Introduction to Differential Privacy

CMSC 23200/33250, Winter 2021, Lecture 22

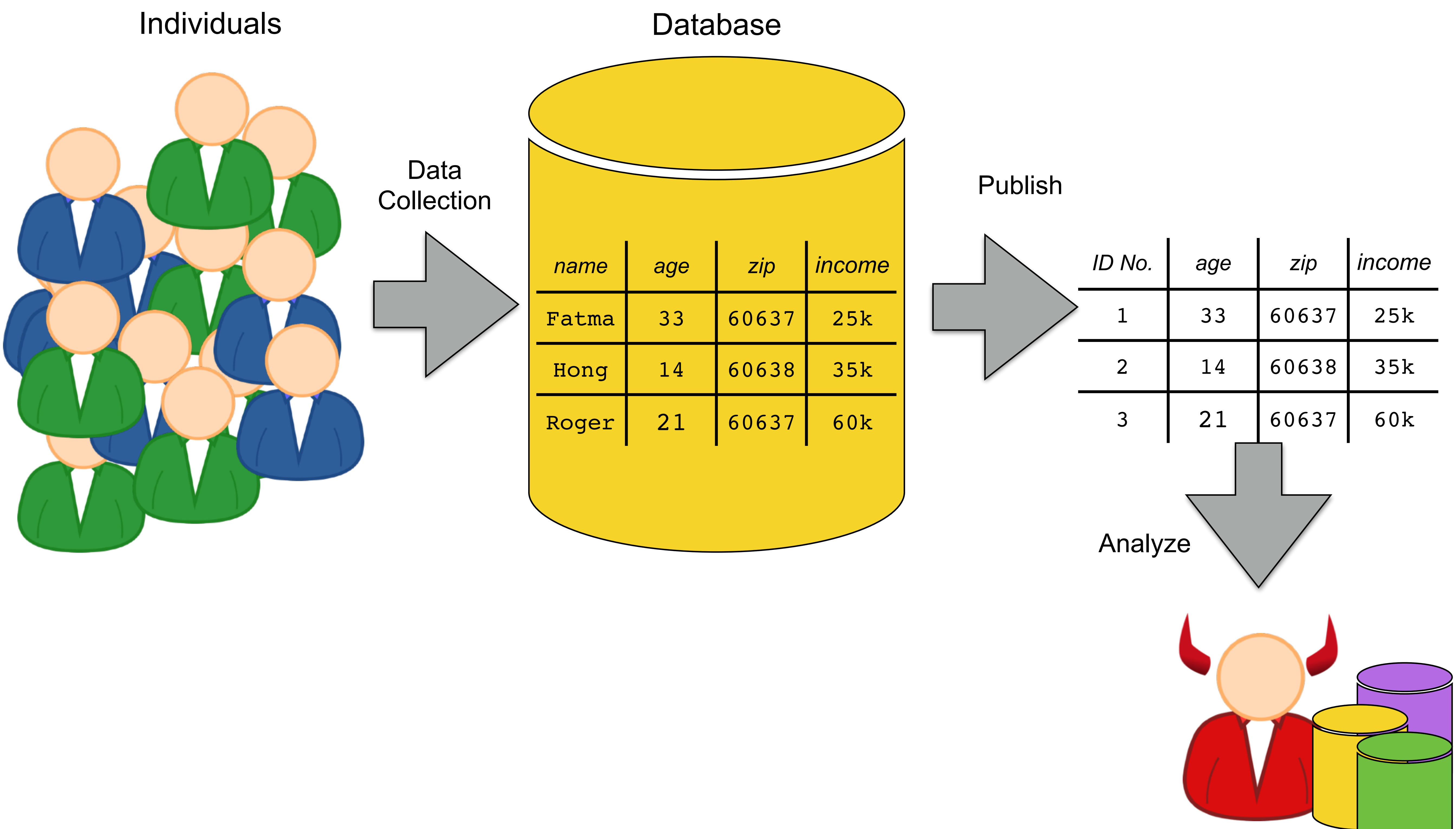
David Cash and Blase Ur

University of Chicago

Outline

1. Basic Setting and Ideas for Differential Privacy
2. Local Differential Privacy and Randomized Response
3. (Traditional) Differential Privacy
4. Some Attacks against Differential Privacy Systems

Recalling the Problem Setting



Lessons from Last Time

1. Old methods (e.g. de-identification) provide little protection

2. Principled methods (k-anonymity, l-diversity) also fail often

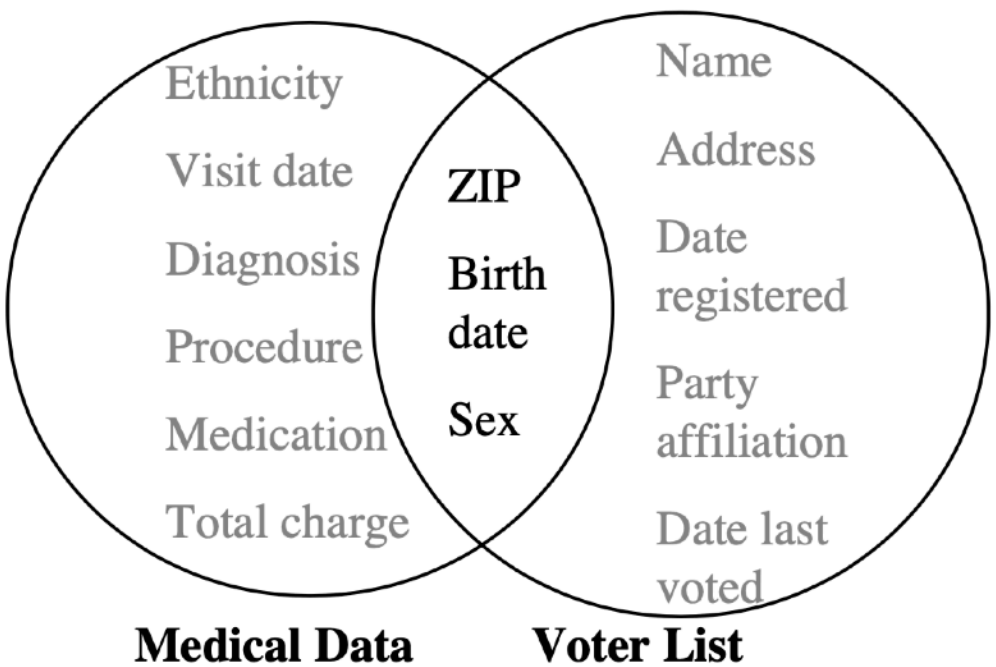


Figure 1 Linking to re-identify data

Source: L. Sweeney. k-anonymity: a model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-based Systems, 10 (5), 2002; 557-570.

