Differentially private data analysis with Tumult Analytics

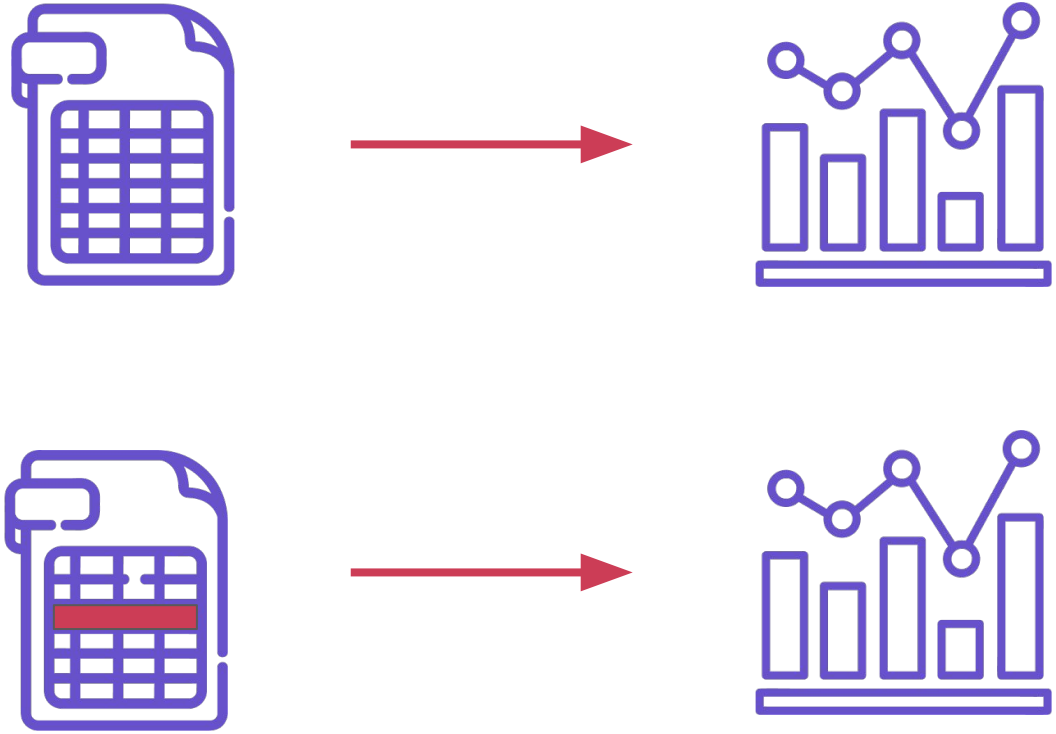
Webinar 2 for Wikimedia Foundation, July 2022

Damien Desfontaines

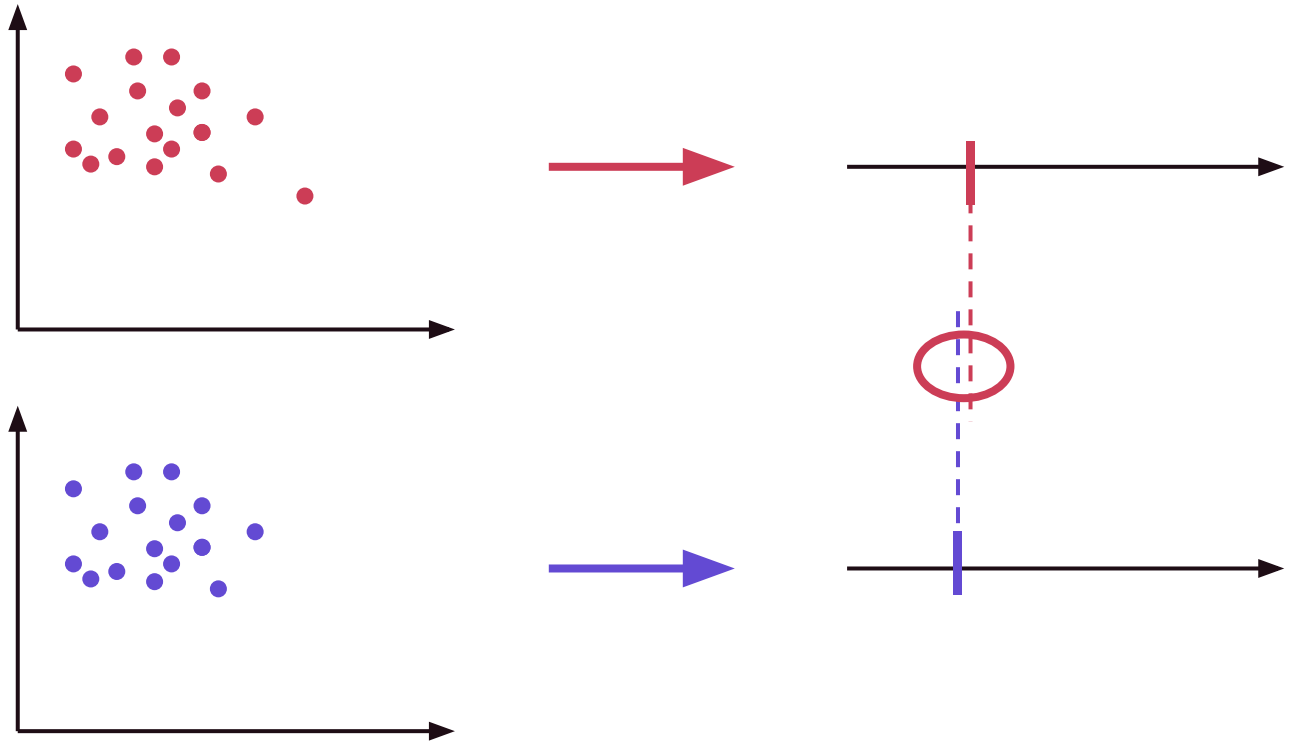
Recap from Webinar 1, outline of Webinar 2

- Differential privacy (DP) is a principled way of publishing data safely
- Previously: *what* is DP? Use cases, properties, vision for Wikimedia…
- Today: *how* to start using DP? Simple examples, with code snippets.

