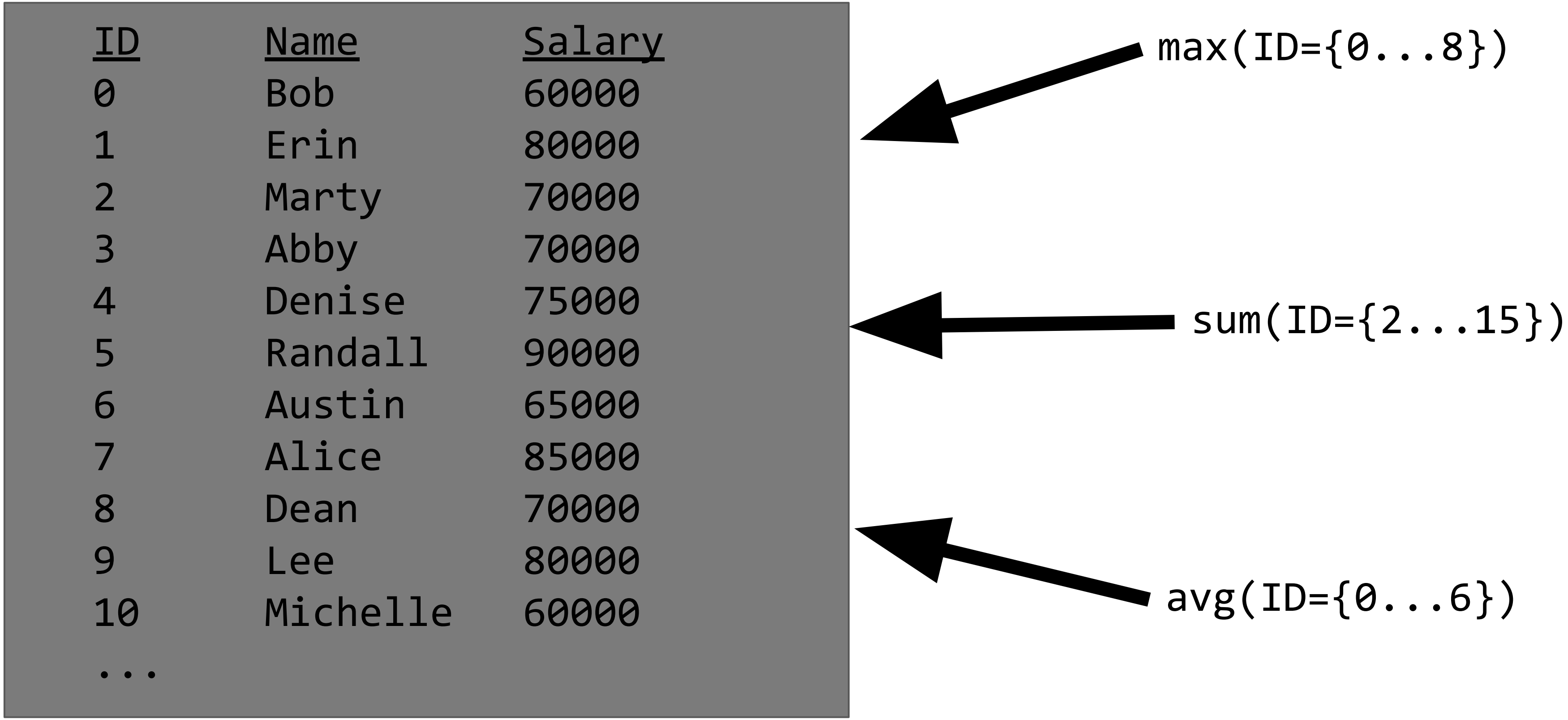
Simulatable Auditing

Serena Chen

ID	Name	Salary
0	Bob	60000
1	Erin	80000
2	Marty	70000
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