	Non-Sensitive			Sensitive
	Zip Code	Age	Nationality	Condition
1	130**	< 30	$*$	Heart Disease
2	130**	< 30	$*$	Heart Disease
3	130**	< 30	$*$	Viral Infection
4	130**	< 30	$*$	Viral Infection
5	1485*	≥ 40	$*$	Cancer
6	1485*	≥ 40	$*$	Heart Disease
7	1485*	≥ 40	$*$	Viral Infection
8	1485*	≥ 40	$*$	Viral Infection
9	130**	3*	$*$	Cancer
10	130**	3*	$*$	Cancer
11	130**	3*	$*$	Cancer
12	130**	3*	$*$	Cancer

Fig. 2. 4-Anonymous Inpatient Microdata

Source: A. Machanavajjhala et al. l-Diversity: Privacy Beyond k-Anonymity. TKDD 2007.

Properties of an Ideal Data Privacy Solution

1. Hide *all* information that may be harmful to individuals.

2. Resist attacks by adversaries with *arbitrary* background data.

3. Still release *useful* information.

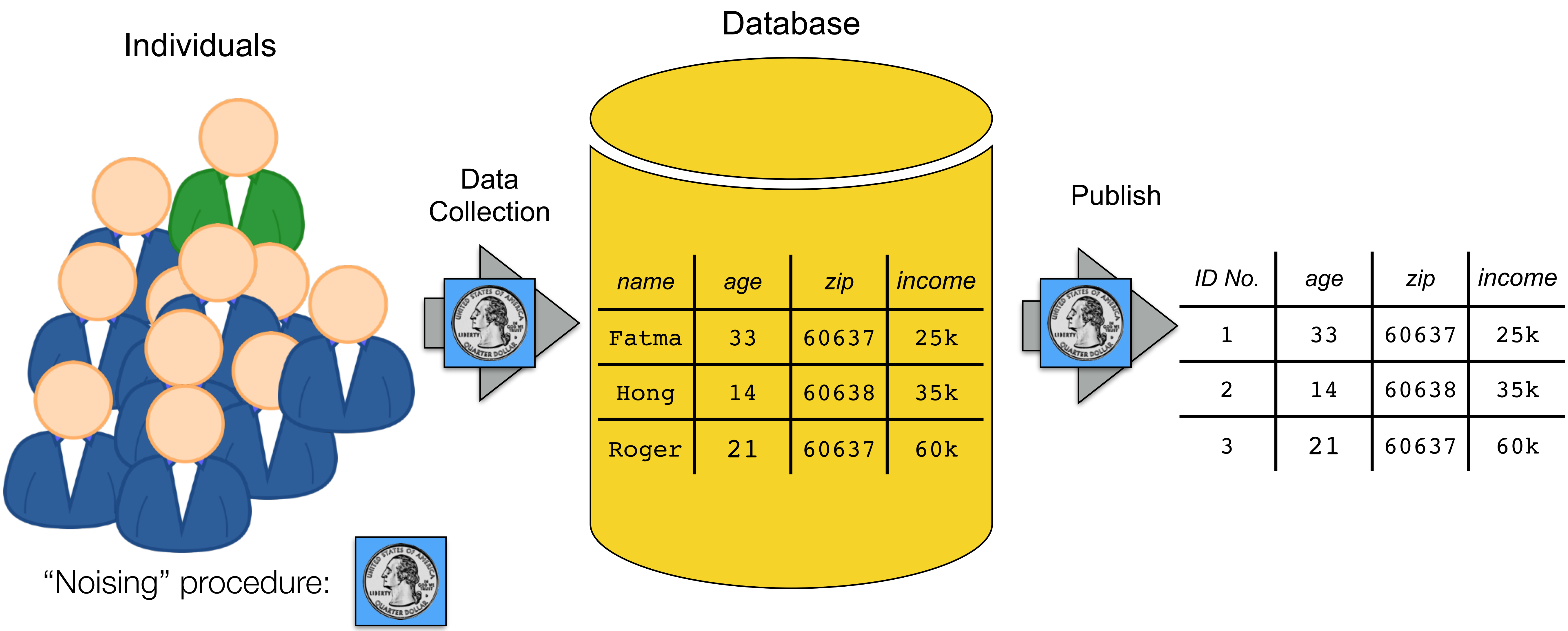
Initial Insights (inspired by Randomized Response)

1. Approximate answers can be (just as) useful.
2. Focus on the *distribution* of what is released, not actual responses.
3. Plausible deniability may be good protection.

Differential Privacy: Main Idea

<u>Design Philosophy</u>: Publish data with some random "noise" added. Adding or removing any individual from the data should not change the *distribution* of the output "by too much".

- If data release does not change much when an individual is included, conclude that they are protected.