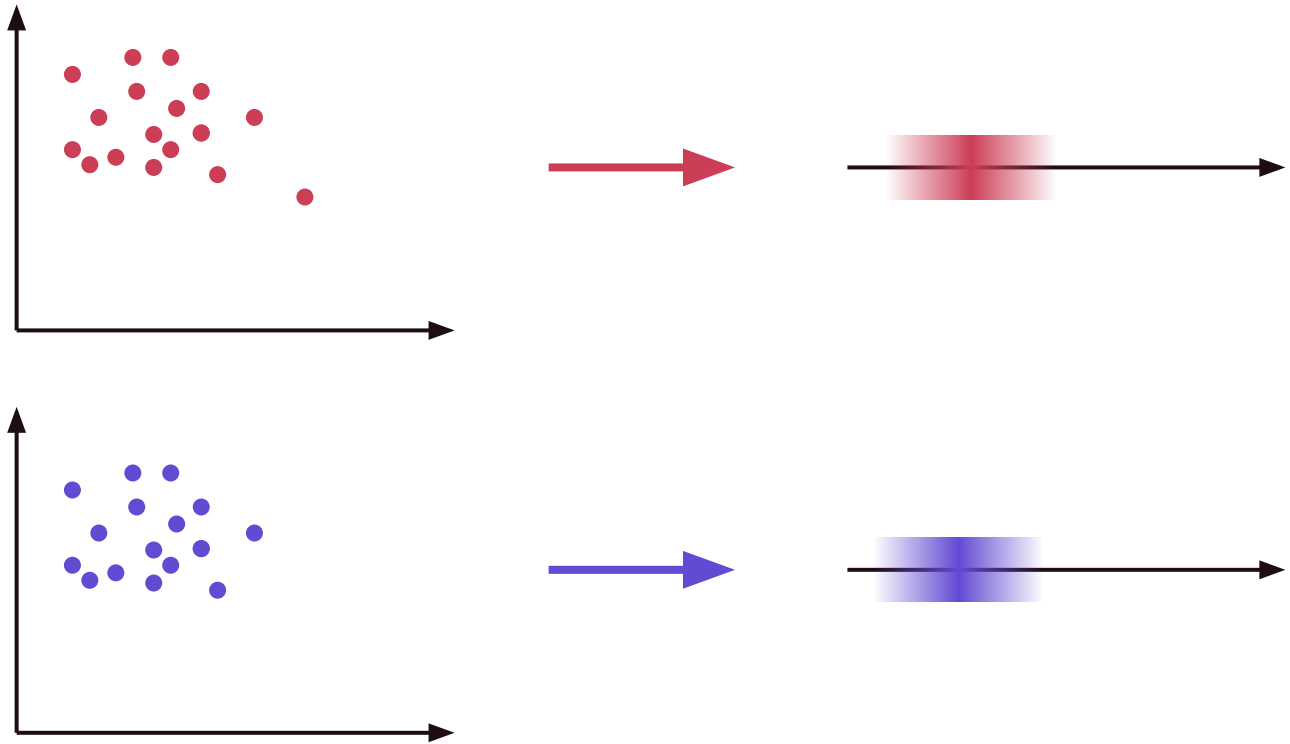
A simple formalization



A simple formalization

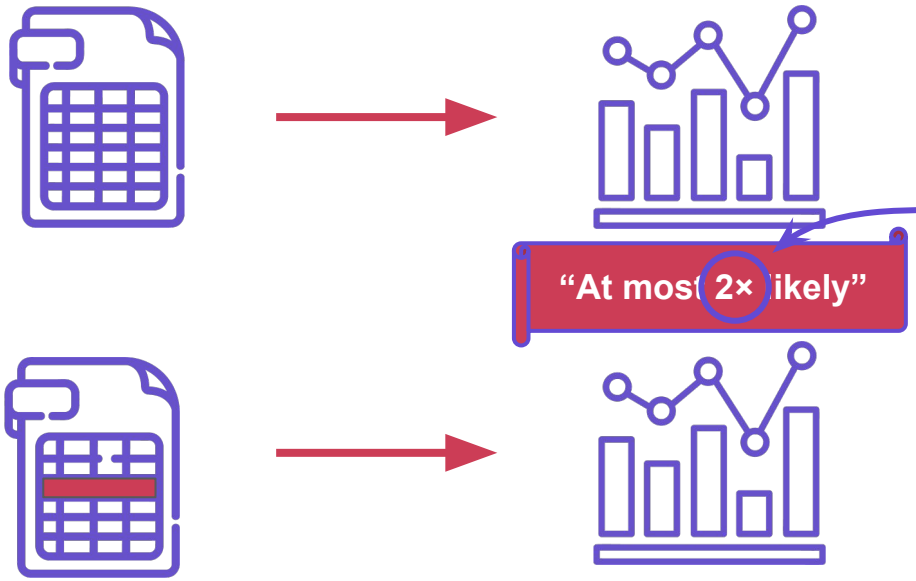


A simple formalization



A simple formalization

