Synthesizing Analytical SQL From Computation Demonstration

Xiangyu Zhou[1], Ras Bodik[1], Alvin Cheung[2], Chenglong Wang[3]

[1]University of Washington,
[2]UC Berkeley,
[3]Microsoft Research

Querying databases with SQL

City	Quarter	Enrolled
A	1	100
A	2	200
A	3	400
B	1	300
B	2	150
B	3	150

How many people enrolled in city A each quarter?

```
Select City, Quarter, Enrolled
From T
Where City = "A"
```

City	Quarter	Enrolled
A	1	100
A	2	200
A	3	400

What's each city's total enrollment number?

```
Select City, Sum(Enrolled)
From T
Group By City
```

City	Sum(Enrolled)
A	700
B	600

The total enrollment number at the end of each quarter?

The moving average?

The quarters with highest enrollment number?

?

Basic SQL is not sufficient for complex analytical tasks

Analytical SQL: SQL + Analytical Function

The total enrollment number at the end of each quarter for each city?

City	Quarter	Enrolled
A	1	100
A	2	200
A	3	400
B	1	300
B	2	150
B	3	150

```
Select City,
         Quarter,
         CumSum(Enrolled) Over (Partition By City)
From T
```

City	Quarter	Cumsum
A	1	100
A	2	300
A	3	700
B	1	300
B	2	450
B	3	600

Argmax

N-th percentile

Cumulative sum

Rank

Moving average

Le

Programming by Example?

Analytical SQL is a rich and expressive language for analytical tasks

(but it's also hard to program with)

Synthesizing Analytical SQL From Input-Output Examples?

Find a query q *such that* $q(T) = E$

T

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

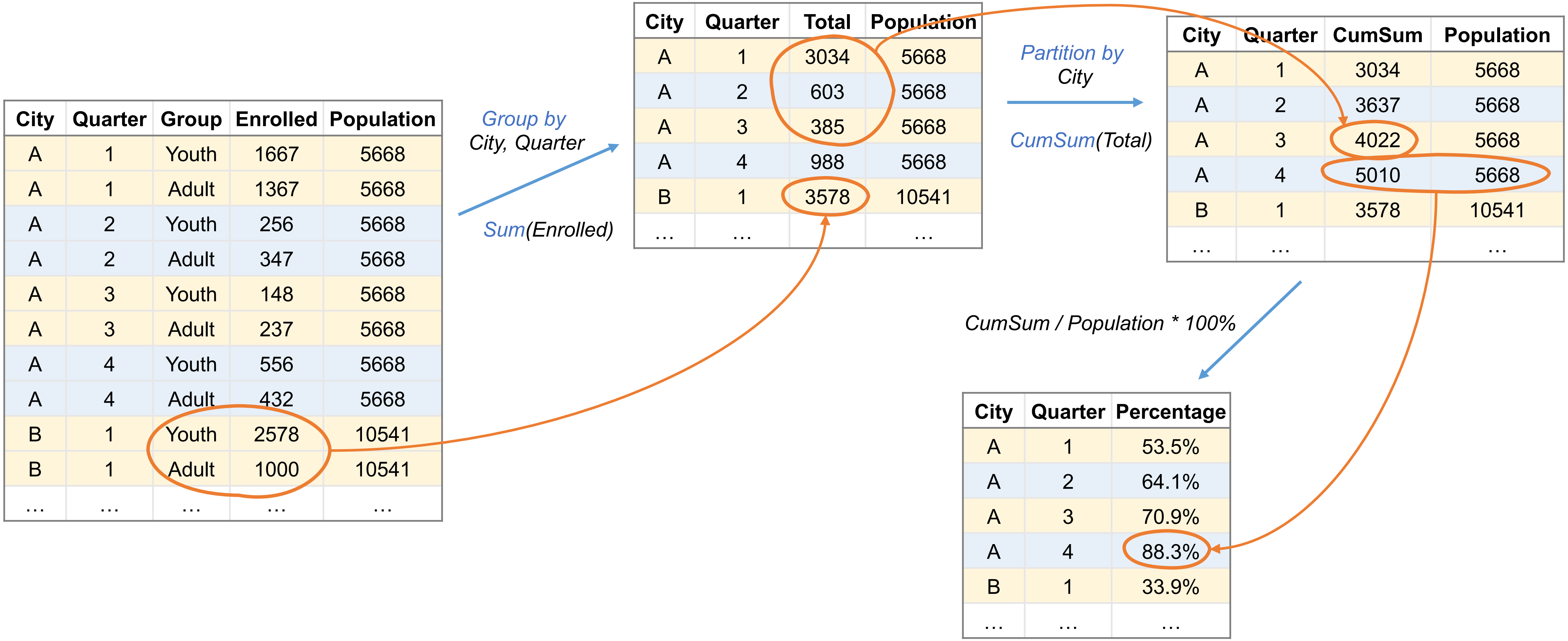
?