Differential Privacy: TODOs

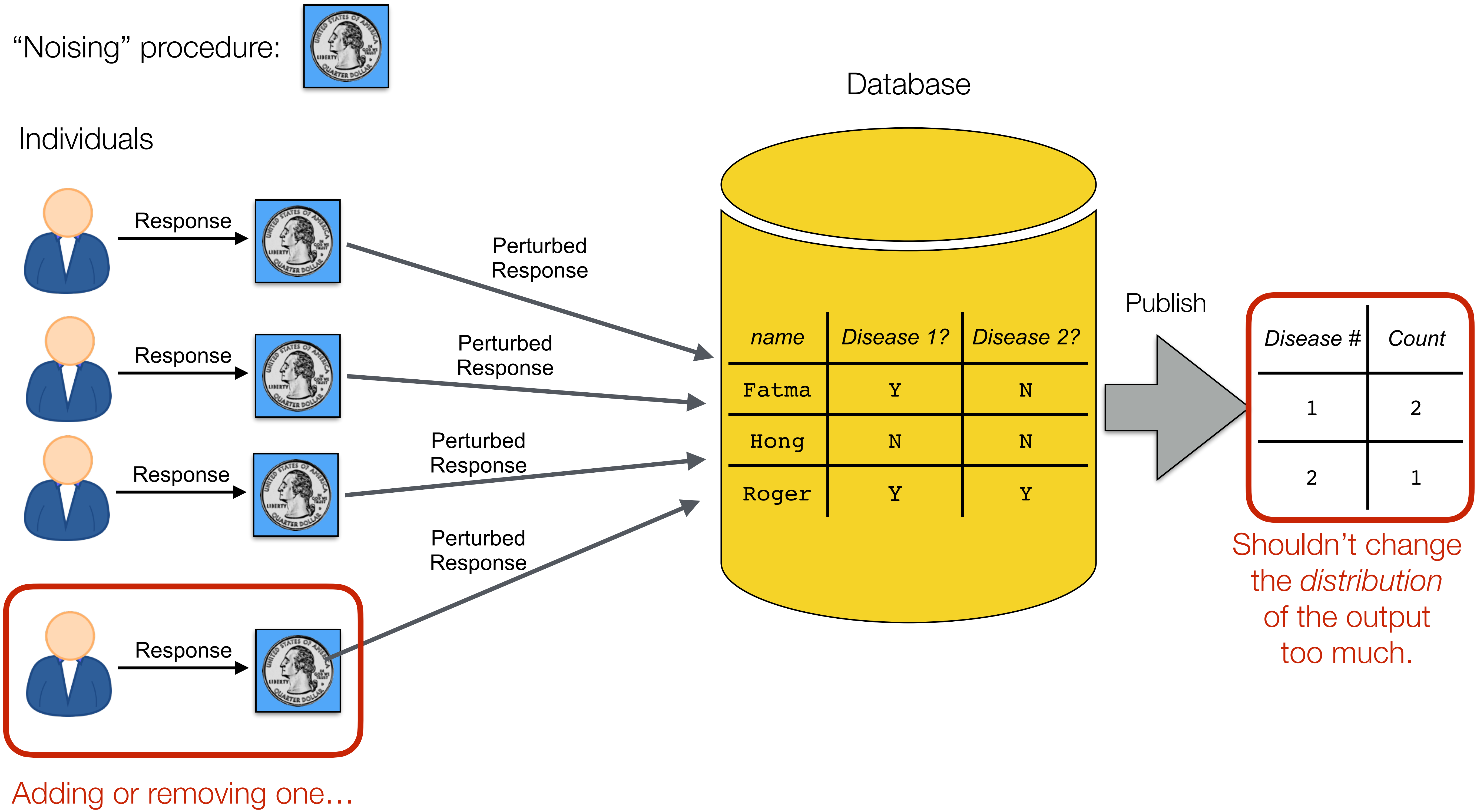
Design Philosophy: Publishing data with some random “noise” added. Adding or removing an individual from the data should not change the *distribution* of the output “by too much”.

- How should this noise be chosen? How much noise?
- How should we measure changes in distributions?
- What do these protections mean in practice?

Outline

1. Basic Setting and Ideas for Differential Privacy
2. **Local Differential Privacy and Randomized Response**
3. (Traditional) Differential Privacy
4. Some Attacks against Differential Privacy Systems

System Architecture for Local Differential Privacy



name	*Disease 1?*	*Disease 2?*
Fatma	Y	N
Hong	N	N
Roger	Y	Y

Disease #	*Count*
1	2
2	1

Defining Local Differential Privacy

Definition: A randomized algorithm A is *ε-locally-differentially-private* if:

- For *every* pair of local inputs x, x'
- For *every* set S of possible outputs

It holds that:

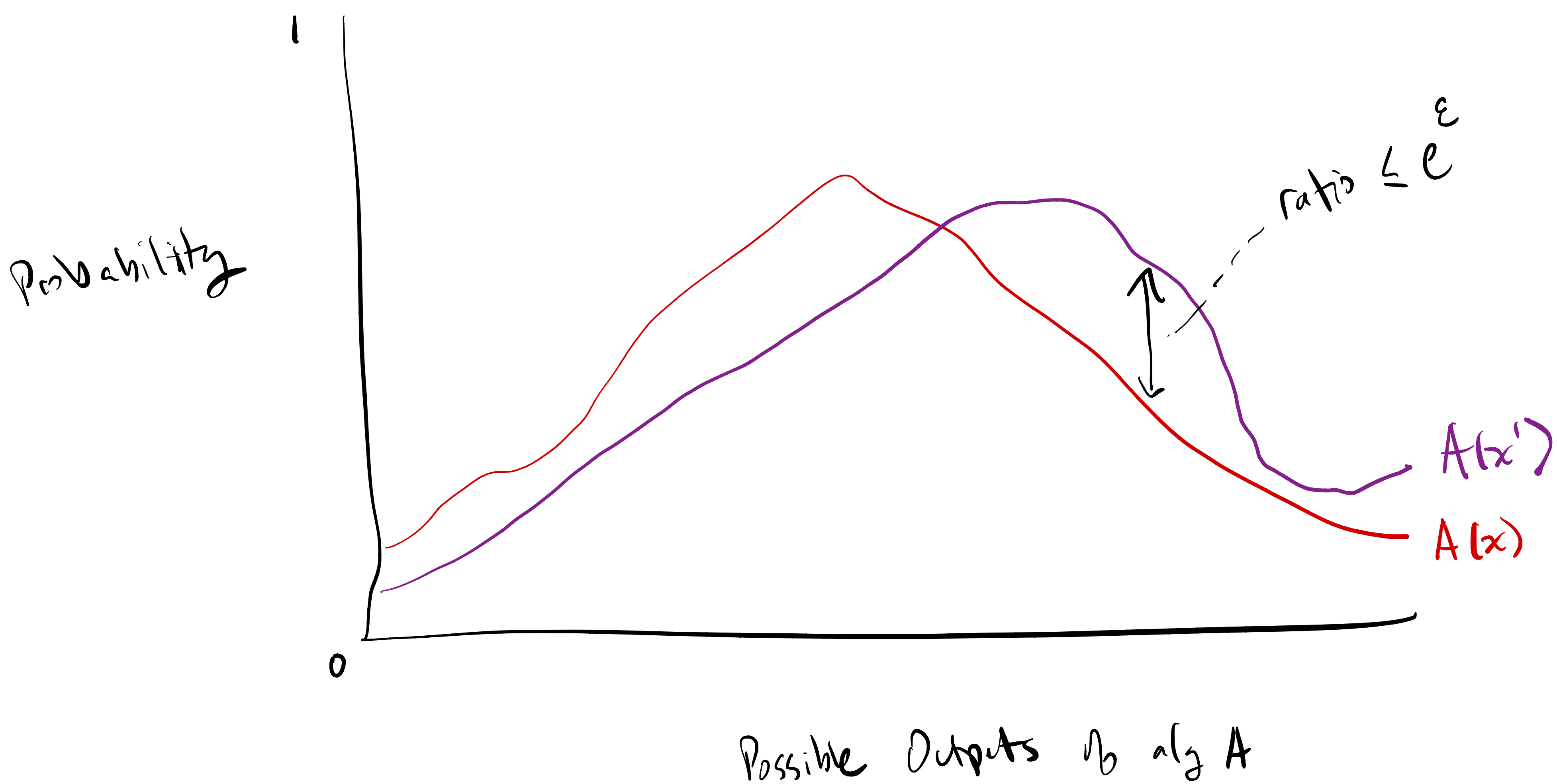
$$\Pr[A(x) \in S] \leq e^{\varepsilon} \cdot \Pr[A(x') \in S]$$

- $e \approx 2.71$ is Euler's number. Could just use ε instead of e^{ε}
- Definition is symmetric in x, x'.
- The event "$A(x) \in S$" can represent any observation as the set S changes. ("Average was in some range" or "Even number of people had disease").
- Smaller ε means better privacy. ε=0 means distributions are *same*.