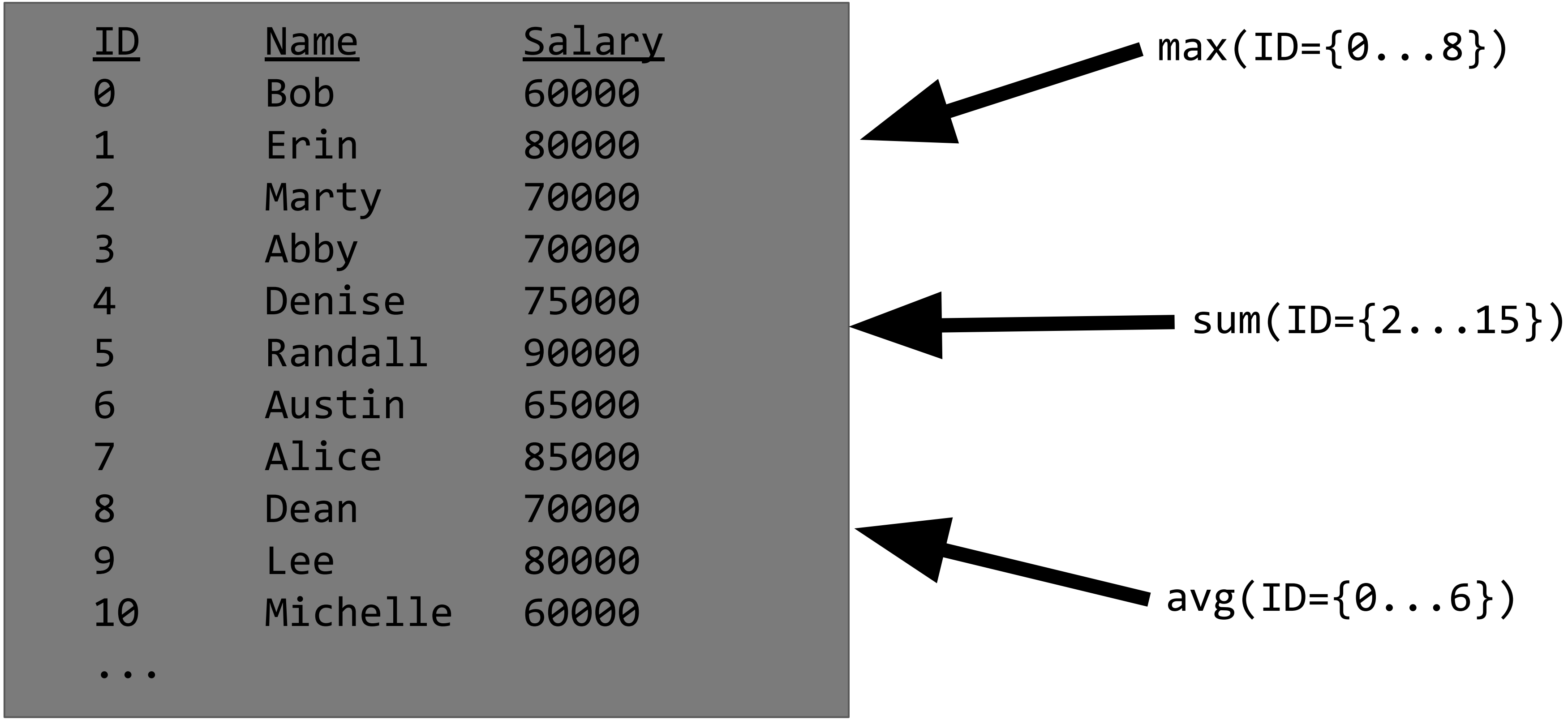


ID	Name	Salary
0	Bob	60000
1	Erin	80000
2	Marty	70000
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

max(ID={0...8})

sum(ID={2...15})

avg(ID={0...6})



ID	Name	Salary
0	Bob	60000
1	Erin	80000
2	Marty	70000
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
max(ID={0...8})
sum(ID={2...15})
avg(ID={0...6})
```

Statistical Database!

ID	Name	Salary
0	**Bob**	**60000**
1	**Erin**	**80000**
2	**Marty**	**70000**
3	**Abby**	**70000**
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
max(ID={0, 1, 2, 3})
    → 80000

max(ID={0, 2, 3})
    → 70000
```

Statistical Database!