E

City	Quarter	Percentage
A	1	53.5%
A	2	64.1%
A	3	70.9%
A	4	88.3%
B	1	33.9%
…	…	…

Synthesizing Analytical SQL From Input-Output Examples?

Task: calculate the percentage of cumulative enrollment over city population



City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

Group by
City, Quarter

Sum(Enrolled)

City	Quarter	Total	Population
A	1	3034	5668
A	2	603	5668
A	3	385	5668
A	4	988	5668
B	1	3578	10541
…	…		…

Partition by
City

CumSum(Total)

City	Quarter	CumSum	Population
A	1	3034	5668
A	2	3637	5668
A	3	4022	5668
A	4	5010	5668
B	1	3578	10541
…	…		…

*CumSum / Population * 100%*

City	Quarter	Percentage
A	1	53.5%
A	2	64.1%
A	3	70.9%
A	4	88.3%
B	1	33.9%
…	…	…

Synthesizing Analytical SQL From Input-Output Examples?

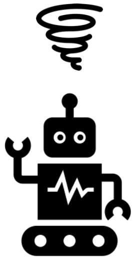
Problem 1: *creating the example requires a lot of manual calculation, and it is error-prone*

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

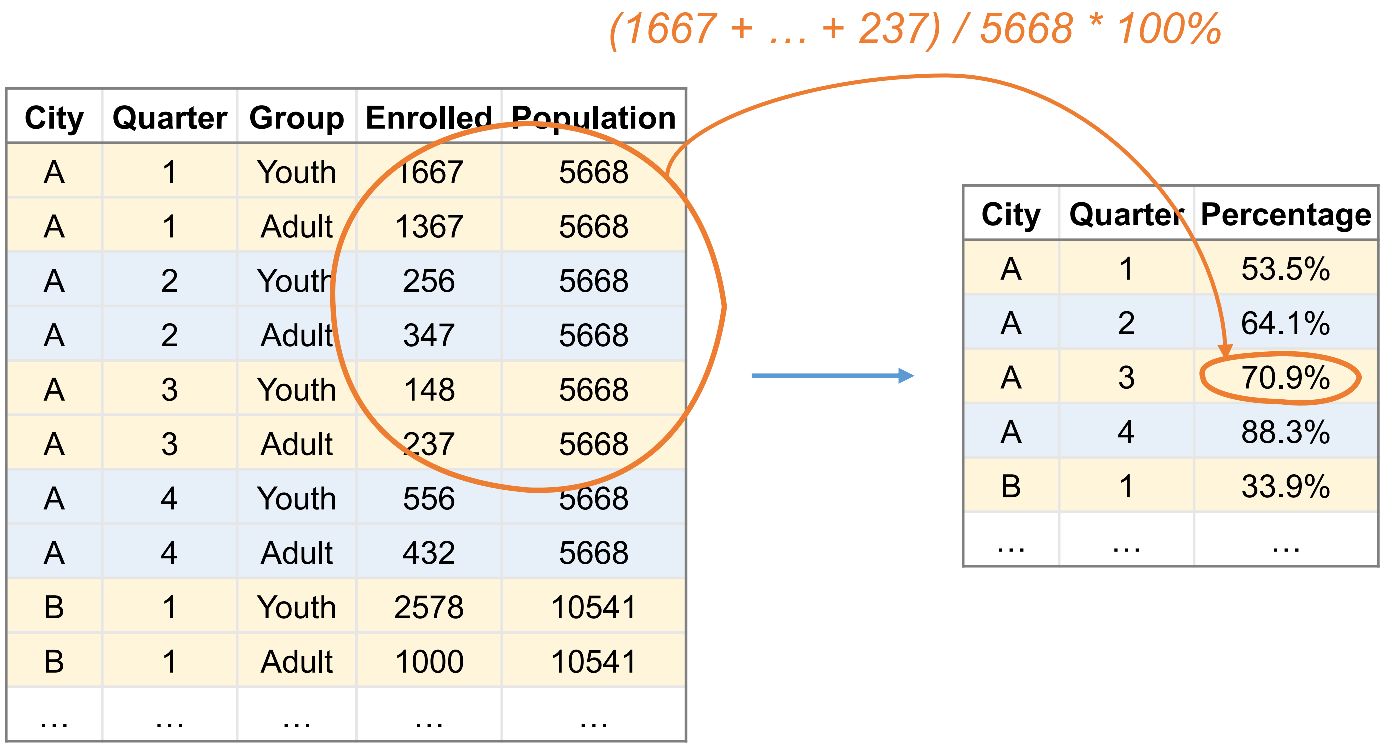
?