Measuring Distribution Change

Fix any local inputs x, x', $A(x), A(x')$ induce two distributions that we hope are close. We know:

$$\Pr[A(x) \in S] \leq e^{\varepsilon} \cdot \Pr[A(x') \in S]$$



Randomized Response is Locally-DP

<u>Definition of</u> A_{rr}: Takes an input $x \in \{Y, N\}$.

- With probability 0.5, $A_{\mathrm{rr}}(x) = x$
- With probability 0.5, $A_{\mathrm{rr}}(x)$ outputs Y or N uniformly at random.

<u>Claim</u>: A_{rr} is ε-locally-DP for $\varepsilon = \ln 3 \approx 1.10$.

<u>Proof</u>: Must show for all $S \subseteq \{Y, N\}$ and all $x, x' \in \{Y, N\}$,

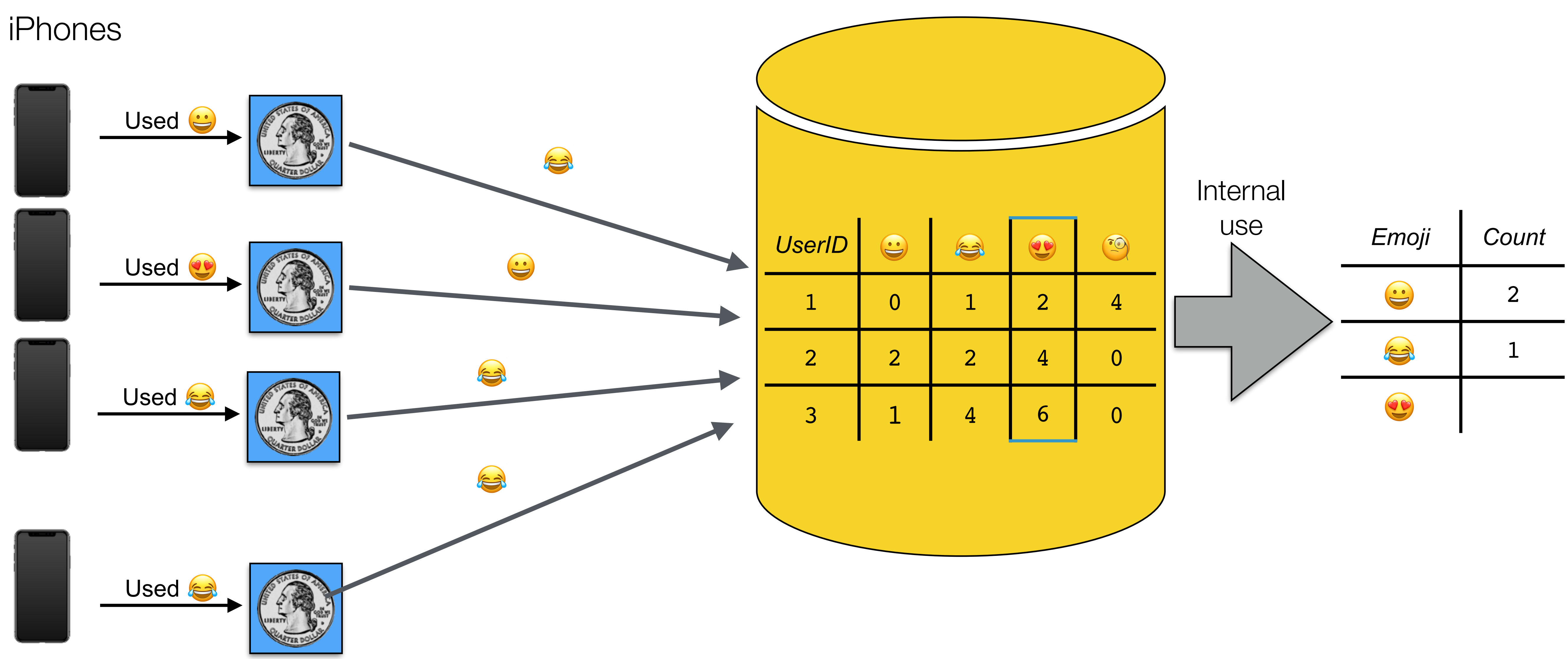
$$\Pr[A_{\mathrm{rr}}(x) \in S] \leq e^{\varepsilon} \cdot \Pr[A_{\mathrm{rr}}(x') \in S].$$

Only need to consider $x = Y, x' = N$ and $S = \{Y\}$ or $S = \{N\}$:

- $\Pr[A_{\mathrm{rr}}(Y) = Y] = 0.5 + 0.5 \cdot 0.5 = 0.75$
- $\Pr[A_{\mathrm{rr}}(N) = Y] = 0.5 \cdot 0.5 = 0.25$ (others are similar)

Checking cases, $\Pr[A_{\mathrm{rr}}(x) \in S] \leq 3 \cdot \Pr[A_{\mathrm{rr}}(x') \in S]$ always.
In other words: A_{rr} is ε-locally-DP for $\varepsilon = \ln 3$.