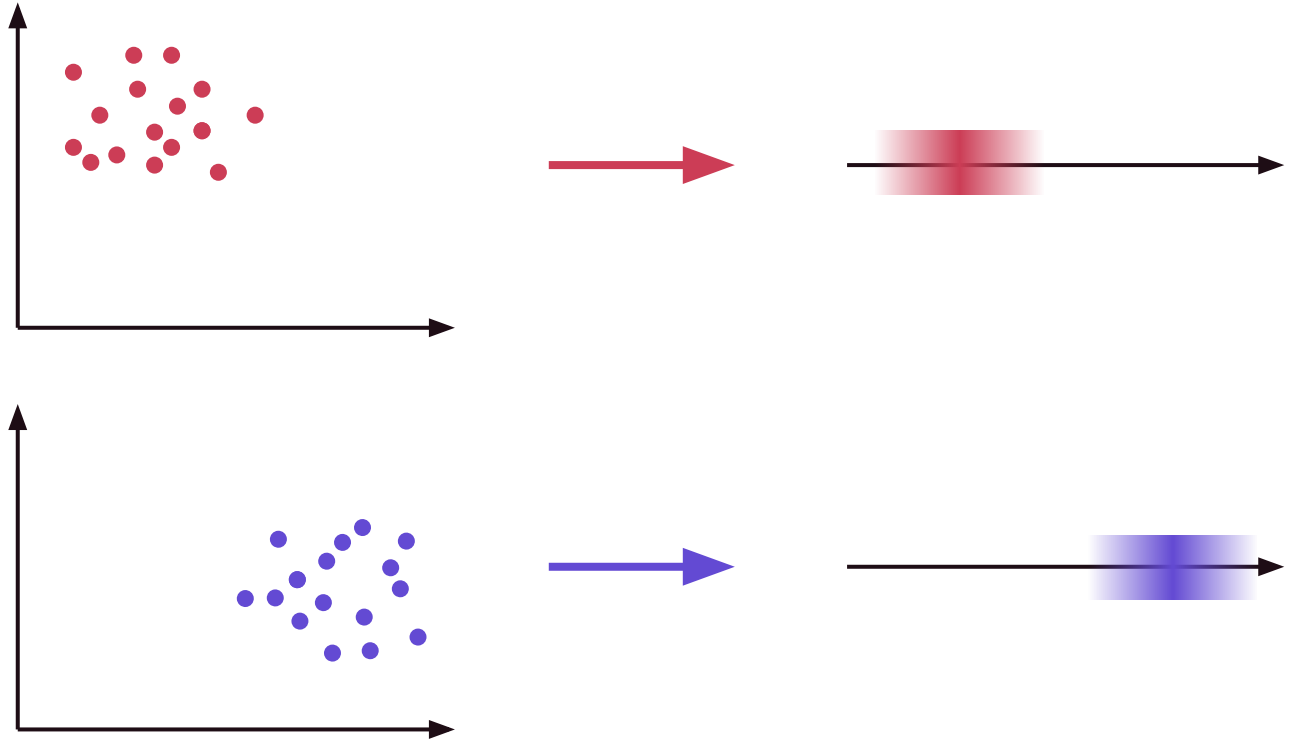
A simple formalization



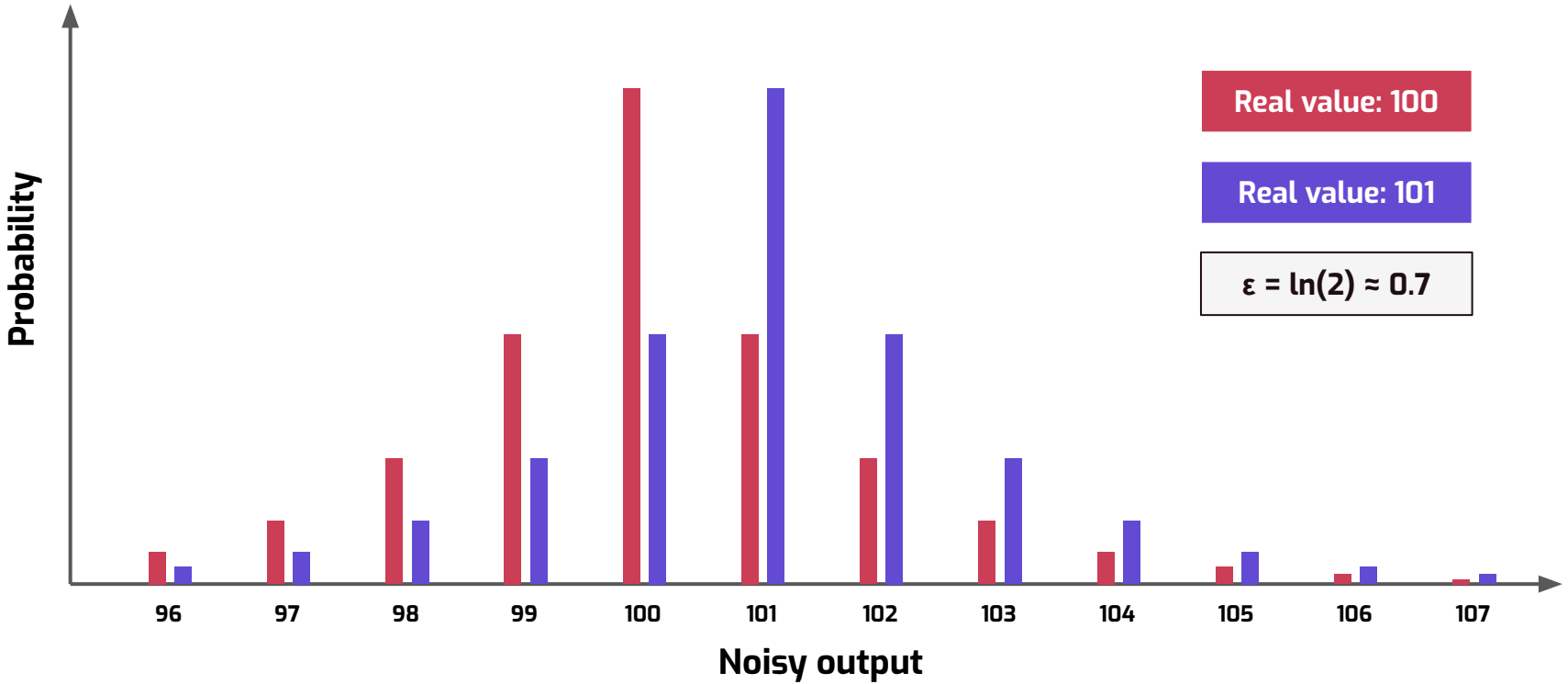
For *all* databases D_1, D_2 differing in a single record, and *all* possible outputs O:

$$P[M(D_1) = O] \leq e^{\varepsilon} \times P[M(D_2) = O]$$

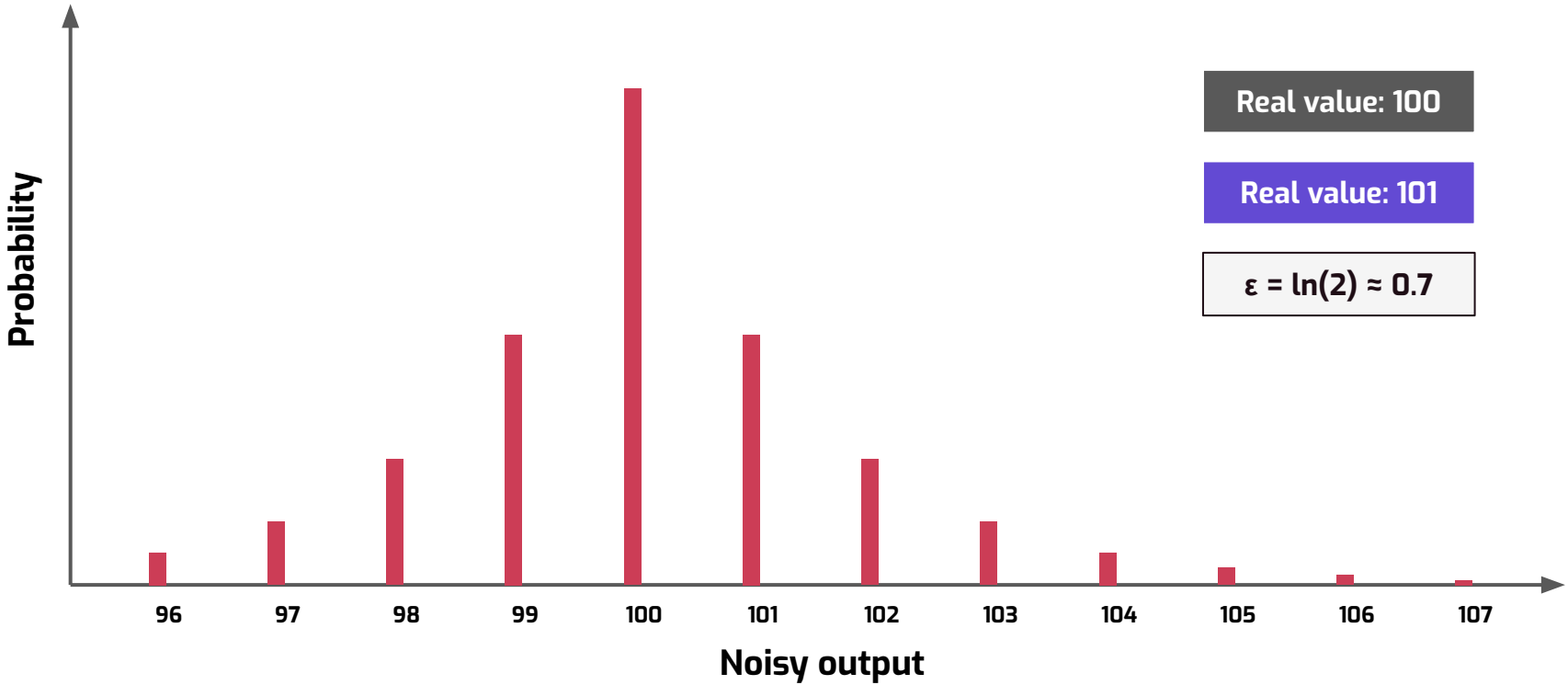
A simple formalization



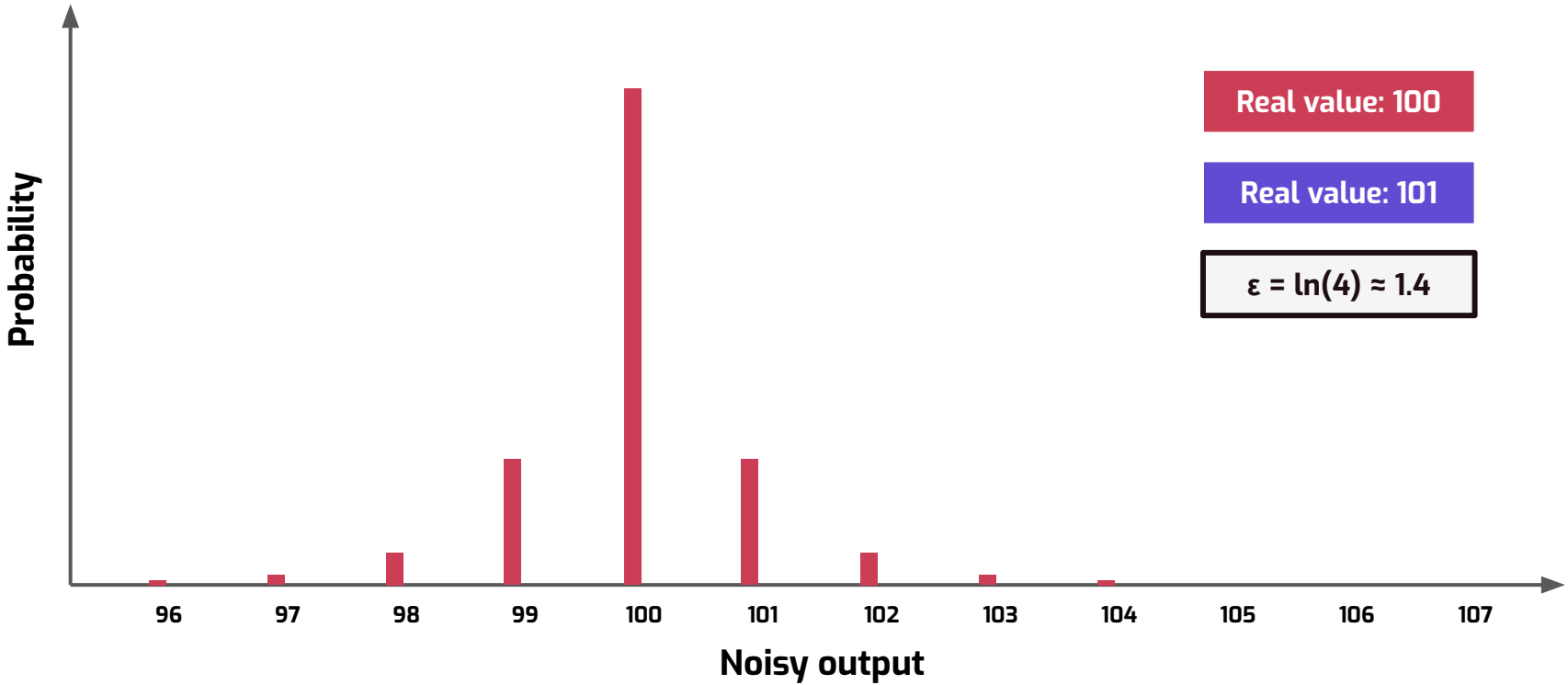
Example: making a count DP



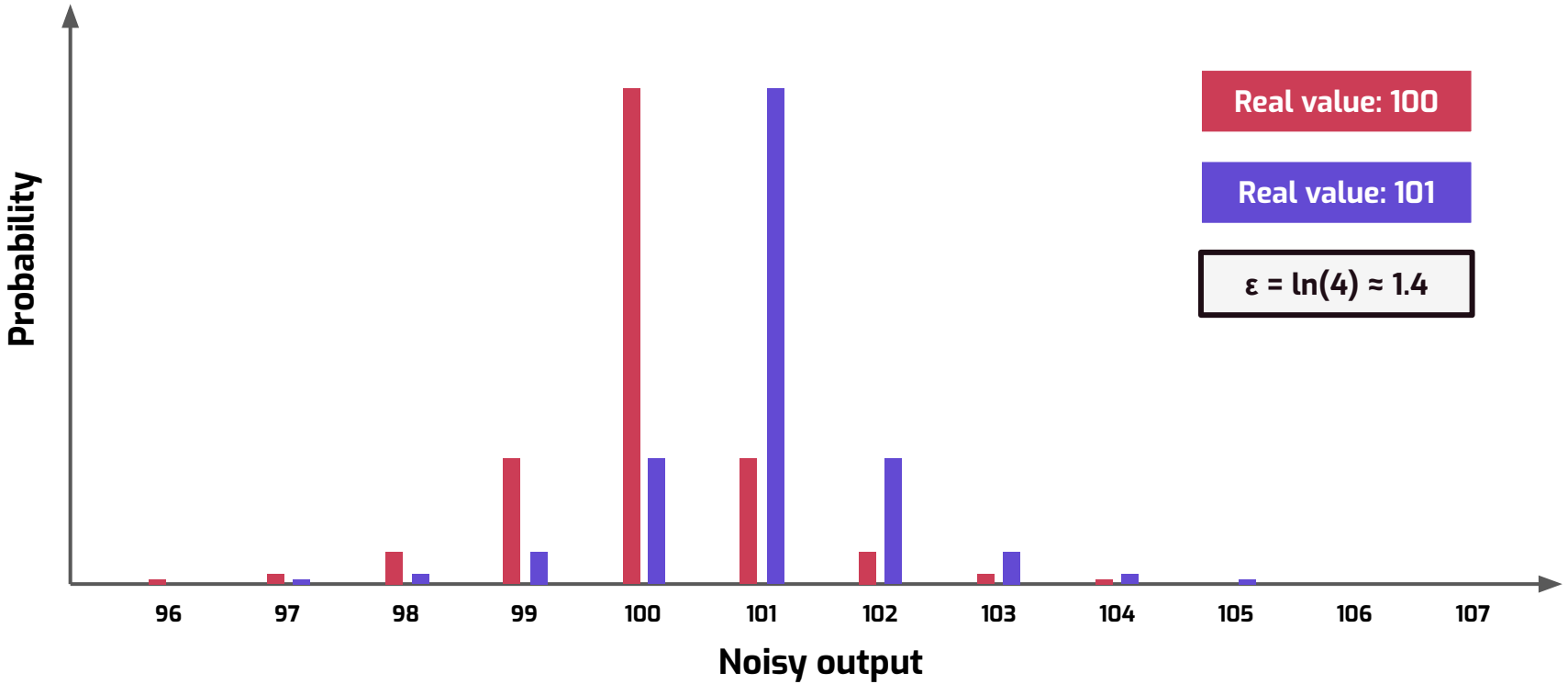
Example: making a count DP



Example: making a count DP



Example: making a count DP



DP count with Tumult Analytics

```
session = Session.from_dataframe(
  private_dataframe,
  source_id="my_data",
  privacy_budget=PureDPBudget(0.7),
)

query = QueryBuilder("my_data").count()

result = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.7),
)
```