City	Quarter	Percentage
A	1	53.5%
A	2	64.1%
A	3	70.9%
A	4	88.3%
B	1	33.9%
…	…	…

Problem 2: *The example hides user intent, making the task ambiguous and hard to solve.*



Synthesizing Analytical SQL From ~~Input-Output Examples~~



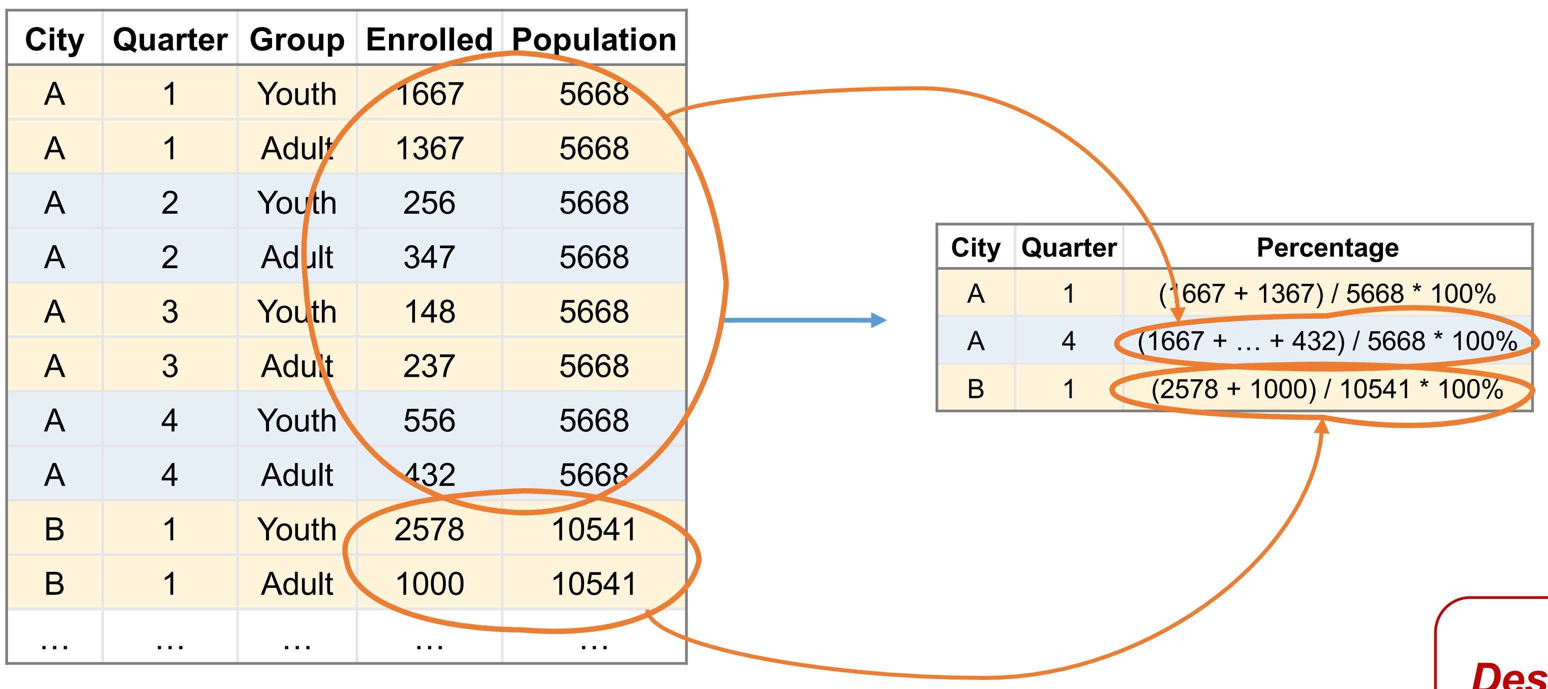
*(1667 + … + 237) / 5668 * 100%*

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

City	Quarter	Percentage
A	1	53.5%
A	2	64.1%
A	3	70.9%
A	4	88.3%
B	1	33.9%
…	…	…

Key: let the user demonstrate the computation process, not just the final value

Synthesizing Analytical SQL From ~~Input-Output Examples~~ Computation Demonstration



City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	4	(1667 + … + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%