Deployed Local DP: Apple iPhone Data Collection



- Per-user table is supposedly not actually stored
- Also collecting: Power usage, text slang (!), …

Outline

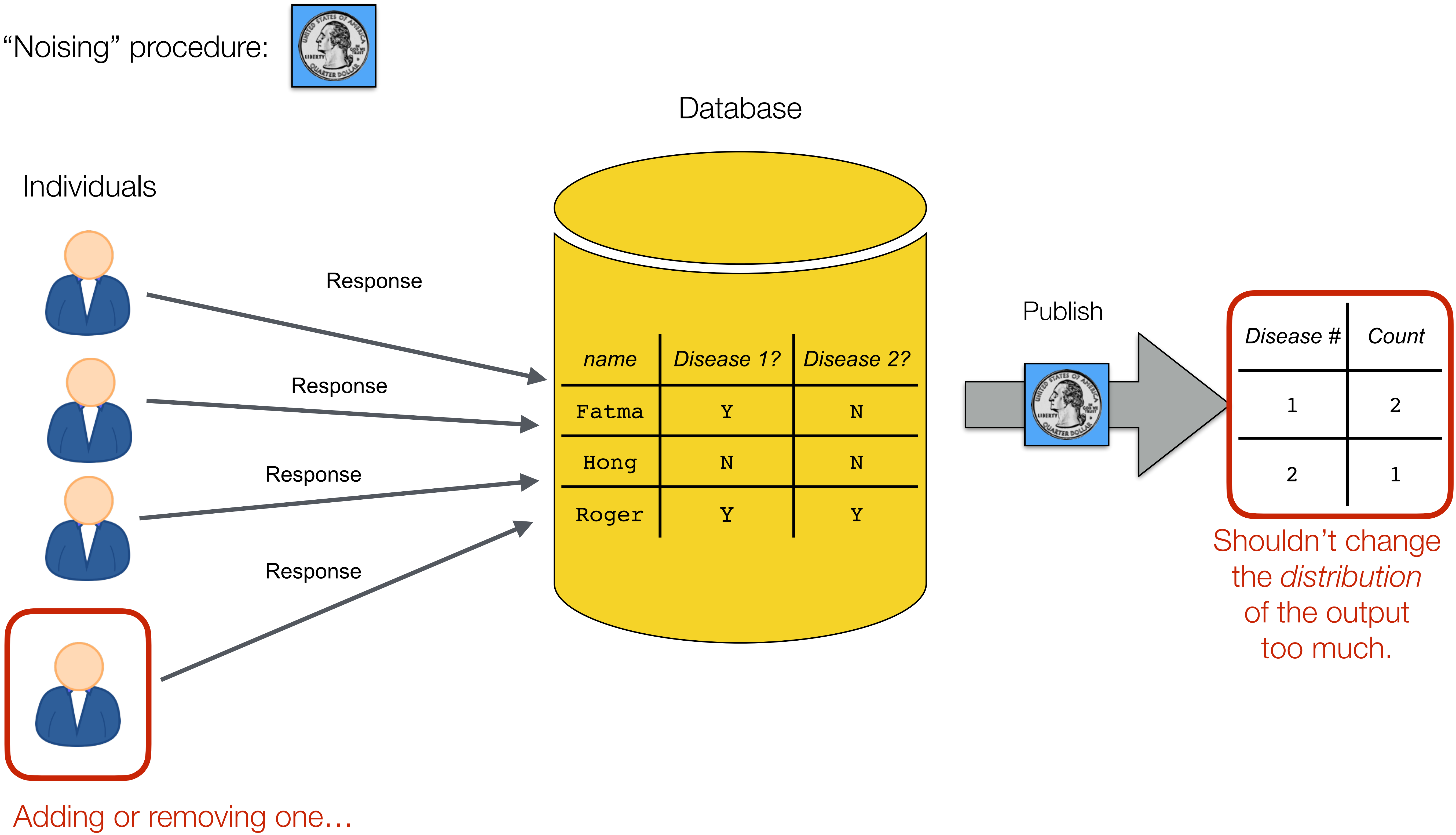
1. Basic Setting and Ideas for Differential Privacy

2. Local Differential Privacy and Randomized Response

3. (Traditional) Differential Privacy

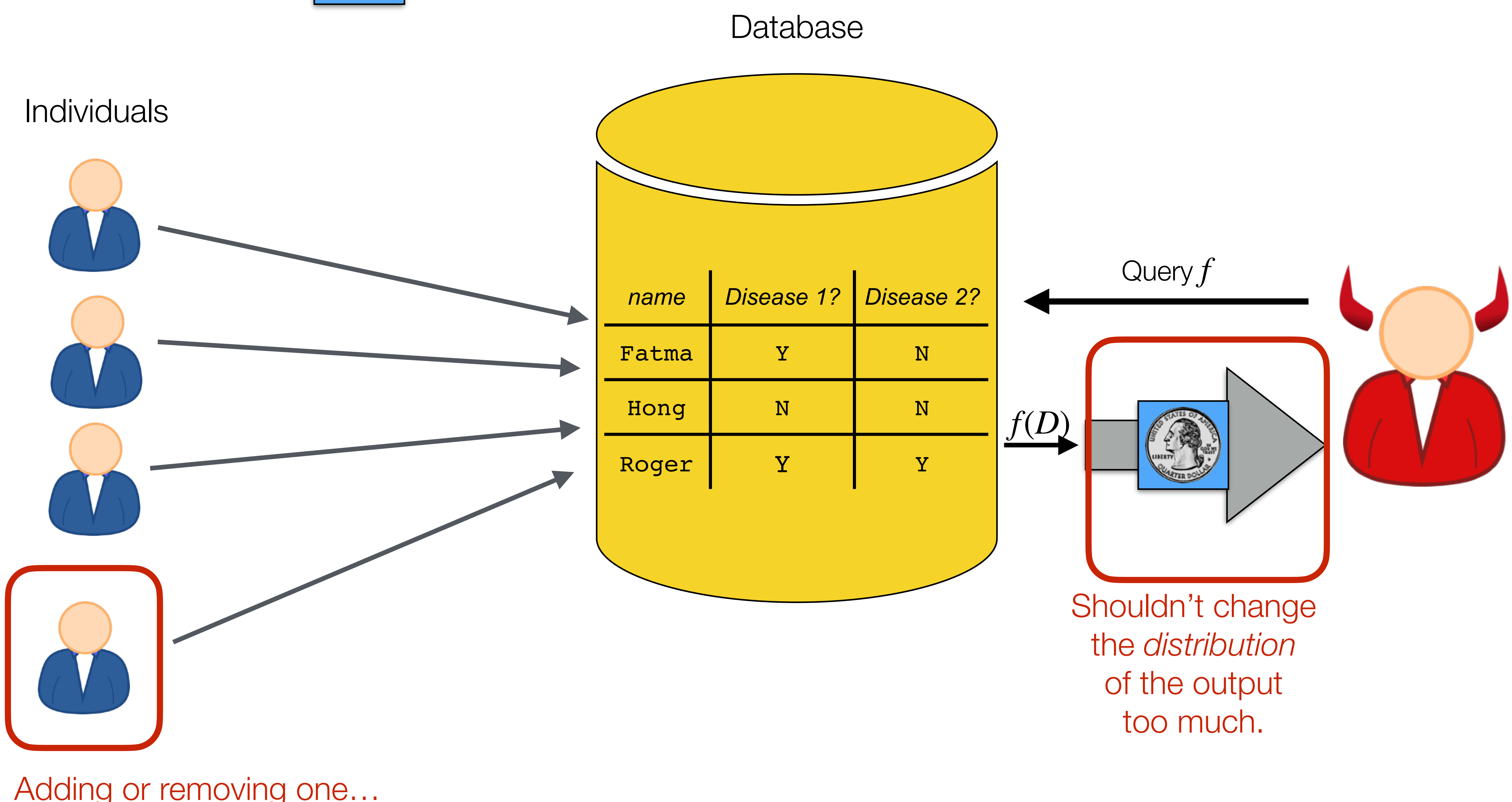
4. Some Attacks against Differential Privacy Systems