- Loading the data in a Session, with a fixed privacy budget
- Building the query
- Evaluating the query with a portion of the total budget

DP count with Tumult Analytics

```
session = Session.from_dataframe(
  private_dataframe,
  source_id="my_data",
  privacy_budget=PureDPBudget(0.7),
)

query = QueryBuilder("my_data").count()

result = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.7),
)
```

```
result.show()

+-----+
|count|
+-----+
|  103|
+-----+
```

Privacy budgets in Tumult Analytics

```
session = Session.from_dataframe(
  private_dataframe,
  source_id="my_data",
  privacy_budget=PureDPBudget(0.7),
)

query = QueryBuilder("my_data").count()

result = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.8),
)
```

```
RuntimeError: Cannot
answer measurement
without exceeding
maximum privacy loss: it
needs 0.8, but the
remaining budget is 0.7
```

Privacy budgets in Tumult Analytics

```
session = Session.from_dataframe(
  private_dataframe,
  source_id="my_data",
  privacy_budget=PureDPBudget(0.7),
)

query = QueryBuilder("my_data").count()

result = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.4))

result_2 = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.4))
```

- Total privacy budget is 0.7
- We spend 0.4 once…
- … and twice.

Privacy budgets in Tumult Analytics

```
session = Session.from_dataframe(
  private_dataframe,
  source_id="my_data",
  privacy_budget=PureDPBudget(0.7),
)

query = QueryBuilder("my_data").count()

result = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.4))

result_2 = session.evaluate(
  query,
  privacy_budget=PureDPBudget(0.4))
```

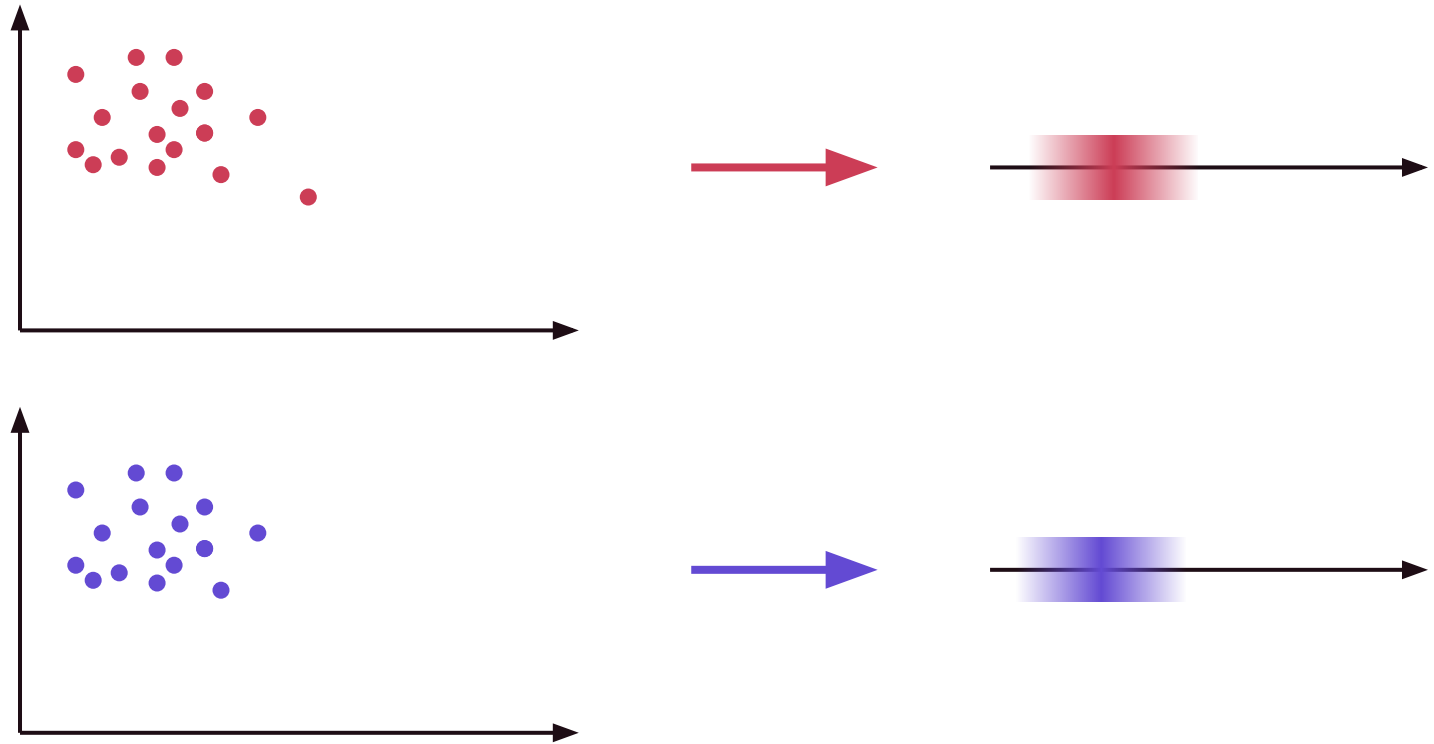
RuntimeError: Cannot answer measurement without exceeding maximum privacy loss: it needs 0.4, but the remaining budget is 0.3