***Design 1:** demonstrate the computation using cell-level expressions*

***Design 3:** support incomplete expressions*

***Design 2:** support partial output table – not all rows needs to be provided*

Synthesis Criteria for Computation Demonstration

T

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

Programming by Example
Given T and E, find q such that exists $q(T) = E$

Computation Demonstration
Given T and E, find q such that $q(T)$ is computationally consistent with E

***1.** Tracking of how each cell in $q(T)$ is computed*

***2.** Capturing the fact that E can be partial*

***3.** Handling incomplete expressions e.g., (1 + 2 + 3 + 4) generalizes (1 + … + 4)*

E

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	4	(1667 + … + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%

$q(T)$

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	2	(1667 + 1367 + 256 + 347) / 5668 * 100%
A	3	(1667 + 1367 + 256 + 347 + 148 + 237) / 5668 * 100%
A	4	(1667 + 1367 + 256 + 347 + 148 + 237 + 556 + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%
…	…	…

Synthesis Algorithm: Enumerative Search with Al Pruning

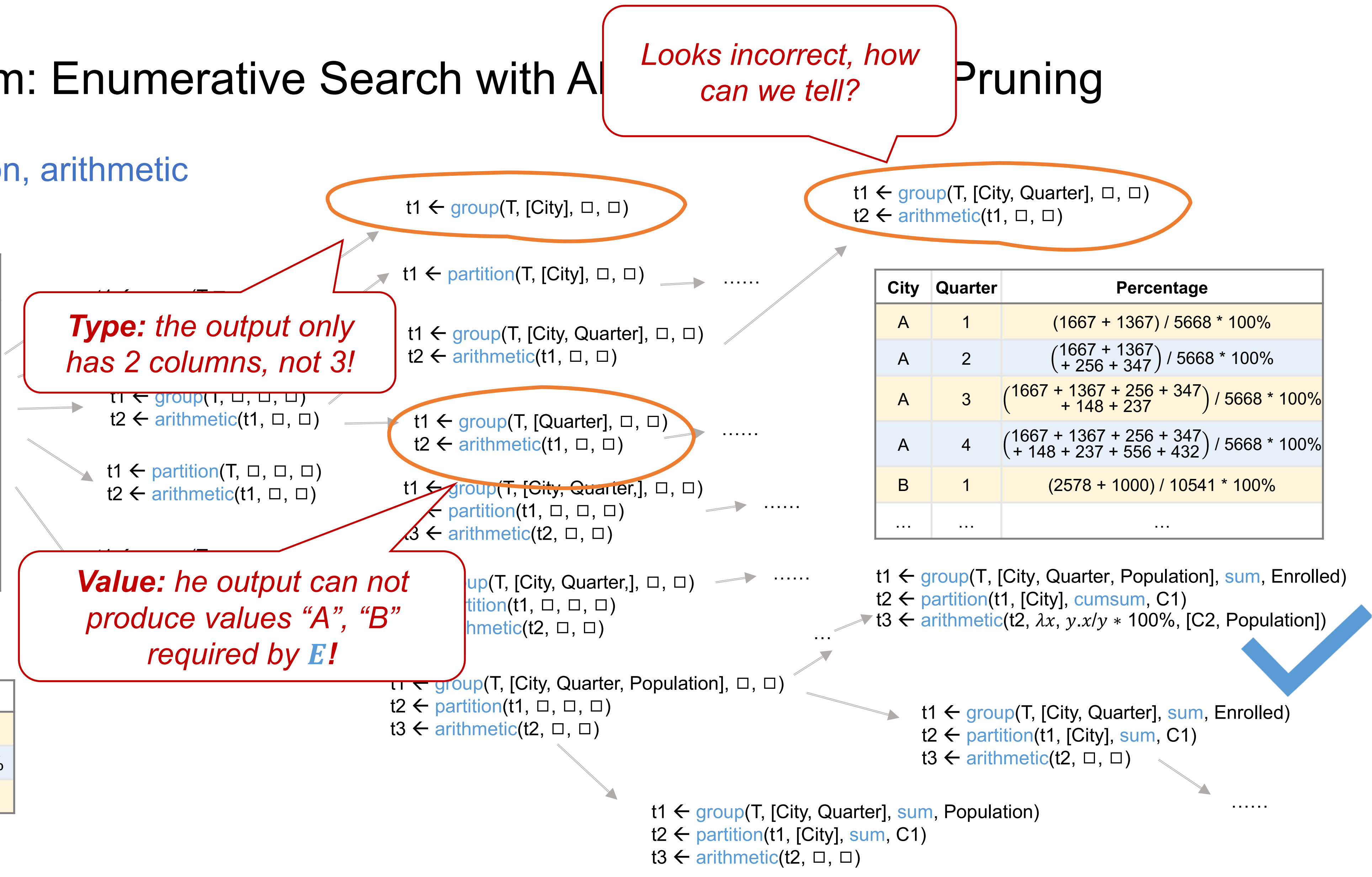
Operators: group, partition, arithmetic

T

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

E

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	4	(1667 + … + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%



