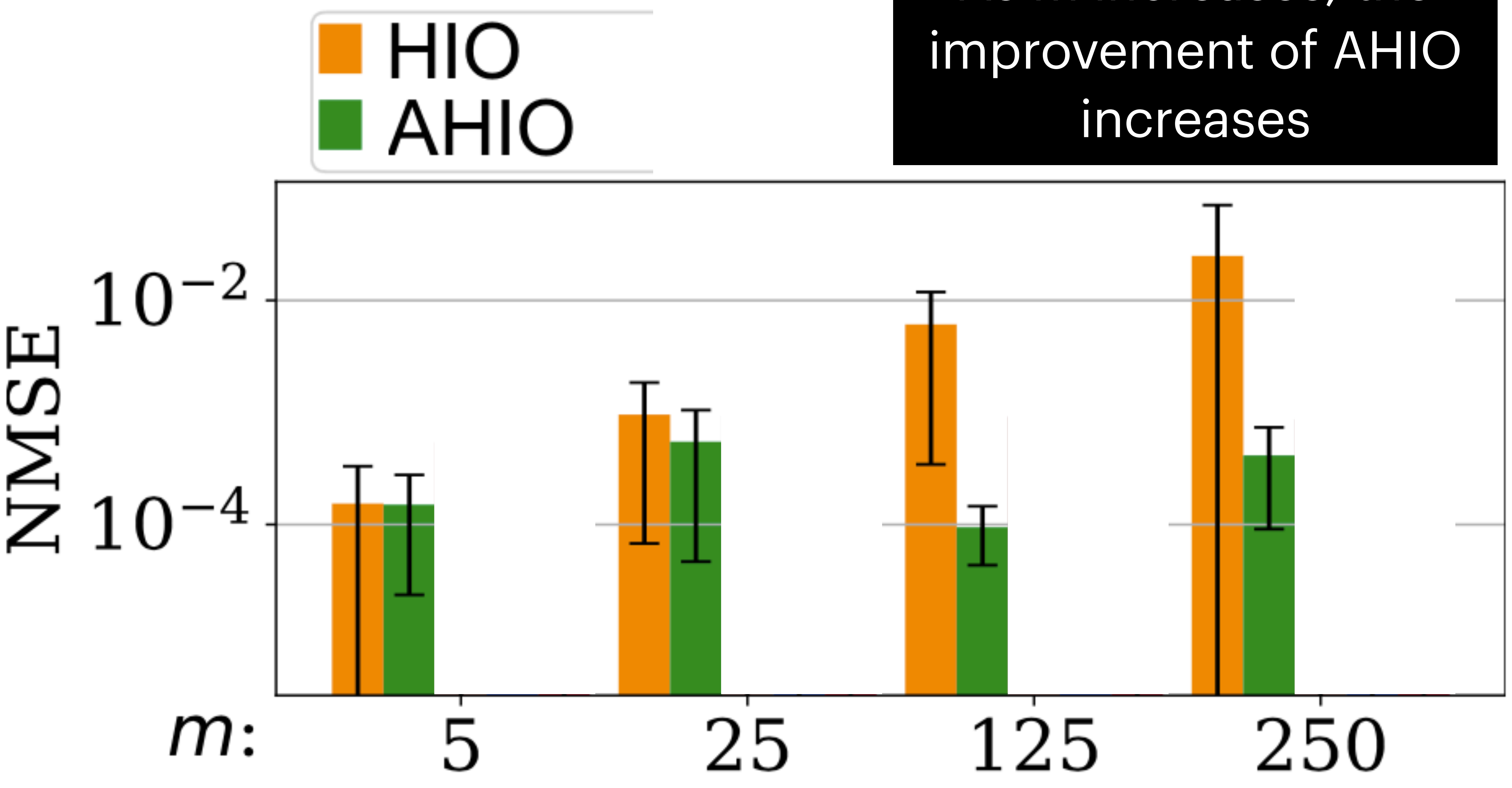
SELECT SUM(A) FROM SYN-1 WHERE C

As d increases, the improvement of AHIO decreases



ϵ=2, m=125, d_q=1