Privacy budgets in Tumult Analytics

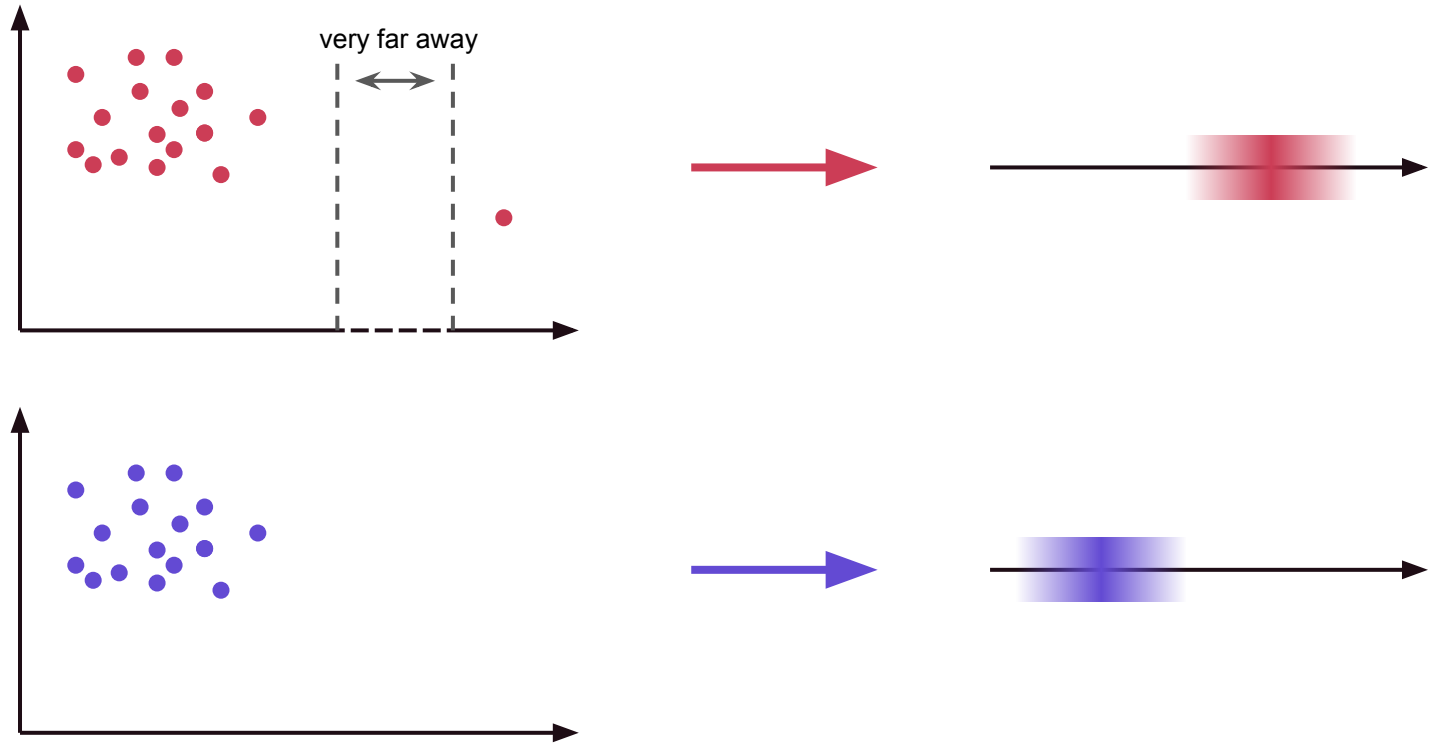
```
session = Session.from_dataframe(
  private_dataframe,
  source_id="my_data",
  privacy_budget=PureDPBudget(0.7),
)

query = ...

...
```

This is all we need to know to understand the full program's privacy guarantee!

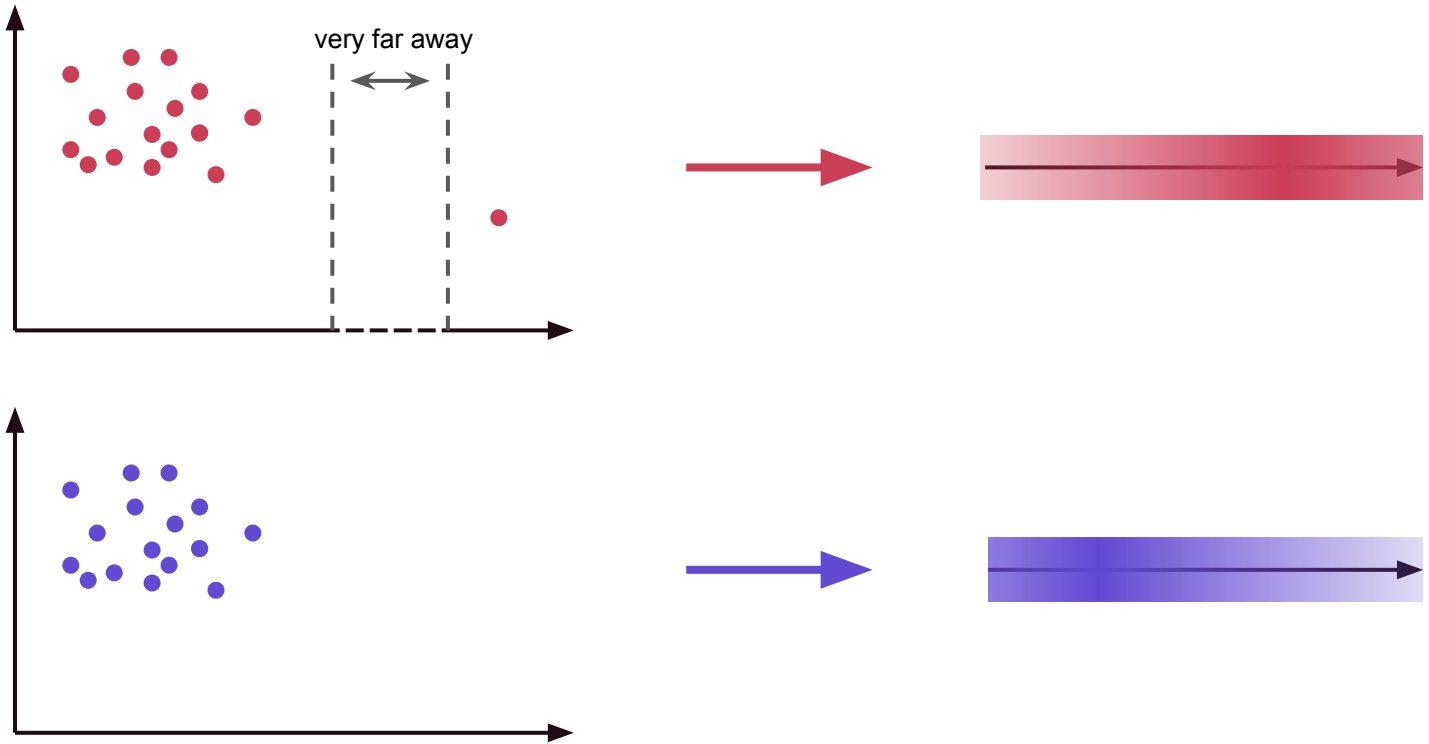
Clamping bounds



Clamping bounds



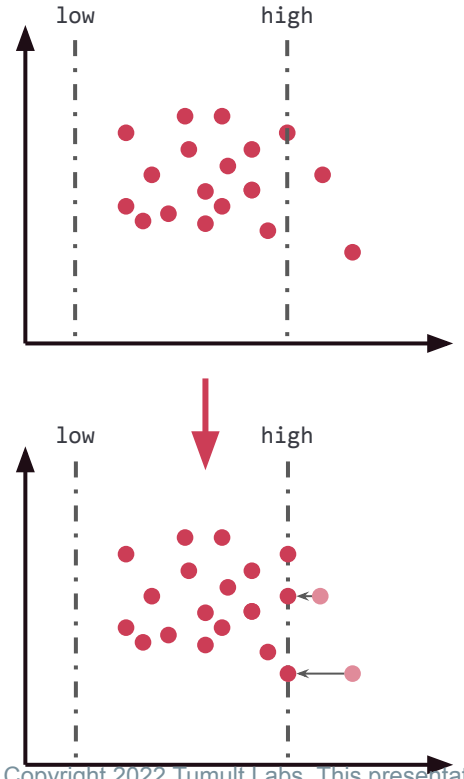
Clamping bounds



Clamping bounds

- We need to **scale** the noise according to the data magnitude…
- … which means we must **bound** the min/max values of the data.
- This is called ***clamping***.