Looks incorrect, how can we tell?

t1 ← group(T, [City], □, □)

t1 ← partition(T, [City], □, □) ……

t1 ← group(T, [City, Quarter], □, □)
t2 ← arithmetic(t1, □, □)

t1 ← group(T, [City, Quarter], □, □)
t2 ← arithmetic(t1, □, □)

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	2	(1667 + 1367 + 256 + 347) / 5668 * 100%
A	3	(1667 + 1367 + 256 + 347 + 148 + 237) / 5668 * 100%
A	4	(1667 + 1367 + 256 + 347 + 148 + 237 + 556 + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%
…	…	…

***Type:** the output only has 2 columns, not 3!*

t1 ← group(T, □, □, □)
t2 ← arithmetic(t1, □, □)

t1 ← group(T, [Quarter], □, □)
t2 ← arithmetic(t1, □, □) ……

t1 ← partition(T, □, □, □)
t2 ← arithmetic(t1, □, □)

t1 ← group(T, [City, Quarter,], □, □)
← partition(t1, □, □, □) ……
t3 ← arithmetic(t2, □, □)

***Value:** he output can not produce values "A", "B" required by **E**!*

up(T, [City, Quarter,], □, □) ……
tition(t1, □, □, □)
hmetic(t2, □, □)

t1 ← group(T, [City, Quarter, Population], □, □)
t2 ← partition(t1, □, □, □)
t3 ← arithmetic(t2, □, □)

…

t1 ← group(T, [City, Quarter, Population], sum, Enrolled)
t2 ← partition(t1, [City], cumsum, C1)
t3 ← arithmetic(t2, $\lambda x, y.x/y * 100\%$, [C2, Population])

t1 ← group(T, [City, Quarter], sum, Enrolled)
t2 ← partition(t1, [City], sum, C1)
t3 ← arithmetic(t2, □, □) ……

t1 ← group(T, [City, Quarter], sum, Population)
t2 ← partition(t1, [City], sum, C1)
t3 ← arithmetic(t2, □, □)

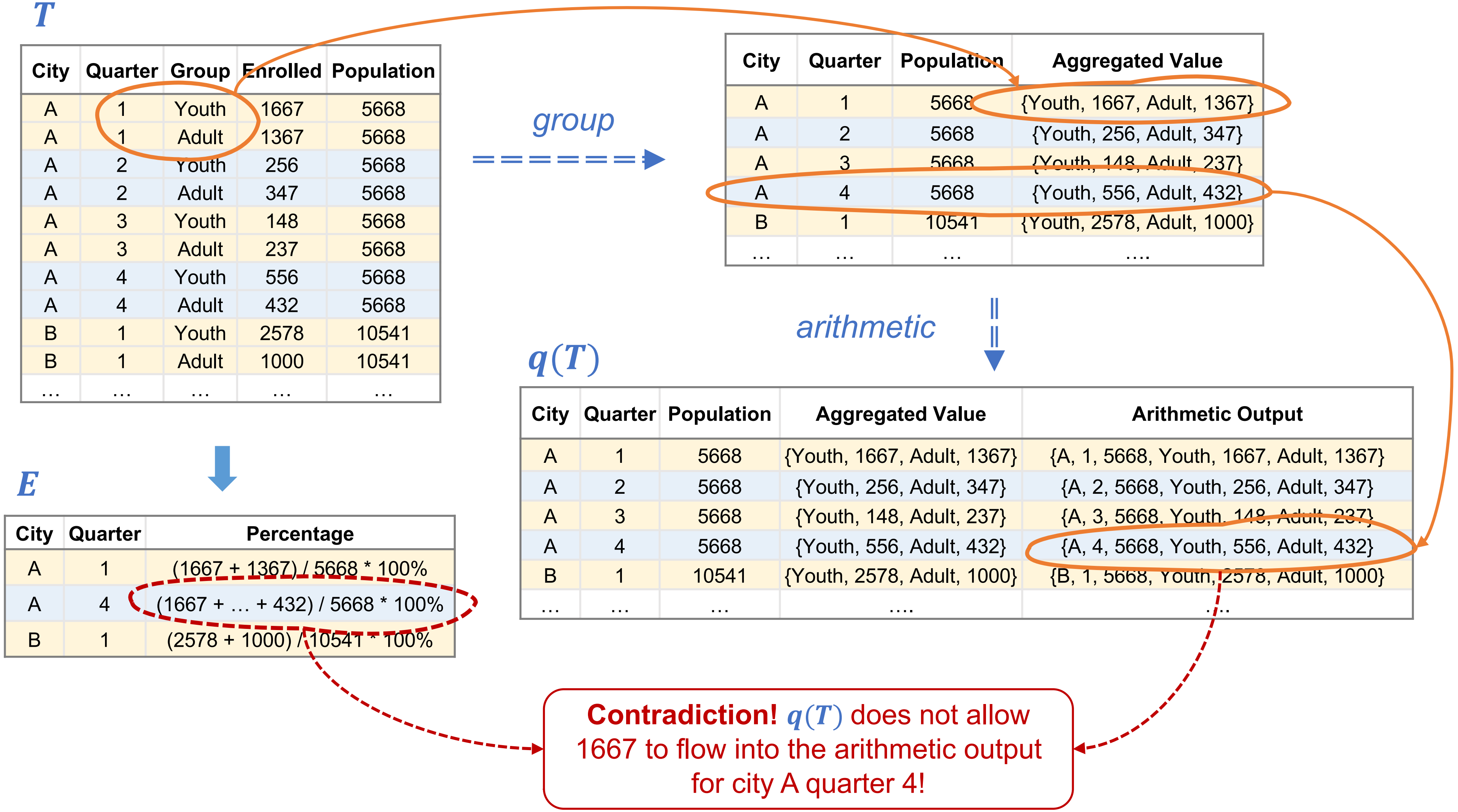
Pruning with ***Abstract Provenance Analysis***