System Architecture: (Traditional) Differential Privacy



System Architecture: (Traditional) Differential Privacy

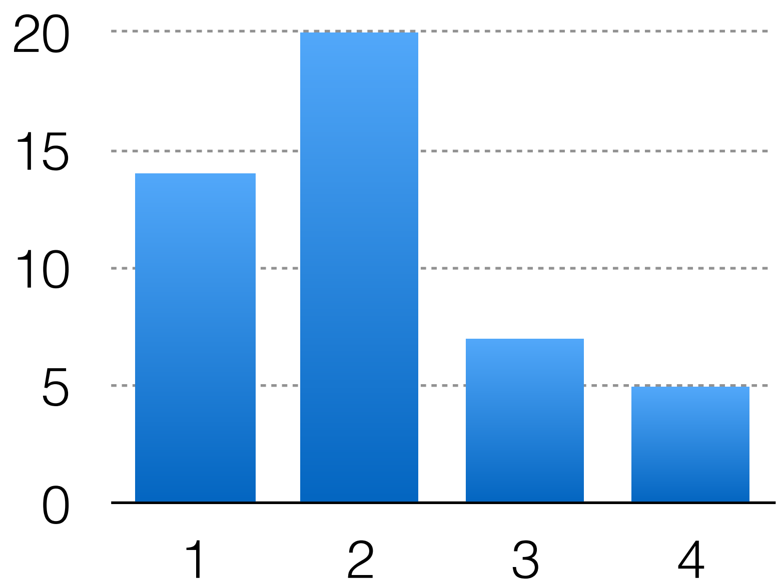
“Noising” procedure: 



- Noised version of $f(D)$ is usually denoted $\mathscr{M}(D)$

Simplifying the Problem: Abstract "Databases"

Type	*Count*
1	14
2	20
3	7
4	5



- Data D is table of counts
- Query f can be arbitrary function of D

<u>Definition</u>: Datasets D, D' are *<u>neighboring</u>* if they are exactly the same, except they differ by exactly 1 in a single count.

Example:

Type	*Count*
1	12
2	20
3	9
4	2

Type	*Count*
1	12
2	**<u>21</u>**
3	9
4	2

Defining Differential Privacy

Definition: A randomized algorithm $\mathscr{M}$ is *ε-differentially-private* if:

- For *every* pair of *neighboring* tables D, D'
- For *every* set S of possible outputs

It holds that:

$$\Pr[\mathscr{M}(D) \in S] \leq e^{\varepsilon} \cdot \Pr[\mathscr{M}(D') \in S]$$

- Goal: Add as little noise as possible while respecting definition

Calibrating Noise: Sensitivity of a Query

Definition: The *sensitivity of a function* f, denoted Δf, is defined to be

$$\Delta f = \max_{D,D'} |f(D) - f(D')|$$

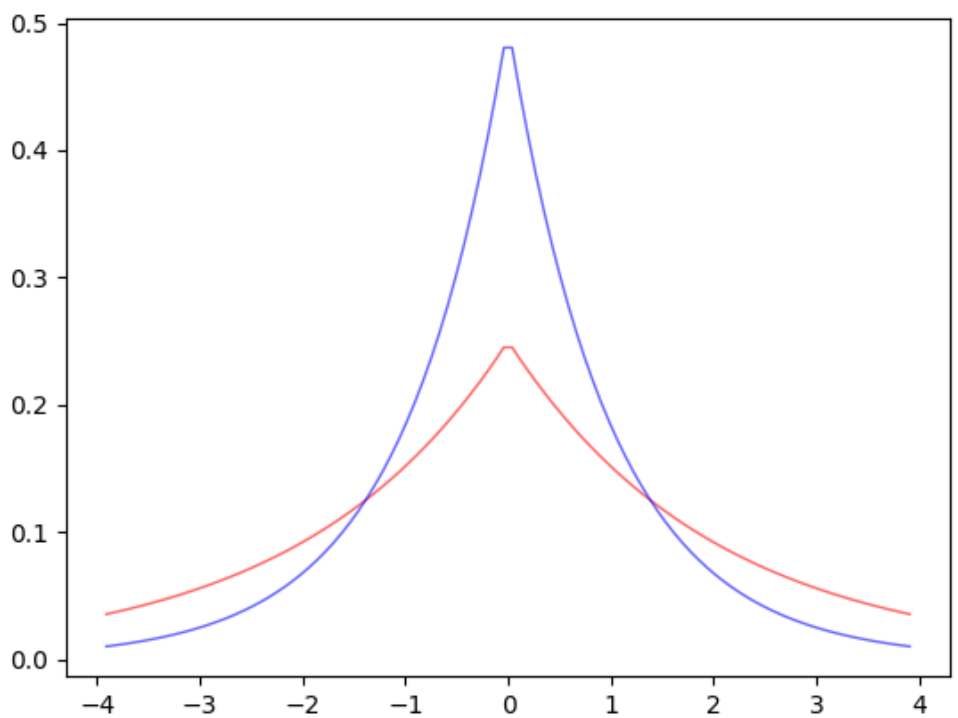
where the maximum is taken over *neighboring* pairs of tables D, D'.

- Adding someone to D can change $f(D)$ by at most Δf.
- Plan: Add more noise when Δf is large, to hide effect of individual.
- Won't need to worry about any other property of f

The Laplace Distribution

<u>Definition</u>: The *Laplace distribution (centered at zero) with scale b* is defined to have probability density function

$$\frac{1}{2b}e^{-|x|/b}.$$



b=1 (blue) and b=2 (red)

- Larger scale ⇒ More variance

The Laplace Mechanism

Definition: The *Laplace Mechanism* $\mathscr{M}$ *for a query* f *with privacy parameter* ε *is defined to be*

$$\mathscr{M}(D) = f(D) + \mathrm{Laplace}(\Delta f/\varepsilon).$$

- Larger Δf ⇒ Larger scale ⇒ More variance ⇒ Less utility
- Smaller ε ⇒ Larger scale ⇒ More variance ⇒ Less utility
- Can show: This is “optimal” distribution amongst ε-DP mechanisms.