As m increases, the improvement of AHIO increases



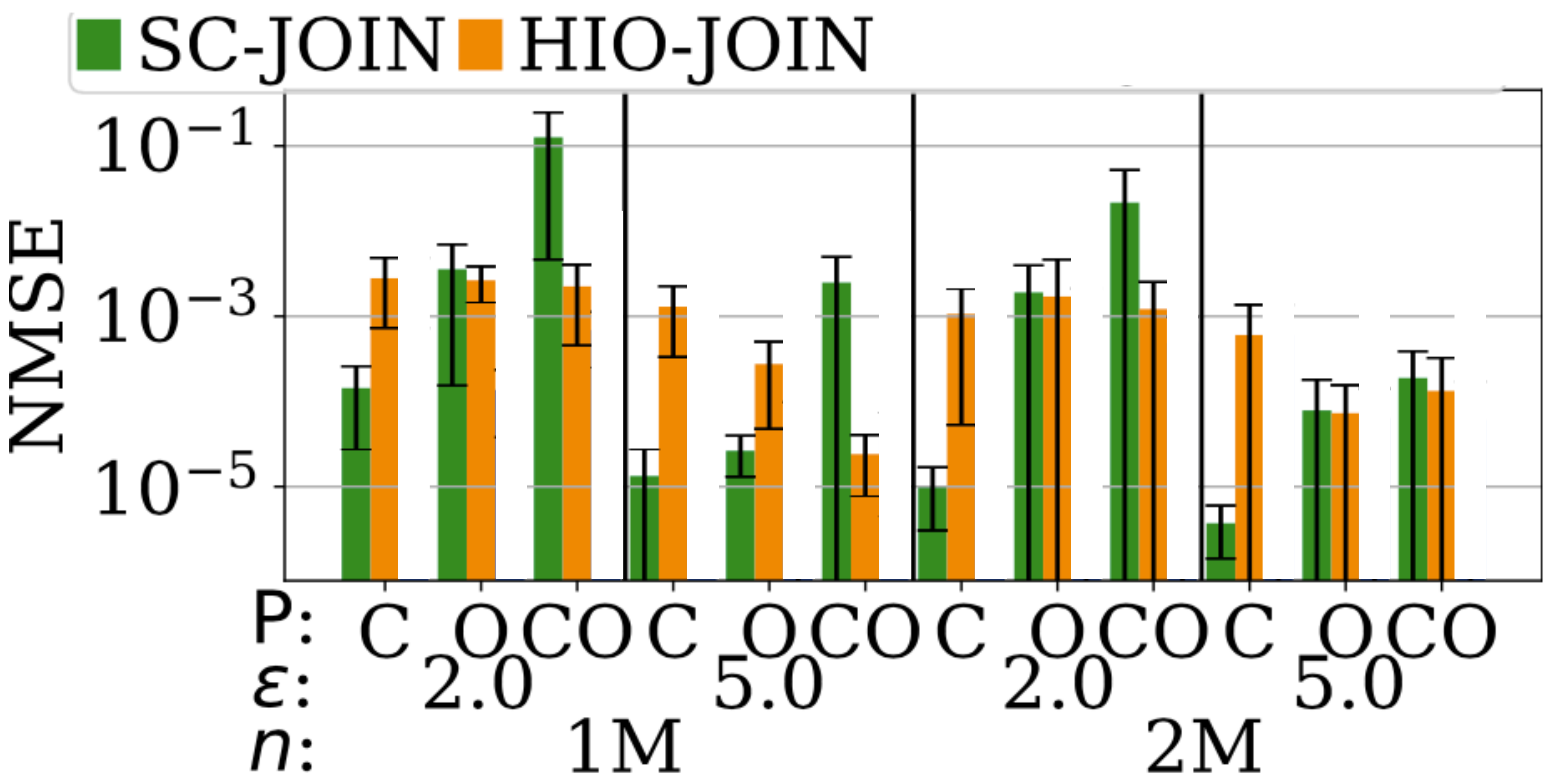
ϵ=2, d=2, d_q=1

Asymptotic Mean Squared Error

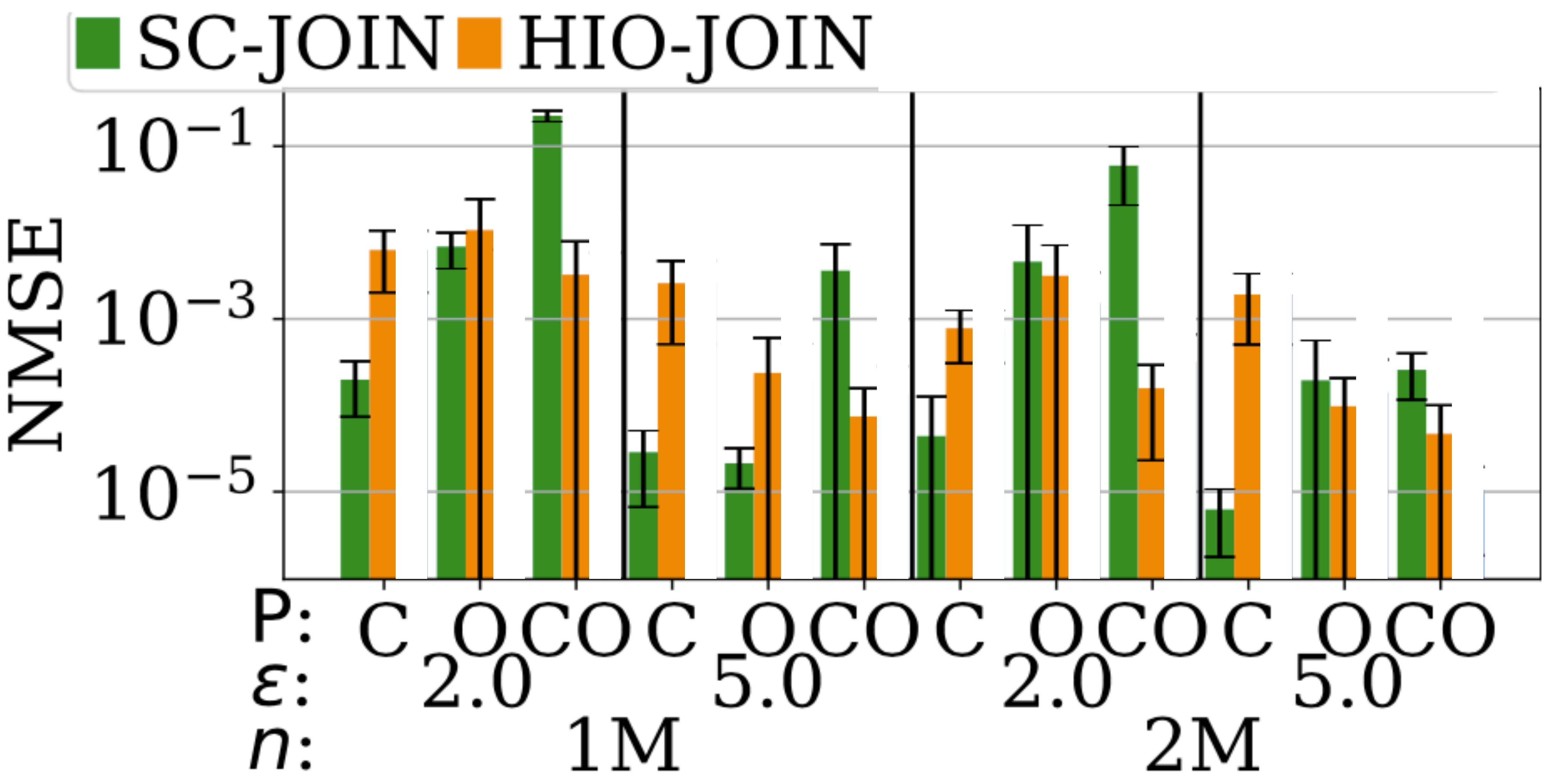
HIO: $O\left(\sum_{v\in A} v^2 \frac{n \log^{d+d_q} m}{\epsilon^2}\right)$, AHIO: $O\left(\frac{2(A_{\min}^2 + A_{\max}^2)d \cdot n \log^{d+d_q} m}{\epsilon^2}\right)$