q t1 ← group(T, [City, Quarter, Population], □, □)
t2 ← arithmetic(t1, □, □)

Observation
E demonstrates how values in T flows to E.

Let's analyze if $q(T)$'s data provenance (flow) is consistent with E

Over-approximates data provenance



T

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

E

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	4	(1667 + … + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%

City	Quarter	Population	Aggregated Value
A	1	5668	{Youth, 1667, Adult, 1367}
A	2	5668	{Youth, 256, Adult, 347}
A	3	5668	{Youth, 148, Adult, 237}
A	4	5668	{Youth, 556, Adult, 432}
B	1	10541	{Youth, 2578, Adult, 1000}
…	…	…	….

$q(T)$

City	Quarter	Population	Aggregated Value	Arithmetic Output
A	1	5668	{Youth, 1667, Adult, 1367}	{A, 1, 5668, Youth, 1667, Adult, 1367}
A	2	5668	{Youth, 256, Adult, 347}	{A, 2, 5668, Youth, 256, Adult, 347}
A	3	5668	{Youth, 148, Adult, 237}	{A, 3, 5668, Youth, 148, Adult, 237}
A	4	5668	{Youth, 556, Adult, 432}	{A, 4, 5668, Youth, 556, Adult, 432}
B	1	10541	{Youth, 2578, Adult, 1000}	{B, 1, 5668, Youth, 2578, Adult, 1000}
…	…	…	….	….

Contradiction! $q(T)$ does not allow 1667 to flow into the arithmetic output for city A quarter 4!

Synthesizing Analytical SQL From Computation Demonstration

***Spec:** Computation Demonstration*

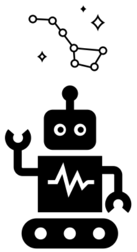
T

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

E

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	4	(1667 + … + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%

***Algorithm:** Enumerative search with abstract provenance analysis*



t1 ← group(T, [City, Quarter, Population], sum, Enrolled)
t2 ← partition(t1, [City], cumsum, C1)
t3 ← arithmetic(t2, $\lambda x, y. x/y * 100\%$, [C2, Population])