Claim: $\mathscr{M}$ is ε-DP.

Laplace Mechanism: Example

Database D:

Disease #	*Count*
1	21
2	11
3	4
4	94
5	77

Query $f(D)$: Output number with disease #1

Question: What is Δf ?

Database D:

Name	*Age*
Fatma	21
Hong	20
Ron	35
Oren	25

Query $f(D)$: Output number of people over 21

Question: What is Δf ?

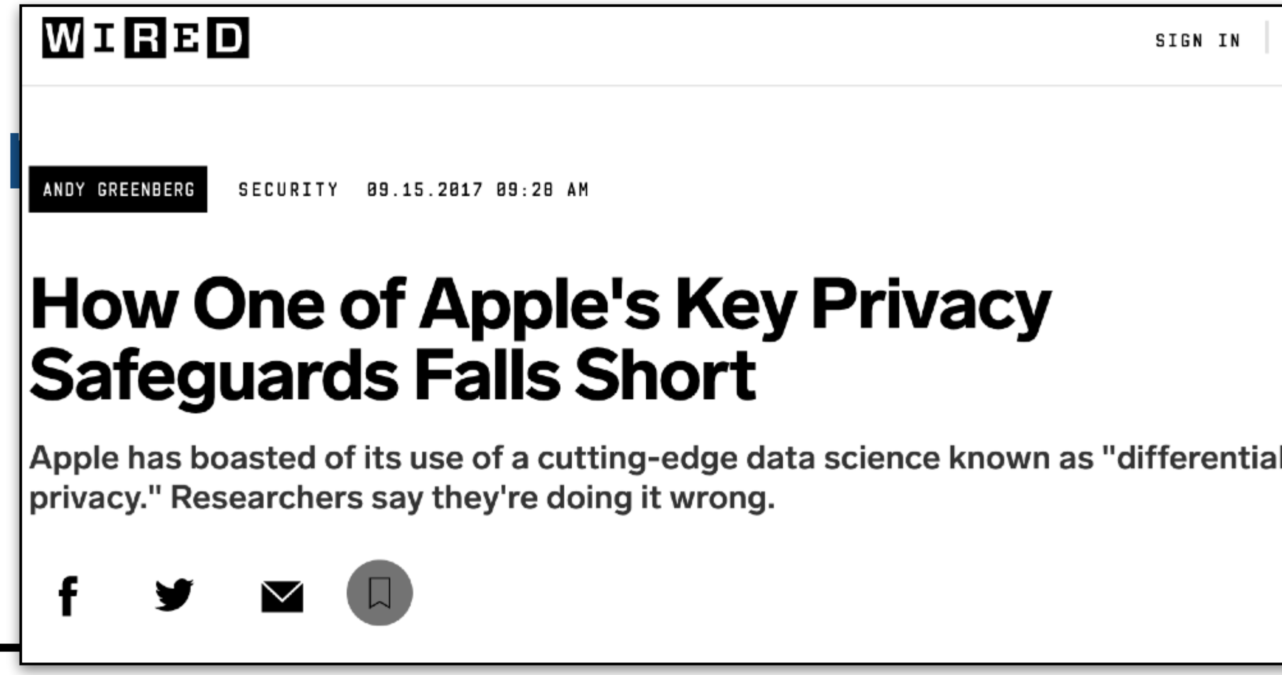
Database D:

Name	*Income*
Fatma	35k
Hong	20k
Ron	100k
Oren	50k

Query $f(D)$: Average income

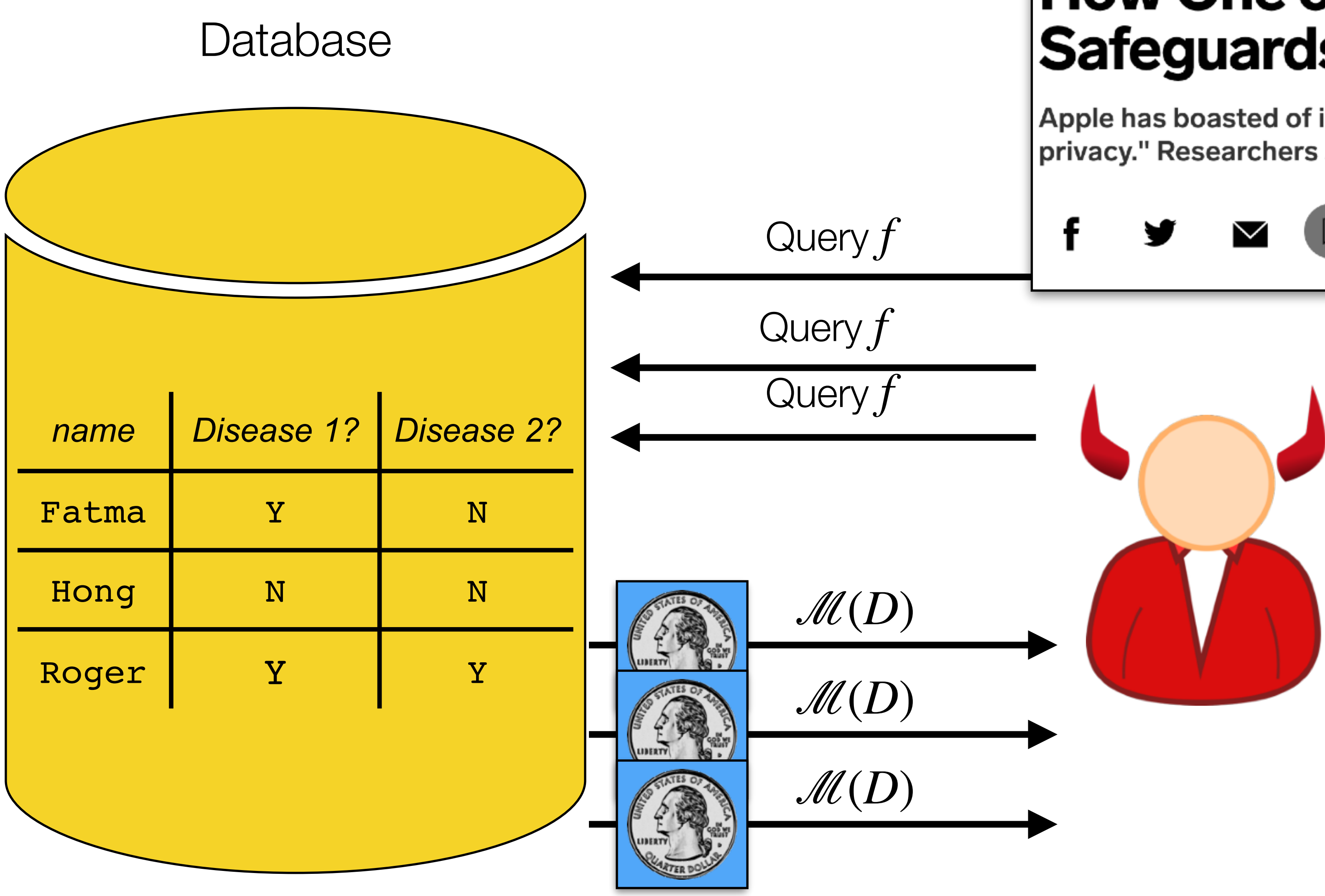
Question: What is Δf ?