ID	Name	Salary
0	Bob	60000
1	Erin	80000
2	Marty	70
3	Abby	[illegible]
4	Denise	[illegible]
5	Ran	000
6	[illegible]	65000
7	[illegible]	85000
8	n	70000
[illegible]	Lee	80000
[illegible]	Michelle	60000
...		

COMPROMISED

```
max(ID={0, 1, 2, 3})
     → 80000

max(ID={0, 2, 3})
     → 70000
```

Statistical Database!

Online Auditing

You have a statistical database, and for each query you can choose to **answer truthfully** or **deny**.

For a given set of previous queries and answers, how should you answer a new query?

Early auditing

Derive a giant set of subqueries based on a bunch of rules.

Logically models how an attacker would deduce knowledge from the set of queries.

Deny if you can deduce a `max({x}) = m`.

Compromise using `sum` and `max`

ID	Name	Salary
0	**Bob**	**60000**
1	**Erin**	**80000**
2	**Marty**	**70000**
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
sum(ID={0, 1, 2})
    → 210000

max(ID={0, 1, 2})
    → 80000
```

Compromise using `sum` and `max`

ID	Name	Salary
0	**Bob**	**70000**
1	**Erin**	**70000**
2	**Marty**	**70000**
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
sum(ID={0, 1, 2})
    → 210000

max(ID={0, 1, 2})
    →
```