***Criteria:** $q(T)$ is computationally consistent with E*

Experiment: Synthesis Efficiency

Benchmarks

60 online posts + 20 queries extracted from TPC-DS

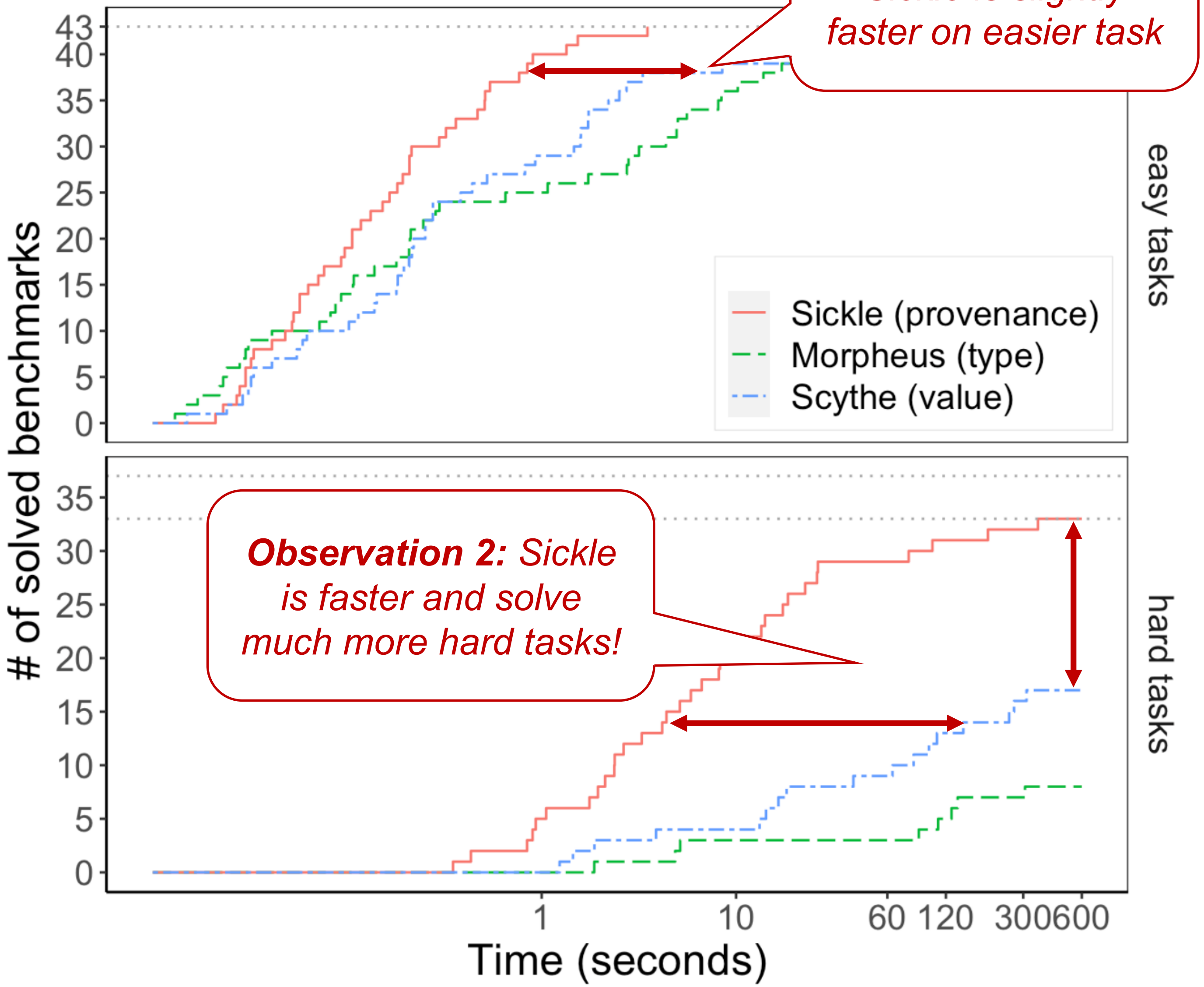
- *43 easy (1-3 operators)*
- *37 hard (4-6 operators)*