Flaws in DP Systems (Assig


WIRED
SIGN IN
ANDY GREENBERG SECURITY 09.15.2017 09:20 AM

How One of Apple's Key Privacy Safeguards Falls Short

Apple has boasted of its use of a cutting-edge data science known as "differential privacy." Researchers say they're doing it wrong.



- If adversary can repeat query many times…
- Average of results will be true answer.
- In practice, systems must manage a “privacy budget”

Floating Point and Laplace Mechanism (Assignment 8)

```
import numpy.random

def laplace_mechanism(val, sensitivity, epsilon):
    noise = numpy.random.laplace(0.0, scale=sensitivity/epsilon)
    return val + noise
```

- Numeric variables above are *floating point.* Not all numbers are representable.

On Significance of the Least Significant Bits For Differential Privacy

Ilya Mironov

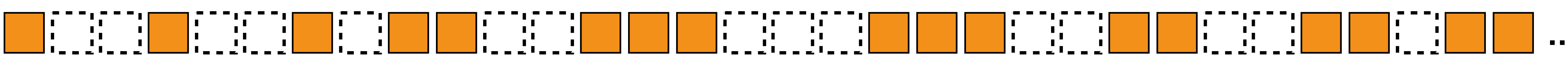
Floating Point and Laplace Mechanism (Assignment 8)

Attack setting:

- Adversary knows $f(D)$ is either 0 or 1
- Adversary gets to see $\mathscr{M}(D) = f(D) + \text{Laplace}(\Delta f / \varepsilon)$
- Adversary tries to guess $f(D)$
- Adversary should do no better than is allowed by ε-DP

Key insight: Most Laplace samplers do not output every possible floating point. Some numbers will *never* be output.

Representable floating point numbers:



⇒ Sampler outputs with non-zero probability

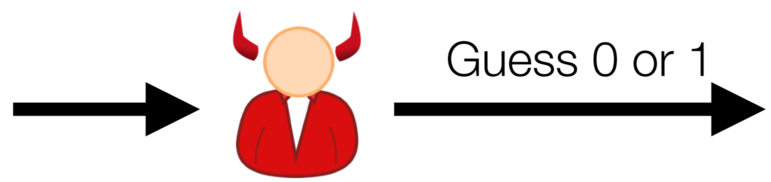
⇒ Sampler will never output

Floating Point and Laplace Mechanism (Assignment 8)

$$\mathscr{M}(D) = 0 + \mathrm{Laplace}(\Delta f/\varepsilon)$$

or

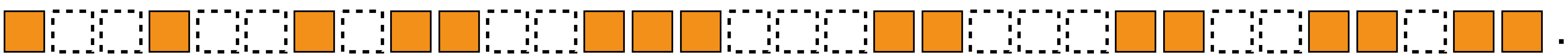
$$\mathscr{M}(D) = 1 + \mathrm{Laplace}(\Delta f/\varepsilon)$$



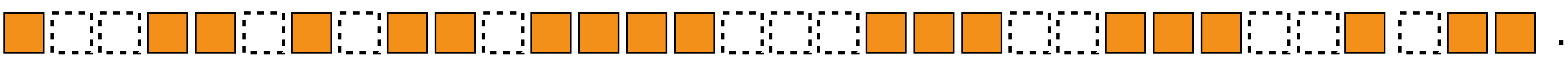
■ ⇒ Sampler outputs with non-zero probability

□ ⇒ Sampler will never output

Possible outputs when $f(D) = 0$ (i.e. $\mathscr{M}(D) = \mathrm{Laplace}(\Delta f/\varepsilon)$):



Possible outputs when $f(D) = 1$ (i.e. $\mathscr{M}(D) = 1 + \mathrm{Laplace}(\Delta f/\varepsilon)$):

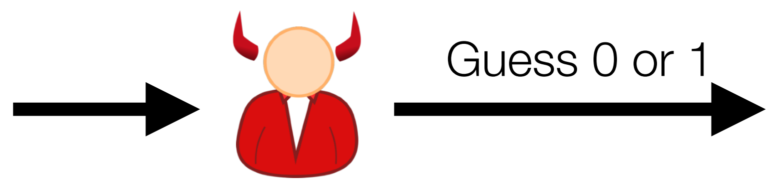


Floating Point and Laplace Mechanism (Assignment 8)

$$\mathscr{M}(D) = 0 + \text{Laplace}(\Delta f/\varepsilon)$$

or

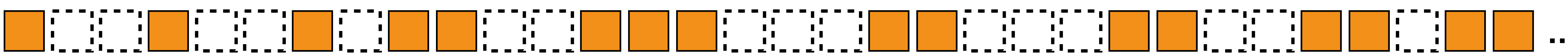
$$\mathscr{M}(D) = 1 + \text{Laplace}(\Delta f/\varepsilon)$$



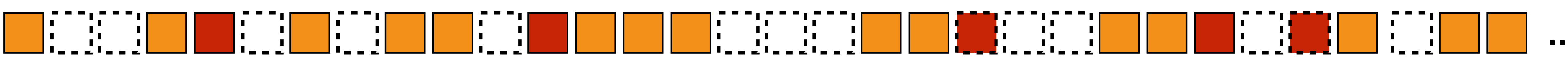
⇒ Sampler outputs with non-zero probability

⇒ Sampler will never output

Possible outputs when $f(D) = 0$ (i.e. $\mathscr{M}(D) = \text{Laplace}(\Delta f/\varepsilon)$):



Possible outputs when $f(D) = 1$ (i.e. $\mathscr{M}(D) = 1 + \text{Laplace}(\Delta f/\varepsilon)$):



⇒ "Smoking gun" samples that would only be output in one case.

The End