1-1 Joint Aggregation

SELECT COUNT(*) FROM T1 JOIN T2 ON UID WHERE C



SELECT SUM(A) FROM T1 JOIN T2 ON UID WHERE C



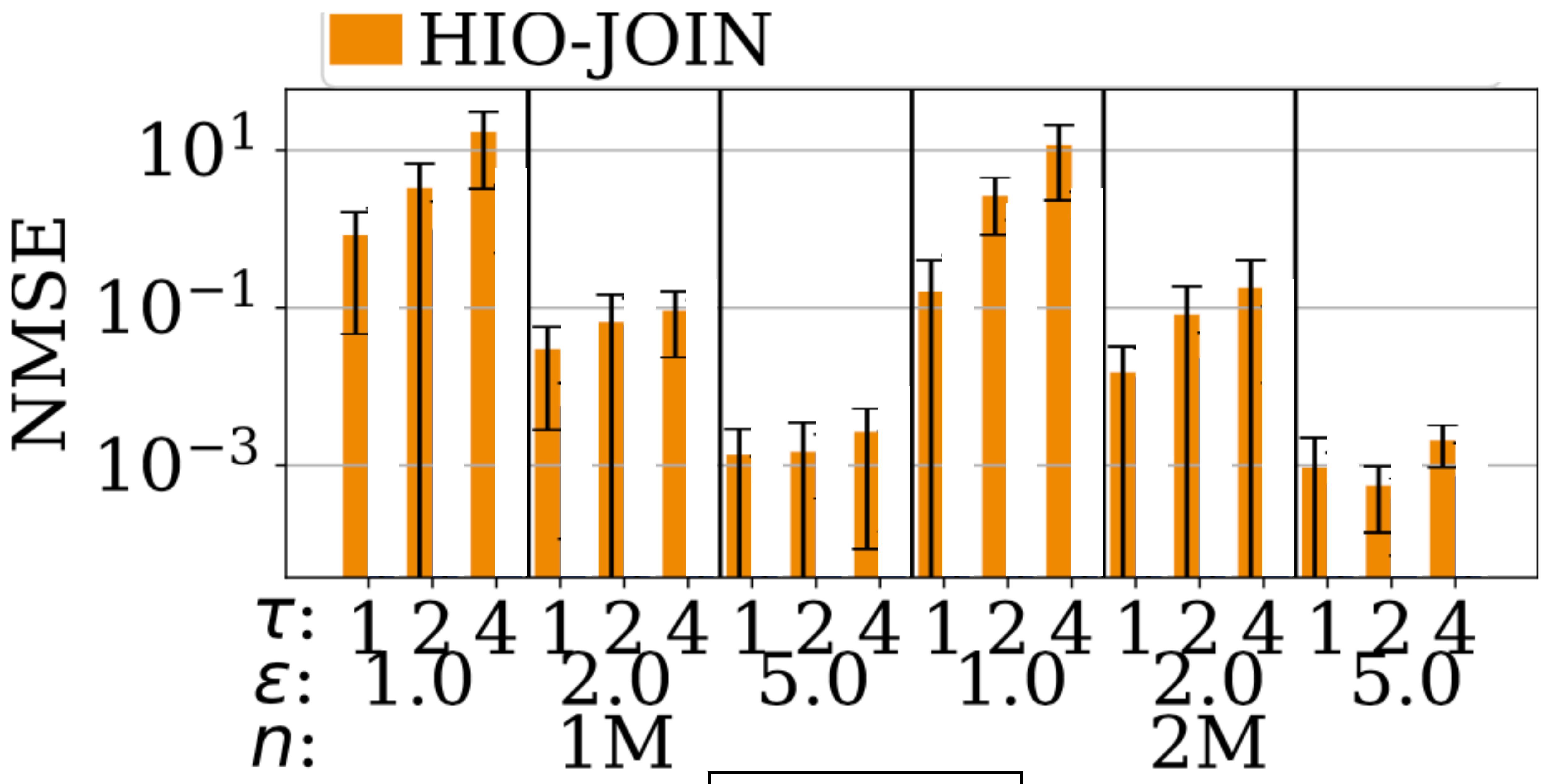
Asymptotic Mean Squared Error

SC-JOIN: $O\left(\frac{n \log^{3Kd_q} m}{(\epsilon/Kd)^{2Kd_q}}\right)$, HIO-JOIN: $O\left(\frac{n \log^{K(d+d_q)} m}{(\epsilon/K)^{2K}}\right)$

As range predicate gets complicated, i.e., d_q increases, error of SC-JOIN increases faster than that of HIO-JOIN