Clamping bounds




```
query = (
  QueryBuilder("my_data")
  .average("age", low=0, high=100)
)

result = session.evaluate(
  query,
  privacy_budget=...,
)
```

Clamping bounds

How to choose clamping bounds in practice?

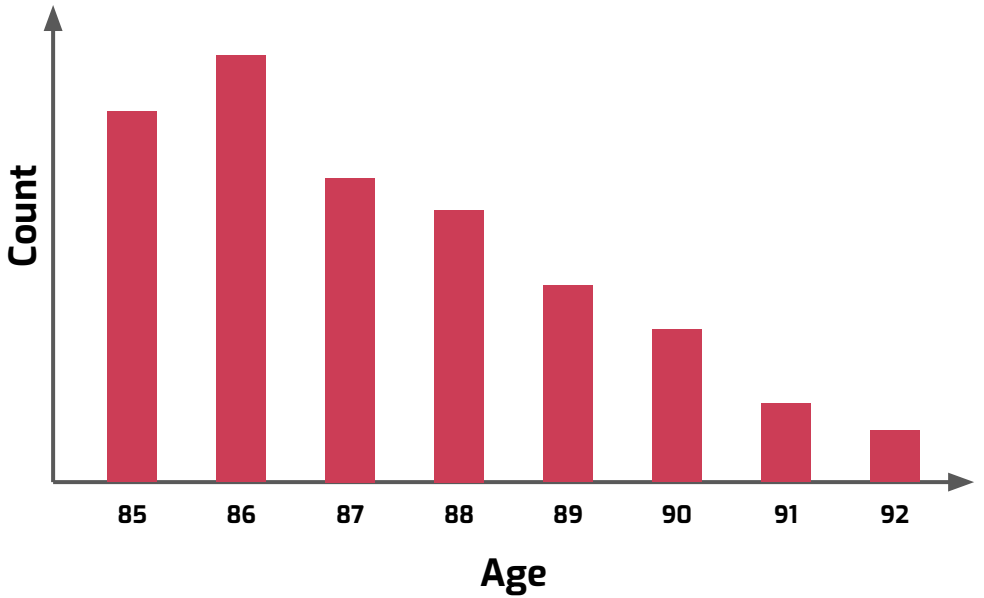
- General rule: large enough to clamp only a small (<5%) fraction of the data
- Taking e.g. min/max of the private data is forbidden!
- For more guidance & experimentation: see Webinar 3

Group-by queries

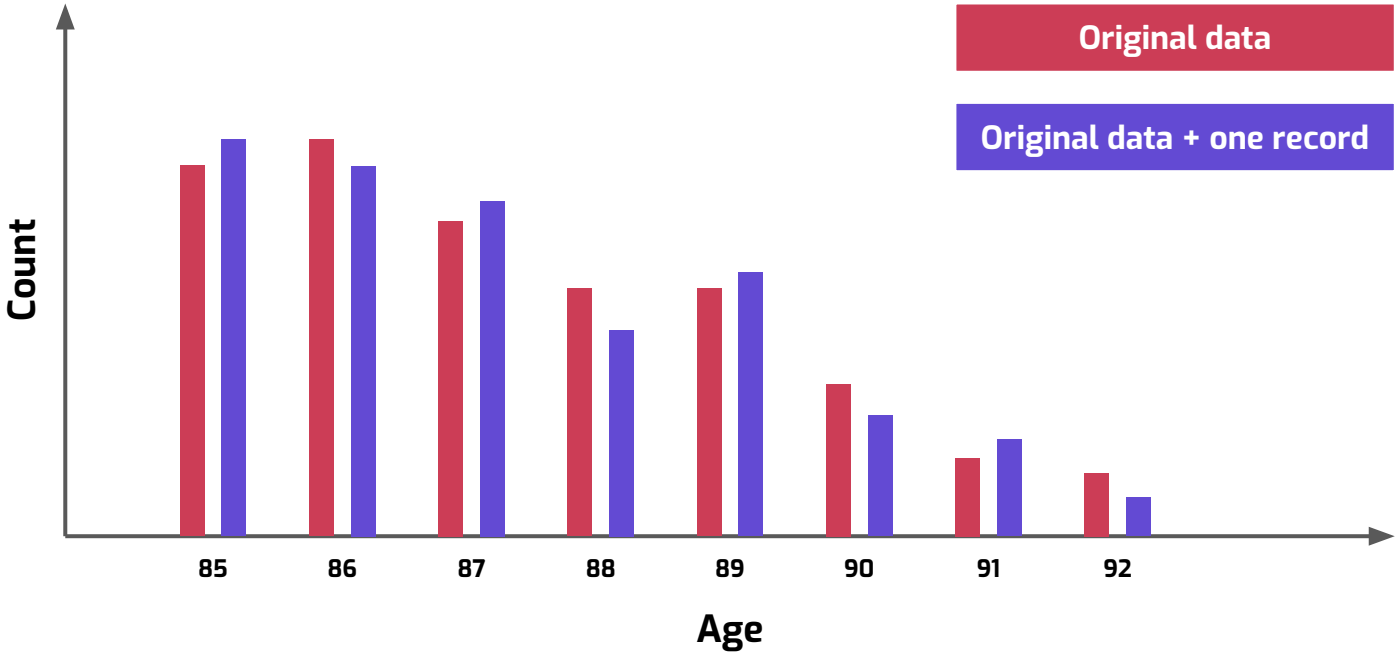
Let's compute a histogram!