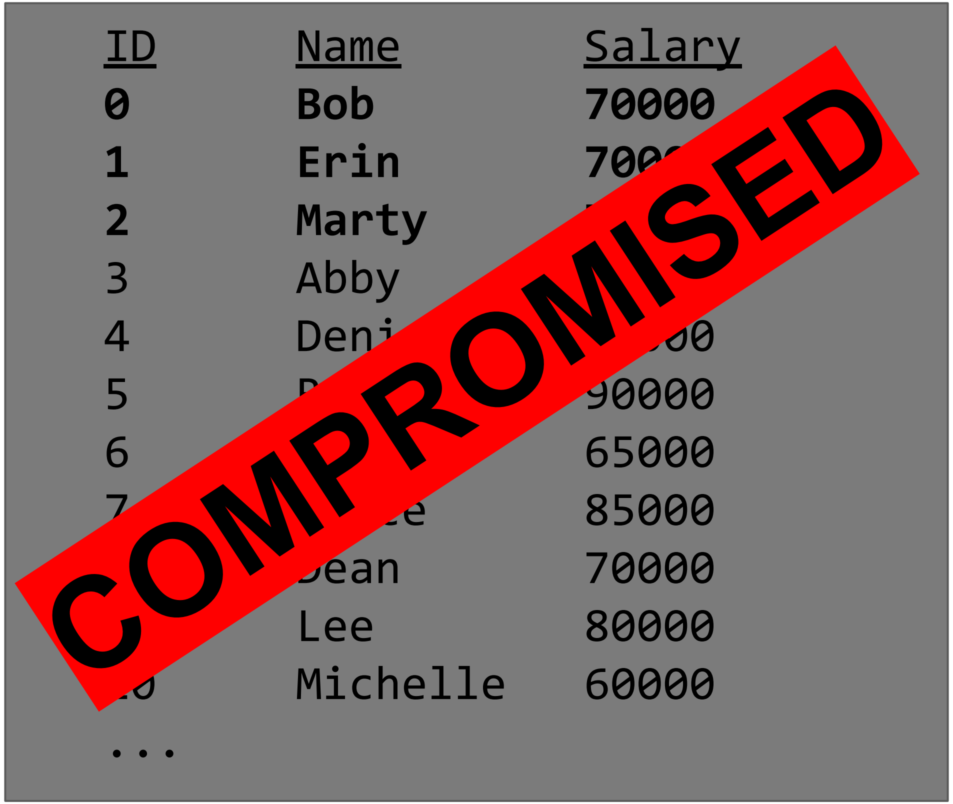
Compromise using `sum` and `max`

ID	Name	Salary
0	**Bob**	**70000**
1	**Erin**	**70000**
2	**Marty**	**70000**
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
sum(ID={0, 1, 2})
    → 210000

max(ID={0, 1, 2})
    → DENY
```

Compromise using `sum` and `max`



ID	Name	Salary
0	**Bob**	**70000**
1	**Erin**	[illegible]
2	**Marty**	[illegible]
3	Abby	[illegible]
4	[illegible]	[illegible]
5	[illegible]	90000
6	[illegible]	65000
7	[illegible]	85000
[illegible]	[illegible]	70000
[illegible]	Lee	80000
10	Michelle	60000
...		

COMPROMISED

```
sum(ID={0, 1, 2})
     → 210000

max(ID={0, 1, 2})
     → DENY
```

The only time this max query denies is when all three elements have the same value.

Simulatable Auditing

How can we design an auditing algorithm that doesn’t leak information in denials?

Simulatable Auditing

How can we design an auditing algorithm that doesn’t leak information in denials?

Don’t deny based on the actual answer. Deny based on whether there exists a possible answer that would compromise an individual.

Simulatable Auditing on Max Queries

User requests `max(k)`.

Simulatable Auditing on Max Queries

User requests `max(k)`.

With `M`: set of all answers to previous overlapping queries, generate the space of **all possible answers** to `max(k)`.

- All of `M`
- The smallest `m ∈ M` minus one
- The largest `m ∈ M` plus one
- The midpoint of every two consecutive $m_1,m_2 \in M$

Simulatable Auditing on Max Queries

User requests `max(k)`.

With `M`: set of all answers to previous overlapping queries, generate the space of **all possible answers** to `max(k)`.

- All of `M`
- The smallest $m \in M$ minus one
- The largest $m \in M$ plus one
- The midpoint of every two consecutive $m_1, m_2 \in M$

If any of those answers compromise the database, **deny**.

Compromise using `sum` and `max`

ID	Name	Salary
0	**Bob**	**60000**
1	**Erin**	**80000**
2	**Marty**	**70000**
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
sum(ID={0, 1, 2})
    → 210000

max(ID={0, 1, 2})
    →
```

Compromise using `sum` and `max`

ID	Name	Salary
0	**Bob**	**60000**
1	**Erin**	**80000**
2	**Marty**	**70000**
3	Abby	70000
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

```
sum(ID={0, 1, 2})
    → 210000

max(ID={0, 1, 2})
    → DENY
```

Compromise using max

ID	Name	Salary
0	**Bob**	**60000**
1	**Erin**	**80000**
2	**Marty**	**70000**
3	**Abby**	**70000**
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

Original Auditor:

```
max(ID={0, 1, 2, 3})
    → 80000

max(ID={0, 2, 3})
    → DENY

max(ID={0, 1, 2})
    → 80000
```

Compromise using max

ID	Name	Salary
0	**Bob**	**60000**
1	**Erin**	**80000**
2	**Marty**	**70000**
3	**Abby**	**70000**
4	Denise	75000
5	Randall	90000
6	Austin	65000
7	Alice	85000
8	Dean	70000
9	Lee	80000
10	Michelle	60000
...		

Simulatable Auditor:

```
max(ID={0, 1, 2, 3})
    → 80000

max(ID={0, 2, 3})
    → DENY

max(ID={0, 1, 2})
    → DENY
```