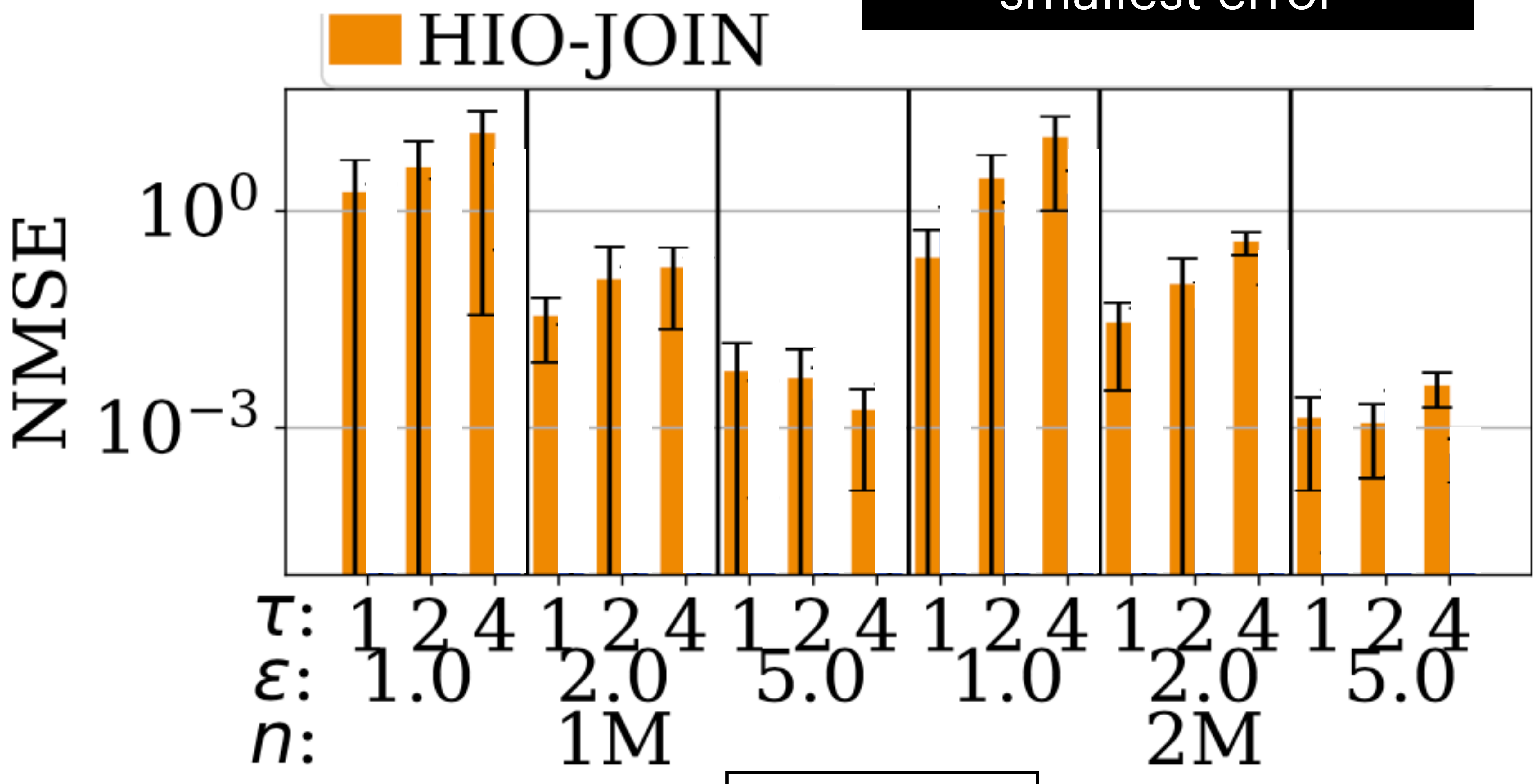
1-Many Joint Aggregation

τ = 1 leads to the smallest error



COUNT

d_q=2



SUM

Asymptotic Mean Squared Error of HIO-JOIN (with τ-truncation):

COUNT: $O\left(\frac{nd\tau(1+\tau)^4\log^{2(d+d_q)} m}{\epsilon^4}(r_{\min}^2+r_{\max}^2)\right)$, SUM: $O\left(\frac{nd\tau(1+\tau)^4\log^{2(d+d_q)} m}{\epsilon^4}(r_{\min}^2+r_{\max}^2)(A_{\min}^2+A_{\max}^2)\right)$

More in the Paper

- Efficient range utility optimization based on range decomposition and consistency
- Support for group-by aggregations
- More experiments using real-world PUMS datasets and other tested schemes
- Alternative instantiation of the Partition-Rounding-Perturb framework by embedding the rounding value in the partition attribute

Thank you!
Q & A