In SQL, it's easy:

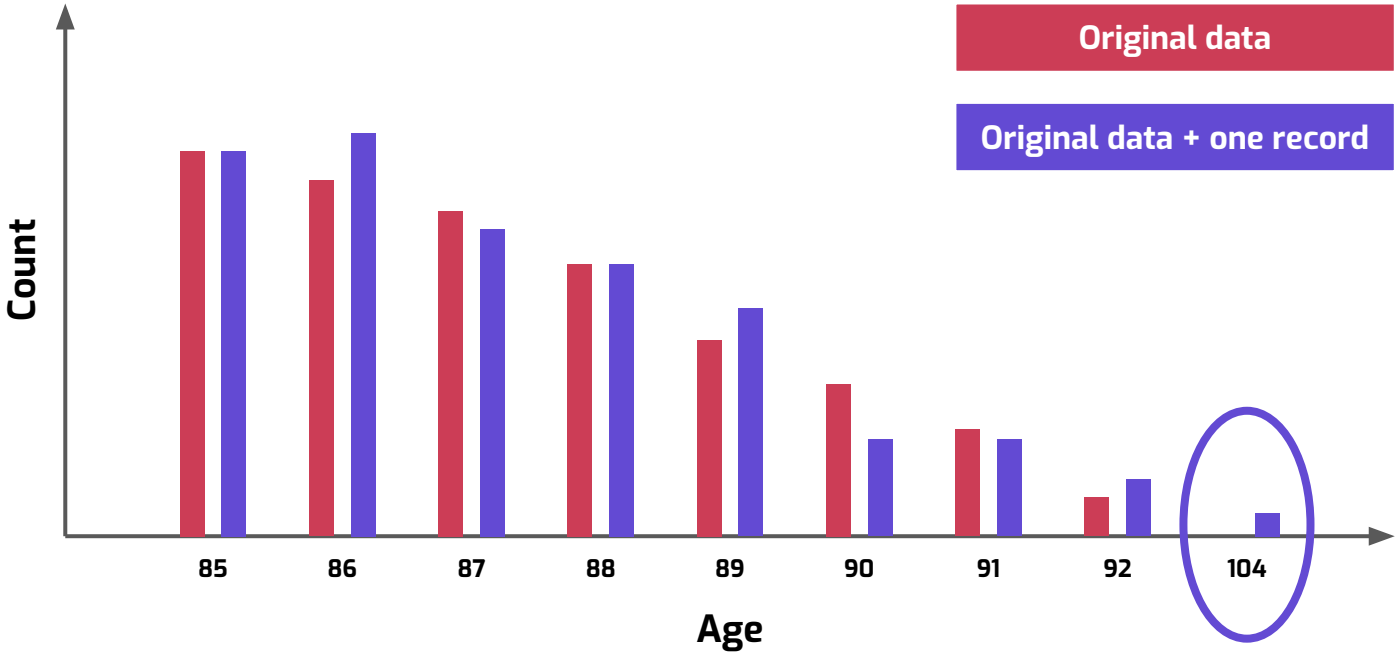
```
SELECT COUNT(*)
FROM data
GROUP BY age
```



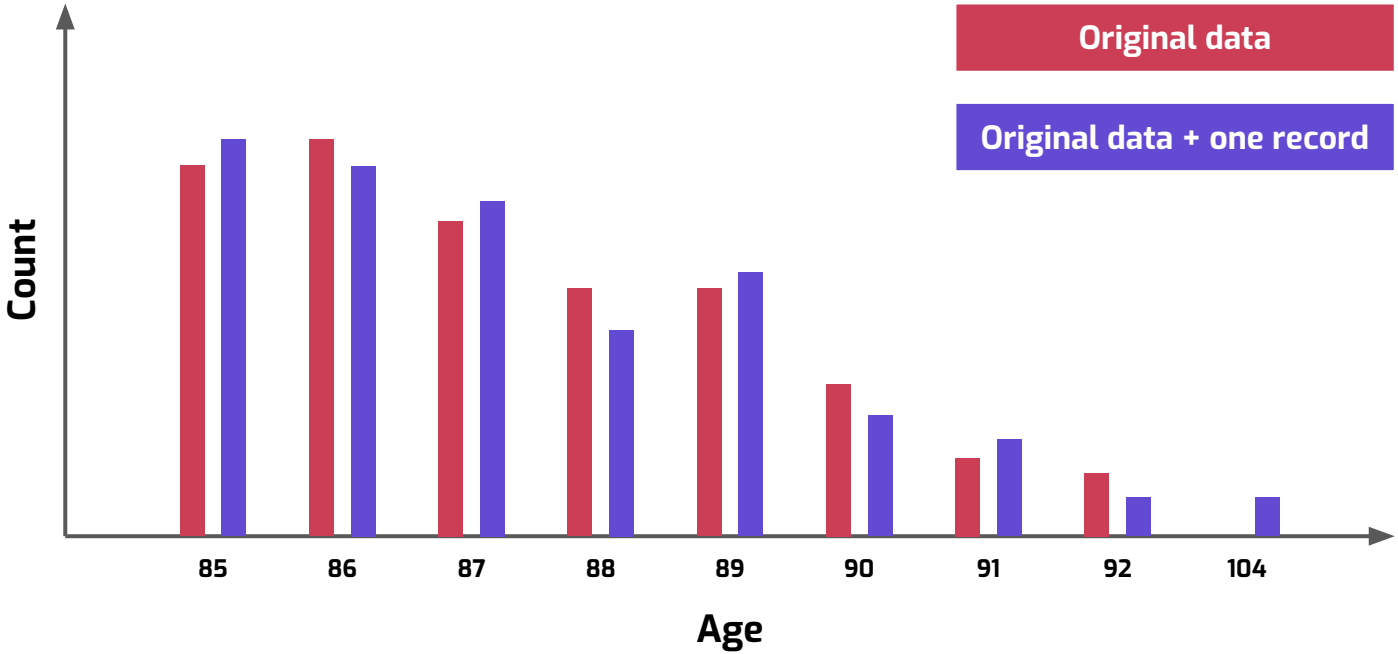
Group-by queries



Group-by queries



Group-by queries



Group-by queries

Let's compute a **DP** histogram!

In SQL, you'd need to add more info:

```
SELECT COUNT(*)
FROM data
GROUP BY age
USING KEYS [85, 86, ..., 92]
```

```
age_keys = KeySet.from_dict({
  "age": list(85, 93),
})

query = (
  QueryBuilder("my_data")
  .groupby(age_keys)
  .count()
)
```

Group-by queries

Tumult Analytics makes handling of group-by keys easy:

- Automatic duplicate removal
- Scaling to large (public!) datasets: `KeySet.from_dataframe(...)`
- Support for “structural zeroes” (impossible combinations)
- Cross-product (`keys_1 * keys_2`) and projection (`keys[“col”]`) operators

Recap

DP requires four changes to analyses:

- Noise addition
- Privacy budget tracking
- Clamping bounds
- Group-by keys specification

```
zip_codes = KeySet.from_dataframe({
  zip_codes_df,
})

query = (
  QueryBuilder("my_data")
  .groupby(zip_codes)
  .average(age, low=0, high=100)
)

result = session.evaluate(
  query,
  privacy_budget=PureDPBudget(1),
)
```

Your turn!

1. Follow the instructions in the notebook to set up the tutorial data & imports

2. Using a total budget of $\varepsilon = 5$, answer the following questions.
   a. How many library users older than 40 have PhDs?
   b. What is the average number of books borrowed by library users?
   c. What is the median age of library users, depending on gender and education level?
   d. How does gender influence the likelihood to list “Poetry” among one’s favorite literary genres?