Pruning Abstraction

Type, Value, Provenance

Setup

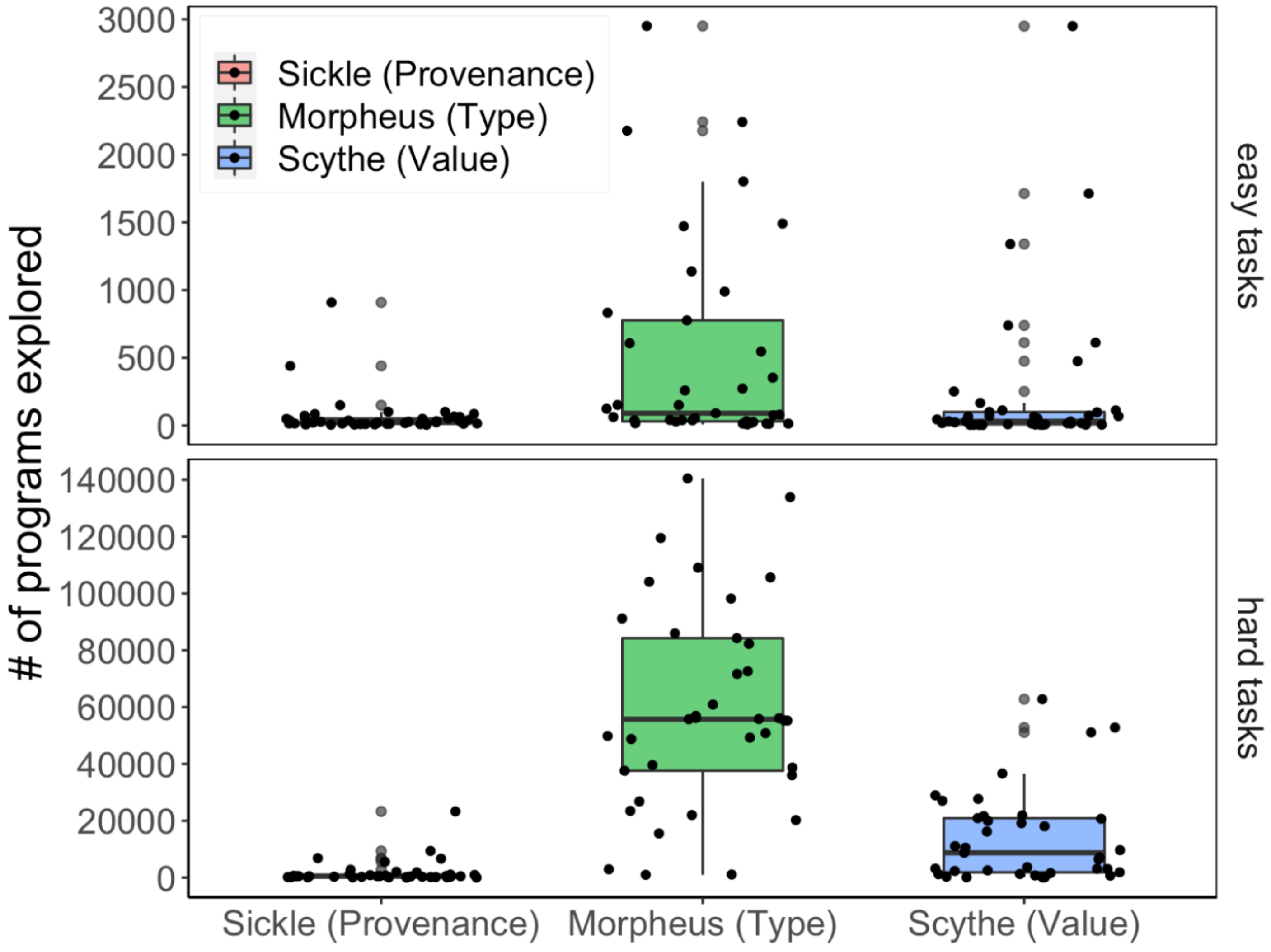
Demonstration generated by sampling from query output



Observation 1: *Sickle is slightly faster on easier task*

Observation 2: *Sickle is faster and solve much more hard tasks!*

Experiment: Number of Queries Visited



Observation 3:
Provenance abstraction significantly reduces programs to be visited

User Experience Study: *6 analytical tasks with 13 participants*

User Specification Method

- Concrete values (PBE)
- Computation demonstration without partial expression
- Computation demonstration

- ***Computation demonstration:***
  - *is faster to create for harder tasks, slower for easier tasks*
  - *increases user confidence*
  - *does not suit all operators (RANK, COUNT).*
- ***Partial expression***
  - *Reduces spec effort*
  - *some participants find it hard to master.*

Synthesizing Analytical SQL From Computation Demonstration

***Spec:** Computation Demonstration*

***Algorithm:** Enumerative search with abstract provenance analysis*

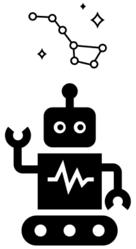
T

City	Quarter	Group	Enrolled	Population
A	1	Youth	1667	5668
A	1	Adult	1367	5668
A	2	Youth	256	5668
A	2	Adult	347	5668
A	3	Youth	148	5668
A	3	Adult	237	5668
A	4	Youth	556	5668
A	4	Adult	432	5668
B	1	Youth	2578	10541
B	1	Adult	1000	10541
…	…	…	…	…

E

City	Quarter	Percentage
A	1	(1667 + 1367) / 5668 * 100%
A	4	(1667 + … + 432) / 5668 * 100%
B	1	(2578 + 1000) / 10541 * 100%

t1 ← group(T, [City, Quarter, Population], sum, Enrolled)
t2 ← partition(t1, [City], cumsum, C1)
t3 ← arithmetic(t2, $\lambda x, y. x/y * 100\%$, [C2, Population])



***Criteria:** $q(T)$ is computation consistent with E*

***Position:** design new specifications for better user-synthesizer communication*

Paper link